A beginner's guide
Understanding C, C + + & Java For Dummies

Manjunath R

manjunath5496@gmail.com

"Talk is cheap. Show me the code."
Linus Torvalds
A Book For All Programmers
(Start programming in C, C++ & Java – no experience required…)

Abstract: This Book is designed for all C, C++ & Java beginners and is also for those who want to interact with the hardware in a more powerful way and this book excepts no formal education in programming and it is written in an easy, reveting and readable style ideally suited for self-study. Of course those who have already familiar with programming are likely to derive more benefits from this book. It does assume, however, a reader able to create at least a simple program. This book introduces you to the fundamental concepts of computer programming using C, C++ & Java languages. If you are just learning C, C++ & Java programs, this book will make an excellent companion to any C, C++ & Java tutorial and serve as a source of knowledge to your specific questions. And, by reading this book, you'll have a broad, basic knowledge of C, C++ & Java languages. This book is for all programmers, whether you are a novice or an experienced pro. The beginner will find its carefully paced discussions and many examples especially helpful.
[Manjunath R. **Understanding C, C + + & Java For Dummies.** *Academ Arena* 2016;8(4s): 1-229]. (ISSN 1553-992X). http://www.sciencepub.net/academia. 1. doi:10.7537/marsaaj0804s1601.

Keywords: C; C++; Java; software; hardware; languages

"What Will Understanding C, C++ & Java Do for Me?"

C, C++ & Java programs are commonly referred to as software and this software is essential to a fast and obedient, smart processing unit called computer because it controls everything the computer does (i.e., performance of a suite of computer operations and storage of the results in its memory can be manipulated by using a programming language such as C, C++ & Java).

I

Operating system: A well-defined set of instructions in the form of statements that is installed into the computer which provide instructions for computer how to operate (i.e., how to receive the raw data through input devices (like key board, mouse etc.), process the input data through processing device called CPU (Central Processing Unit) and store the processed data (in information storage devices like hard disks) and display the processed data through output devices (like monitor, printer etc.)). A well-defined instruction is called a code and a well-defined set of instructions constitute a program (i.e., compilation of codes gives a program). For example: word is a code and a paragraph is a program (i.e., compilation of words gives a paragraph).
Examples of Operating system (a well-defined set of instructions that is installed into the computer which provide instructions for computer how to operate) are: DOS (Disk operating system developed by Bill Gates and Paul Allen in 1980 for IBM PCs), Linux (operating system developed by Linus Torvalds at the University of Helsinki with the assistance of developers around the world), Windows NT, 95 & 2000 (developed by Microsoft corporation for PC), UNIX (developed by AT&T Bell Laboratories, Murray Hill, New Jersey) etc.

II

Drivers: A well-defined set of instructions (what we call programs or software) that is installed into computer and stored in the form of files in the computer that allows the computer to communicate with its hardware components (i.e., hardware components like mouse, key board, printer etc.). Without drivers, the computer cannot communicate with its hardware components – as a result a mouse, keyboard, or a printer won't work properly.

III

Server: If we type a request (a request to search information about computer) in the search engine websites (like Google or Firefox) in the web browser – the request is sent from the browser to the server – a system that acts like a data center from where the required information is taken and sent to the browser and is displayed in the web browser. Examples of server are: IIS (Internet Information server LATTER NAMED Internet Information service) – a web server developed by Microsoft corporation, Apache HTTP (HTTP mean Hyper Text Transfer Protocol) – a web server developed by Robert McCool at the national center for supercomputing applications (university of Illinois, Urbana-Champaign) – to provide web hosting service.

IV

Hosting: Host is a system that contains information and this information can be accessed by computer users by a means of internet. This process is called hosting.

V

IP address or Internet Protocol address: Just like every house on a street has a postal address which helps the post man to find that house on a street, every computer connected to internet has an Internet Protocol address or IP address which helps the other computers to find that computer on the network.
Suppose A B, C, D, E, F and G – are the computers connected to each other by means of internet (i.e., they are on the network). If computer A has not assigned any IP address, then users at computers B, C, D, E, F and G cannot send any email or other data to user at computer A or user at computer A cannot receive any email or other data from the users at computers B, C, D, E, F and G by a means of internet.

IP address is of four types:
Public IP address and Private IP address
Static and dynamic IP address
Static IP address → permanent IP address
Dynamic IP address → temporary IP address (exist only for a limited time i.e., IP address leased for a limited time).
Public and Private IP address:

Amazon organization is assigned an IP address IP_A and Google organization is assigned an IP address IP_G. And the systems (1, 2, 3, 4, 5…..etc.) within the Amazon or the Google organization are assigned an IP addresses IP_1, IP_2, IP_3 … etc.

IP_A and IP_G imply public IP addresses

IP_1, IP_2, IP_3 … etc. implies private IP addresses
Which means: Public IP address is used for external communication (i.e., used for the communication between the Amazon and the Google organization) and Private IP address is used for internal communication (i.e., used for communication between the systems within the Amazon or the Google organization).

VI

Domain name: If we type www.google.com (which is called the domain name) in the browser, then the domain name is sent to DNS (domain name system) where the domain namewww.google.com is converted to IP address 74.125.224.72 (because website / web pages are only identified by their IP address in the server) and this IP Address is sent to the web server (a system that acts like a data center from where the required information (i.e., web page of google.com) is taken and sent to the browser and the www.google.com web page is displayed in the web browser). If you type the IP address in the browser, then DNS is not required. For human convenience (difficult remember numbers, for example: www.google.com is domain name, IP address is 74.125.224.72. Because it is difficult to remember 74.125.224.72 so www.google.com is preferred).

VII

ASP.NET:
ASP → Active Server Page
ASP.NET (Active Server Page Network Enabled Technology) is a technology developed by Microsoft corporation using the languages -- C#, Visual Basic. Net, J script & J# -- to build dynamic web pages / websites and web applications.

Dynamic web page contains information (say date, month or year or time zone of the day) change automatically daily without a developer editing its source codes while static web page contains information (say date, month or year or time zone of the day) cannot change automatically daily without a developer editing its source codes.

Virtual Memory: If the RAM (i.e., Random Access Memory) is full and it is running out of space available for storage of further information and there is no access to store further information, the idea of extending memory by using disk is called virtual memory (i.e., the further information is stored in disk and retrieved when required). This process is called paging or swapping.

VIII

C

A high level language (which uses alphabets, digits, punctuations and some special symbols and cannot be executed directly without being converted into machine level language (the language which uses only 0 and 1)) developed by a man named Dennis Ritchie (in 1970s at Bell Telephone laboratories (now named AT & T Bell laboratories), Murray Hill, New Jersey, using the two early programming languages-- Basic combined programming language and BASIC (Beginner's All-purpose Symbolic Instruction Code) language), used in the development of operating systems like LINUX, UNIX, because of its reliability, simplicity and easy to understand, easy to use, write, modify and debug and quick to learn.

Process of C program execution: A C program:

```
#include<stdio.h>
main()
{
printf("Hello, crazy world!");
}
```

is written in C editor and is saved as source program and this source program is sent to the C compiler where the source program is compiled (i.e., the program is entirely read and translated to instructions the computer can understand i.e., machine understandable/ readable language i.e., to machine code sequence of 0's and 1's). If the C compiler finds any error during compilation, it provides information about the error to the user. The programmer has to review code and check for the solution. And if there are no errors the translated program is stored in computers main memory as object file and the program is executed and

```
Hello, crazy world!
```

is displayed on the screen. C is case sensitive language: only lower case letters (or small letters) must be used. Capital letters (or upper case letters) must be avoided to prevent the display of error on the screen (For example: If the statement PRINTF("Hello, crazy world!"); is written instead of printf("Hello, crazy world!"); or MAIN() is written instead of main(), compilation Error will be displayed on the screen). And if we forget to end each statement within the curly braces{} or each statement within the body of the main function

```
main()
{
```

} with a semicolon (;), then the compilation Error will be displayed on the screen. There should be no space between main and the parentheses () and no space inside the parentheses () to prevent the display of compilation error on the screen.

#include <stdio.h>→if we type a program (a well-defined set of instructions in the form of statements) in C editor, (as said earlier) the program is saved as source program and this source program is sent to the C compiler where the source program is compiled i.e., it is translated into machine level language i.e., into a machine code sequence of 0's and 1's (because computer can understand only machine level language). The statement #include<stdio.h> tells the complier to include the text from the file stdio.h (which is already present in the operating system) before it

translates or complies the program into a sequence of 0's and 1's. stdio means standard input output and stdio.h means standard input output header file (stdio.h comprises standard input output functions like scanf, printf etc. — note: scanf is an input function and printf is an output function and it is included into the C program by writing the statement #include <stdio.h>). #include tell the compiler to include the contents of the file stdio.h before compilation. If a program is written without the statement #include<stdio.h>, then the C compiler can't compile and a compilation error is displayed on the screen.

Note: We can also write #include "stdio.h" instead of #include <stdio.h> but sometimes compiler will flag error message. So the statement #include <stdio.h> is generally preferred.

main() →After the compilation of the source program, the translated (or the complied) program is stored in the computer's memory as object file and the program is executed. The program begins its execution with the function main() (which is called the user defined function (function defined by the user) – the main function -- the entry point of the program execution i.e., the function from where the execution of C program begins) . The left curly brace "{" implies the beginning of the main function and the right curly brace "}"implies the end of the main function
main() → main function
main()
{

} → body of the main function within which the sequence of instructions in the form of statements i.e., the program is written and executed.
Note: if a program begins its execution with main function "main()", it takes the control of the computer from the operating system. And after the complete execution of the program, the execution is terminated and the function main() returns back the control to the operating system. Semicolon: program is a well-defined set of instructions and each well-defined instruction (in the form of a statement) is ended by a semicolon (which is C language punctuation — like a period in English i.e., in an English paragraph each sentence is ended by a full stop which tells that one sentence ends and another begins, semicolon implies that one instruction (or statement) ends and another begins).
printf → output function of the C language which make provision to print the output on the screen. The letter f in the word printf stands for formatted.
The sentence / text Hello, crazy world! should be enclosed by the double quotation marks (" ") and should be written inside the parentheses of the printf function i.e.,
printf("Hello, crazy world!")
otherwise the compilation error will be displayed on the screen.
The statement
printf("Hello, crazy world!");
make provision to display the output:
Hello, crazy world!
on the screen.

Note: if " " is used instead of " ", Error will be displayed on the screen
The statement printf("Hello, crazy world!"); will not display any error on the screen.
The statement printf("Hello, crazy world!"); will display error on the screen.
Past few years back, the statement return(0); was included in the body of the main function i.e.,
{
printf("Hello, crazy world!");
return(0);
}
But now due to the advancement of technology and emergence of advanced online compilers like CodeChef (www.codechef.com/)
&
Coding Ground – TutorialsPoint (www.tutorialspoint.com/codingground.htm)
-- without the statement return(0); the program is compiled and executed without flag of any error on the screen. However, as the execution encounters the statement return (0); the execution stops and the main function ends at "}" and the main function returns back the control to the operating system.

Note:

If the statement return (0); is replaced by the statement
return 0;
or
return (1);
or
return(-2);
or
return;
there will be no change in the output on the screen (and no error will be flagged or displayed on the screen) i.e., for the programs

(a)
```
#include<stdio.h>
main()
{
printf("Hello, crazy world!");
return 0;
}
```

(b)
```
#include<stdio.h>
main()
{
printf("Hello, crazy world!");
return (1);
}
```

(c)
```
#include<stdio.h>
main()
{
printf("Hello, crazy world!");
return (-2);
}
```

(d)
```
#include<stdio.h>
main()
{
printf("Hello, crazy world!");
return;
}
```

The output on the screen is:
Hello, crazy world!
i.e., there will be no change in the output on the screen.

Program 1.1

C program to print the word “hello Bill Gates” on screen

```
#include<stdio.h>
main()
{
printf("hello Bill Gates");
}
```

The output on the screen:
hello Bill Gates

Even if
main(void) is written instead of main()
int main is written instead of main()
void main is written instead of main()
main(computer) is written instead of main()
main(comp2016) is written instead of main()
No error will be displayed on the screen.
hello Bill Gates will be outputted on the screen.
But if
main(2016comp) is written instead of main(comp2016)
Error will flagged on the console screen.

Program 1.2

C program to print
*
*****
*****
*****
*****
on screen

```
#include<stdio.h>
main()
{
printf("\n          *         ");
printf("\n    ***** ");
printf("\n    ***** ");
printf("\n    ***** ");
printf("\n    ***** ");
}
```

The output on the screen:
*
*****
*****
*****
*****
If new line \n is not included in the above program then the output on the screen is:
****

Write a program to print the following outputs:

(a)

*
****
****
****
*
(b)

****
* *
* Hello World! *

```
* *
***************
```

(c)

Braces come in pairs!
Comments come in pairs!
All statements end with a semicolon!
Spaces are optional!
Must have a main function!
C is done mostly in lowercase. It's a case-sensitive language

Answers:

```
#include<stdio.h>
main()
{
printf("\n              *        ");
printf("\n           ****    ");
printf("\n      ******* ");
printf("\n           ****       ");
printf("\n              *          ");
}
```

```
#include<stdio.h>
main()
{
printf("\n                ****************              ");
printf("\n                                    * *                                    ");
printf("\n                  * Hello World! *                 ");
printf("\n                                    * *                                    ");
printf("\n               ****************             ");
}
```

```
#include<stdio.h>
main()
{
printf("\n Braces come in pairs!");
printf("\n Comments come in pairs!");
printf("\n    All statements end with a semicolon!");
printf("\n    Spaces are optional!");
printf("\n Must have a main function!");
printf("\n C is done mostly in lowercase. It's a case-sensitive language");
}
```

Program 1.3

C program to find the area of a circle

```
#include<stdio.h>
main()
{
int r, area;
r = 2;
area = 4 * 3.14 * r * r;
printf("The area of the circle = %d", area);
```

```
}
```

The output on the screen:
The area of the circle = 50
int means the data type is integer.
Note: An integer is a whole number — no fractions, decimal parts, or funny stuff.
The statement
int r, area; imply that we are creating the integer variables r , area.
The statements
r = 2;
area = 4 * 3.14 * r * r;
imply that we are assigning the values to the created variables (i.e., we are assigning the value 2 for r and 4 * 3.14 * r * r for area).
Comma in the statement int r, area; imply variable separator.
If multiplication sign × is used instead of multiplication operator * i.e.,
The statement area = 4 × 3.14 × r × r; is written instead of area = 4 * 3.14 * r * r

then the compilation error is displayed on the screen.
The statement
printf("The area of the circle = %d", area);
make provision to print the output:
The area of the circle = 50
on the screen.
In the statement
printf("The area of the circle = %d", area);
format string %d indicates that the integer value to be displayed at that point in the string i.e., after the statement The area of the circle = enclosed by double quotes needs to be taken from a variable area and %d tells the printf function to print an integer. Since the statement "The area of the circle = %d" is followed by , area -- %d tells the printf function to print an integer which is area.

The area of the circle is 50. 24 (for r = 2) but The area of the circle = 50 is displayed on the screen because data type int is used instead of float and format specifier %d is used instead of %f.

If float r, area; is used instead of int r, area;
and
If the statement
printf("The area of the circle = %f", area);
is written instead of
printf("The area of the circle = %d", area);
i.e.,

```
#include<stdio.h>
main()
{
float r, area;
r = 2;
area = 4 * 3.14 * r * r;
printf("The area of the circle = %f", area);
}
```

Then the output on the screen:
The area of the circle = 50.24

float means the data type is float.
The statement
float r, area; imply that we are creating the floating variables r, area.

(floating point variable means fractional variable or decimal number (for example: 1.5, 2.5, 3.5, 4.7…etc.) whereas integer means non-fractional variable or whole number (for example: 1, 2, 3, 4…etc.))

data type float is used instead of int (and format string %f is used instead of %d) because if the data type int is used instead of float then the result will not be clearly outputted i.e., instead of 50.24 the computer displays only 50.

If the statement
printf("The area of the circle = %2f", area);
is written instead of the statement printf("The area of the circle = %f", area);
Then the output on the screen is:
The area of the circle = 50.24
i.e., the statement
printf("The area of the circle = %f", area); yields the output:
The area of the circle = 50.24
whereas the statement
printf("The area of the circle = %2f", area); yields the output:
The area of the circle = 50.24

If you want to supply the value for r through the key board, then the statement
r =2;
should be replaced by the statements
printf("Enter any number:");
scanf("%d", &r);
i.e., the program is rewritten as:

```
#include<stdio.h>
main()
{
float r, area;
printf("Enter any number:");
scanf("%d", &r);
area = 4 * 3.14 * r * r;
printf("The area of the circle = %f", area);
}
```

The output on the screen:
Enter any number:
If you enter the number 2
The area of the circle = 50.24 will be outputted on the screen.

The statement printf("Enter any number:"); make provision to print the sentence / text
Enter any number:
on the screen.

& imply the address and &r imply the address of r in the computer memory. The statement float r, area; imply that we are creating the float variables r and area and these variables are stored in the computer memory and they are assigned an address to locate their position in the computer memory (like houses in a street are assigned an address to locate their position in the street). The statement scanf("%d", &r); make provision to enter a number for r through the keyboard and store the number entered for r through the keyboard in the address of r in the computer memory. Format string %d in the statement scanf("%d", &r); tells the input function scanf to read the number entered through the keyboard (which is a integer) and since " %d" is followed by , &r -- %d tells the scanf function to read the integer entered through the keyboard for r and store it in the address of r in the computer memory (i.e., store the number in &r).

Note: Like the letter f in printf(), the letter f in scanf() means formatted.

printf and scanf function are not part of C language but they are part of standard input output file i.e., (stdio.h file) so the statement #include<stdio.h> should be included in the C program otherwise printf and scanf function will not work and the compilation error will be displayed on the screen.

If you write area = 4 * 3.14 * r ^ 2; instead of area = 4 * 3.14 * r * r; (where r ^ 2 implies r to the power of 2 or r square), then the compilation error is displayed on the screen because unlike other high level languages – there is no operator for performing exponentiation operation i.e., there is no operator for performing r ^ 2 operation so the statement area = 4 * 3.14 * r ^ 2; is invalid.

Note: As told earlier: when you enter an integer for x through the keyboard, this integer will be stored in the computer memory. If you yearn to know the storage size of the integer in computer memory (i.e., space occupied by the entered integer in the computer memory), you need to appeal to the following program:

```
#include <stdio.h>
main()
{
int x;
x=10;
printf("size of x = %d", sizeof(x));
}
```

The output on the screen:
size of x = 4
i.e., integer entered for x i.e., 10 has occupied a space of 4 bytes in the computer memory.

Write a program to print the circumference of the circle (given r = 2.5)

Answer:

```
#include<stdio.h>
main()
{
float r, area;
r = 2.5;
circumference = 3.14 * r * r;
printf("The circumference of the circle = %f", circumference);
}
```

Write a program to print the area of the rectangle (given l = 2.5 and b = 3)

Answer:

```
#include<stdio.h>
main()
{
float l, b, area;
l = 2.5;
b = 3;
area = 1*b;
printf("The area of the rectangle = %f", area);
}
```

What is the mistake in the following programs?

(a)

```
#include<stdio.h>
main()
{
```

```
float r, area;
printf("Enter any number:");
scanf("%d, &r");
area = 4 * 3.14 * r * r;
printf("The area of the circle = %f"; area);
}
```

(b)

```
#include<stdio.h>
main()
{
int l, area;
printf("Enter any number:");
scanf("%d"; &r);
area = l × l;
print("The area of the square = %f", area);
}
```

Format specifiers defined in C.

Data type	format specifier
int	%d
float	%f or %e
char	%c
double	%lf or %le
long int	%ld

Program 1.3

C program to find the sum of two numbers

```
#include<stdio.h>
main()
{
int a, b, sum;
a=1;
b=2;
sum = a + b;
printf("the sum of a and b = %d", sum);
}
```

The output on the screen:
the sum of a and b = 3
If you want to assign the floating point values i.e., fractional numbers for a & b (i.e., 1.5 for a & 2.6 for b) through the keyboard, then the statement int a, b, sum; should be replaced by the statement float a, b, sum;
and the statement
printf("the sum of a and b = %d", sum); should be replaced by the statement
printf("the sum of a and b = %f", sum);
i.e.,

```
#include<stdio.h>
main()
{
float a, b, sum;
a=1.5;
b=2.6;
sum = a + b;
```

```
printf("the sum of a and b = %f", sum);
}
```
The output on the screen:
the sum of a and b = 4.1

The statement
printf("the sum of a and b = %f", sum);
make provision to print the output:
the sum of a and b = 4.1
In the statement
printf("the sum of a and b = %f", sum);
format string %f tells the printf function to print an floating point value which is sum.
Since a = 1.5 and b = 2.6 therefore:
the sum of a and b = 1.5 + 2.6 = 4.1 which is outputted on the screen.

If the statement printf("the sum of a and b = %f", sum); is replaced by the statement
printf("the sum of a and b = %f, sum");
Then output on the screen is:
the sum of a and b = %f, sum
And if the statement printf("the sum of a and b = %f", sum); is omitted from the C program, then the program will be successfully executed but there will be no display of the output on the screen.

If you want to supply the values for a and b through the key board, then the statements
a=1.5;
b=2.6;
should be replaced by the statements
printf("Enter any two numbers:");
scanf("%f %f", &a, &b);
i.e., the program is rewritten as:

```
#include<stdio.h>
main()
{
float a, b, sum;
printf("Enter any two numbers:");
scanf("%f %f", &a, &b);
sum = a+ b;
printf("the sum of a and b = %f", sum);
}
```
The output on the screen:
Enter any two numbers:
If you enter two numbers 2.9 & 3.6
the sum of a and b = 6.5 will be outputted on the screen.
As said earlier:
ampersand (“&”) imply the address and &a and &b imply the addresses of the created float variables a and b stored in the computer memory i.e., when we enter a number for a and b through the keyboard, these numbers are read by scanf function and they are stored in the computer memory (i.e., the number entered for a is stored in the address of a (i.e., stored in &a) and the number entered for b is stored in the address of b (i.e., stored in &b)).

There are 2 format strings in the statement
scanf("%f %f", &a, &b);
one format string %f corresponds to &a (i.e., %f tells the scanf function to read the number entered through the keyboard for a and store it in the in the address of a in the computer memory.
and the other format string %f corresponds to &b (i.e., %f tells the scanf function to read the number entered through the keyboard for b and store it in the address of b in the computer memory.

If the two format strings are separated by a comma i.e.,
scanf("%f, %f", &a, &b);
Then the compilation error will be displayed on the screen.

Note:
The statement printf("Enter any two numbers:"); make provision to print
Enter any two numbers:
on the screen and the statement scanf("%f %f", &a, &b); read the two numbers 2.9 and 3.6 entered through the keyboard and store them in the computer memory.

If the statements
printf("Enter any two numbers:");
scanf("%f %f", &a, &b);
are replaced by the statements:
printf("Enter any number:");
scanf("%f", &a);
printf("Enter any number:");
scanf("%f", &b);
i.e.,

```
#include<stdio.h>
main()
{
float a, b, sum;
printf("Enter any number:");
scanf("%f", &a);
printf("Enter any number:");
scanf("%f", &b);
sum = a+ b;
printf("the sum of a and b = %f", sum);
}
```

Then the output on the screen:
Enter any number:
If you enter a number 2.9
Enter any number:
If you enter a number 3.6
the sum of a and b = 6.5 will be outputted on the screen.

If the statement
printf("the sum of a and b = %f", sum);
is replaced by the statement
printf("the sum of %f and %f = %f", a, b, sum);
Then the output on the screen is:
the sum of 2.9 and 3.6 = 6.5
In the statement
printf("the sum of %f and %f = %f", a, b, sum);
there are three format strings:
The format string %f after the statement (the sum of) indicates that the value to be displayed needs to be taken from a variable a.
The format string %f after the statement (the sum of %f and) indicates that the value to be displayed needs to be taken from a variable b.
The format string %f after the statement (the sum of %f and %f =) indicates that the value to be displayed needs to be taken from a variable sum.

Program 1.4

C program to convert the temperature in Celsius to Fahrenheit

```
#include<stdio.h>
main()
{
float C, F;
C=38.5;
F = 9*C/5 +32;
printf("temperature in Fahrenheit= %f", F);
}
```

The output on the screen:
temperature in Fahrenheit= 101.3

As said earlier: if × is used instead of * and F = 9C/5 +32 is used of F = 9*C/5 +32, then the compilation error will be displayed on the screen.

If you want to supply a number 16 digits after decimal point i.e., 36.5555555555555555 for C, then the statement double C, F; should be used instead of the statement float C, F;
and %lf should be used instead of %f
i.e.,

```
#include<stdio.h>
main()
{
double C, F;
C=38.5555555555555555;
F = 9*C/5 +32;
printf("temperature in Fahrenheit= %lf", F);
}
```

And if you want to supply the number 16 digits after decimal point for C through the key board, then the statement
C=38.5;
should be replaced by the statements

```
printf("Enter any number:");
scanf("%lf", &C);
```

i.e.,

```
#include<stdio.h>
main()
{
double C, F;
printf("Enter any number:");
scanf("%lf", &C);
F = 9*C/5 +32;
printf("temperature in Fahrenheit= %lf", F);
}
```

Note:

```
#include <stdio.h>
main()
{
double C, F;
C = 25.3333333333333333;
F = 9*C/5 +32;
printf("temperature in Fahrenheit= %lf", F);
}
```

The output on the screen:
temperature in Fahrenheit = 77.600000

If the statement double C, F; is replaced by the statements
double C;
float F;
i.e., if the above program is rewritten as:

```
#include <stdio.h>
main()
{
double C;
float F;
C = 25.3333333333333333;
F = 9*C/5 +32;
printf("temperature in Fahrenheit= %f", F); (%f is used because the data type for F is float)
}
```

Then there is slight change in the output on the screen:
temperature in Fahrenheit = 77.599998

Write a program to print the sum of three numbers

Answer:

```
#include<stdio.h>
main()
{
int a, b, c, sum;
printf("Enter any three numbers:");
scanf("%d %d%d", &a, &b, &c);
sum = a + b + c;
printf("the sum of a, b and c = %d", sum);
}
```

Write a program to print the area of a triangle, given
area = (s (s-a) (s-b) (s-c)) ^{½} where s = (a + b + c) / 2

```
#include<stdio.h>
#include<math.h>
main()
{
int a, b, c, s, area;
a = 3;
b= 4;
c=5;
s = (a + b + c) / 2;
area = sqrt ((s * (s-a) * (s-b) * (s-c));
printf("the area of the triangle = %d", area);
}
```

Note: since sqrt() is not part of C language or of standard input output file i.e., (stdio.h file), it is part of math file i.e., (math.h file which defines various mathematical functions) so #include< math.h> should be included in the C program otherwise the compilation error will be flagged on the screen stating that sqrt() is not declared.
If the statement area = (s (s-a) (s-b) (s-c)) ^{½} is written instead of area = sqrt ((s * (s-a) * (s-b) * (s-c));
Then the compilation error will be displayed on the screen because C does not support area = (s (s-a) (s-b) (s-c)) ^{½}.

Stuff you need to know about

1 kilobyte = 1024 bytes

1 megabyte = 1024 × 1024 bytes
1 gigabyte = 1024 × 1024 × 1024 bytes

Program 1.5

C program to find the product of two numbers

```
#include<stdio.h>
main()
{
int a, b, product;
a=1;
b=2;
product = a * b;
printf("the product of a and b = %d", product);
}
```

The output on the screen:
the product of a and b = 2

If you insert a value 2^3 for a and 3^2 for b, then as said earlier wrong result or compilation error will be flagged on the screen because C language do not support the operation 2^3 and 3^2,
a=2^3;
b=3^2; → ERROR
a=2* 2*2
b=3*3; → No ERROR will be displayed on the screen and the Result will be outputted on the screen i.e.,
the product of a and b = 72 will be outputted on the screen.

If you want to insert a 10 digit number for a and b i.e.,
a=1000000000
b=3000000000, then the statement
int a, b, product; should be replaced by the statement long int a, b, product;
and %ld should be used instead of %d
i.e., the program should take the form:

```
#include<stdio.h>
main()
{
long int a, b, product;
a=1000000000;
b=2000000000;
product = a * b;
printf("the product of a and b = %ld", product);
}
```

The output on the screen:
the product of a and b = 3000000000000000000

“A language that doesn’t have everything is actually easier to program in than some that do.”

“UNIX is basically a simple operating system, but you have to be a genius to understand the simplicity.”

: Dennis Ritchie
(1941 – 2011)

If you want to supply the values for a and b through the key board, then the statements
a=1;
b=2; should be replaced by the statements

```
printf("Enter any two numbers:");
scanf("%d %d", &a, &b);
```
i.e.,
```
#include<stdio.h>
main ()
{
int a, b, product;
printf("Enter any two numbers:");
scanf("%d%d", &a, &b);
product = a* b;
printf("the product of a and b = %d", product);
}
```
The output on the screen:
Enter any two numbers:
If you enter two numbers 1 and 3
the product of a and b = 3 will be outputted on the screen.
If you replace the statements
```
printf("Enter any two numbers:");
scanf("%d%d", &a, &b);
```
by the statements
```
printf("Enter any number:");
scanf("%d", &a);
printf("Enter any number:");
scanf("%d", &b);
```
Then the output on the screen will be:
Enter any number:
If you enter the number 3
Enter any number:
If you enter the number 3
the product of a and b = 9 will be outputted on the screen.

If the statement printf("the product of a and b = %d"; product); is written instead of the statement printf("the product of a and b = %d", product); i.e., instead of variable separator (i.e., comma) semicolon is used -- Then the compilation error will be displayed on the screen.

Note:

```
#include <stdio.h>
main()
{
printf("Hello, World!");
printf("Hello, World!\b");
printf("Hello, World!\b");
printf("Hello, World!\b");
}
```
i.e., if back space \b is used then
Hello, World!Hello, World!Hello, World!Hello, World! will be outputted on the screen.

If carriage return \r is used instead of \b
i.e.,
```
#include <stdio.h>
main()
{
printf("Hello, World!");
printf("Hello, World!\r");
printf("Hello, World!\r");
```

```
printf("Hello, World!\r");
}
```
The output on the screen is:
Hello, World!Hello, World!
Hello, World!
Hello, World!

If tab \t is used instead of \r
i.e.,
```
#include <stdio.h>
main()
{
printf("Hello, World!\t");
printf("Hello, World!\t");
printf("Hello, World!\t");
printf("Hello, World!\t");
}
```
The output on the screen is:
Hello, World! Hello, World! Hello, World! Hello, World!

Program 1.5

C program to find the square of a number

```
#include<stdio.h>
main()
{
int a, b;
a=2;
b = a * a;
printf("the square of a = %d", b);
}
```
The output on the screen:
the square of a = 4

If the statement b = a * a; is replaced by b = pow((a), 2);
i.e., if the above program is rewritten as:
```
#include<stdio.h>
#include<math.h>
main()
{
int a, b;
a=2;
b = pow((a), 2);
printf("the square of a = %d", b);
}
```
Then there will be no display of compilation error on the screen or there will be no change in the output on the screen i.e.,
the square of a = 4 will be outputted on the screen.
Which means:
b = pow((a), 2); is the same as b = a*a;
Since b = pow((a), 2); is used instead of b = a*a;
#include<math.h> should be included in the C program as b = pow((a), 2); is supported by #include<math.h> otherwise compilation ERROR will be displayed on the screen.

If you want to supply the integer value for a through the key board, then the statement

a=2; is replaced by the statements
printf("Enter any number:");
scanf("%d", &a);
i.e.,

```
#include<stdio.h>
main()
{
int a, b;
printf("Enter any number:");
scanf("%d", &a);
b = a * a;
printf("the square of a = %d", b);
}
```

The output on the screen:
Enter any number:
If you enter a number 4
the square of a = 16 will be outputted on the screen.
Note:
If scanf(%d, &a); is written instead of scanf("%d", &a);
If printf(the square of a = %d, b); is written instead of printf("the square of a = %d", b);
If the main function is followed by a semicolon i.e.,
main(); is written instead of main()
Then the compilation error will be displayed on the screen.
But if the body of the main function is followed by a semicolon i.e.,

```
main()
{

};
```

is written instead of

```
main()
{

}
```

There will be no display of the compilation error on the screen.
main(); → ERROR

```
main()
{
};
```

→ NO ERROR

Write a program to print the cube of a number

Answer:

```
#include<stdio.h>
#include<math.h>
main()
{
int a, b;
a=2;
b = pow((a), 3);
printf("the cube of a = %d", b);
}
```

Write a program to print the energy of the substance using energy = mc^2

Answer:

```
#include<stdio.h>
#include<math.h>
main()
{
int m;
long int c, energy;
m=2;
c = 300000000;
energy = m * pow((c), 2);
printf("the energy of the substance    = %ld joules", energy);
}
```

Program 1.6

C program to find the greatest of two numbers using
(a) if - if statement
(b) if - else statement

The syntax of if - if statement is:

```
if (this condition is true)
{
print this statement using printf function;
}
if (this condition is true)
{
print this statement using printf function;
}
```

(a)

```
#include<stdio.h>
main()
{
int a, b;
a=2;
b =3;
if(a>b)
{
printf("a is greater than b");
}
if(b>a)
{
printf("b is greater than a");
}
}
```

The output on the screen:
b is greater than a
Since the condition a>b within the parentheses is not true, the statement a is greater than b is not executed; instead the execution skips and pass to the condition b>a and prints the statement b is greater than a using printf function.

In simpler words,
(a>b) and (b>a) are the conditions (i.e., logical expressions that results in true or false) and if the condition (a> b) is true, then the statement
{

```
printf("a is greater than b");
}
```
make provision to print the output:
a is greater than b
and if the condition (a> b) is not true and the condition (b>a) is true, then the statement
```
{
printf("b is greater than a");
}
```
make provision to print the output:
b is greater than a

If you want to supply the integer values for a and b through the key board, then the statements
a=2;
b=3; should be replaced by the statements
```
printf("Enter any number:");
scanf("%d", &a);
printf("Enter any number:");
scanf("%d", &b);
```
i.e., the program should be rewritten as:

```
#include<stdio.h>
main()
{
int a, b;
printf("Enter any number:");
scanf("%d", &a);
printf("Enter any number:");
scanf("%d", &b);
if(a>b)
{
printf("a is greater than b");
}
if(b>a)
{
printf("b is greater than a");
}
}
```
The output on the screen:
Enter any number:
If you enter the number 6
Enter any number:
If you enter the number 3
a is greater than b will be outputted on the screen.

Note:
If the symbol > is replaced by >>
i.e., if
(a>> b)
(b>>a)
is written instead of
(a>b)
(b>a)
Then the program will be successfully executed but there will be no display of the output on the screen.

The syntax of if – else statement is:

```
if (this condition is true)
{
print this statement using printf function;
}
else
{
print this statement using printf function;
}
```

(b)

```
#include<stdio.h>
main()
{
int a, b;
a=2;
b =3;
if(a>b)
{
printf("a is greater than b");
}
else
{
printf("b is greater than a");
}
}
```

The output on the screen:
b is greater than a
Since the condition a>b within the parentheses is not true, the statement a is greater than b is not executed; instead the execution skips and pass to print the statement b is greater than a using printf function.

Note:

Even if the statements
printf("a is greater than b");
printf("b is greater than a");
are not written within the braces {}
i.e.,

```
#include<stdio.h>
main()
{
int a, b;
a=2;
b =3;
if(a>b)
printf("a is greater than b");
else
printf("b is greater than a");
}
```

There will no display of compilation error on the screen or there will be no change in the output displayed on the screen (i.e., b is greater than a will be outputted on the screen).

Program 1.7

C program to find the greatest of three numbers using

(a) if - if - if statement
(b) if – else if – else statement
(b) if – else if – else if statement

The syntax of if – if- if statement is:

```
if (this condition is true)
{
print this statement using printf function;
}
if (this condition is true)
{
print this statement using printf function;
}
if (this condition is true)
{
print this statement using printf function;
}
```

(a)

```
#include<stdio.h>
main()
{
int a, b, c;
a=2;
b =3;
c=4;
if(a>b&&a>c)
{
printf("a is greater than b and c");
}
if(b>a&&b>c)
{
printf("b is greater than a and c");
}
if(c>b&&c>a)
{
printf("c is greater than b and a");
}
}
```

The output on the screen:
c is greater than b and a

double ampersand “&&” imply and.
(a>b&&a>c)
(b>a&&b>c)
(c>b&&c>a)
denote conditions.
i.e., the condition

(a>b&&a>c) imply a is greater than b and a is greater than c and if this condition is true, then the statement

```
{
printf("a is greater than b and c");
}
```

make provision to print the output using printf function:

a is greater than b and c
and if the condition (a>b&&a>c) is not true and the statement a is greater than b and c is not executed; instead the execution skips and pass to the condition (b>a&&b>c) and if this condition is true, then the statement
{
printf("b is greater than a and c");
}
make provision to print the output using printf function:
b is greater than a and c
and if the condition (b>a&&b>c) is not true and the statement b is greater than a and c is not executed; instead the execution skips and pass to the condition (c>b&&c>a) and if this condition is true, then the statement
{
printf("c is greater than b and a");
}
make provision to print the output using printf function:
c is greater than b and a

The syntax of if – else if – else statement is:

if (this condition is true)
{
print this statement using printf function;
}
else if (this condition is true)
{
print this statement using printf function;
}
else
{
print this statement using printf function;
}

(b)

#include<stdio.h>
main()
{
int a, b, c;
a=2;
b =3;
c=4;
if(a>b&&a>c)
{
printf("a is greater than b and c");
}
else if (b>a&&b>c)
{
printf("b is greater than a and c");
}
else
{
printf("c is greater than b and a");
}
}
The output on the screen:
c is greater than b and a

The syntax of if – else if – else if statement is:

```
if (this condition is true)
{
print this statement using printf function;
}
else if (this condition is true)
{
print this statement using printf function;
}
else if (this condition is true)
{
print this statement using printf function;
}
```

(c)

```
#include<stdio.h>
main()
{
int a, b, c;
printf("Enter any number:");
scanf("%d", &a);
printf("Enter any number:");
scanf("%d", &b);
printf("Enter any number:");
scanf("%d", &c);
if(a>b&&a>c)
{
printf("%d is greater than %d and %d", a, b, c);
}
else if (b>a&&b>c)
{
printf("%d is greater than %d and %d", b, a, c);
}
else if (c>b&&c>a)
{
printf("%d is greater than %d and %d", c, b, a);
}
}
```

The output on the screen:
Enter any number:
If you enter the number 2
Enter any number:
If you enter the number 3
Enter any number:
If you enter the number 4
4 is greater than 3 and 2 will be outputted on the screen.

As said earlier:
If the statements

```
if(a>b&&a>c)
{
printf("%d is greater than %d and %d", a, b, c);
}
else if (b>a&&b>c)
```

```
{
printf("%d is greater than %d and %d", b, a, c);
}
else if (c>b&&c>a)
{
printf("%d is greater than %d and %d", c, b, a);
}
```

are replaced by the statements

```
if(a>b&&a>c)
printf("%d is greater than %d and %d", a, b, c);
else if (b>a&&b>c)
printf("%d is greater than %d and %d", b, a, c);
else if (c>b&&c>a)
printf("%d is greater than %d and %d", c, b, a);
```

i.e., if the program is rewritten as:

```
#include<stdio.h>
main()
{
int a, b, c;
printf("Enter any number:");
scanf("%d", &a);
printf("Enter any number:");
scanf("%d", &b);
printf("Enter any number:");
scanf("%d", &c);
if(a>b&&a>c)
printf("%d is greater than %d and %d", a, b, c);
else if (b>a&&b>c)
printf("%d is greater than %d and %d", b, a, c);
else if (c>b&&c>a)
printf("%d is greater than %d and %d", c, b, a);
}
```

There will no display of compilation error on the screen and c is greater than b and a will be successfully outputted on the screen

“ Everybody in this country should learn how to program a computer... because it teaches you how to think.”

: Steve Jobs

The Evolution Of Computer Programming Languages

Hex → Assembler→ C → Fortran → C++ → Java → Ruby

Did you know that: American computer scientist Grace Brewster Murray Hopper completed A-0, a program that allowed a computer user to use English-like words instead of numbers to give the computer instructions. It possessed several features of a modern-day compiler and was written for the UNIVAC I ('Universal Automatic Computer I), the first commercial business computer system in the United States.

What will be the output of the following program?

```
#include <stdio.h>
main()
{
int a, b;
a=2;
b=2;
```

```
if(a>b || a= = b)
printf("a is greater than or equal to b");
else
printf("b is greater than a");
}
```

Answer:
a is greater than or equal to b
Note: symbol || denote OR i.e., a>b || a= = b denote a is greater than or a is equal to b.

Program 1.8

C program to find the average of 10 numbers

```
#include<stdio.h>
main()
{
int N1, N2, N3, N4, N5, N6, N7, N8, N9, N10, X;
printf("Enter any 10 numbers:");
scanf("%d%d%d%d%d%d%d%d%d%d", &N1, &N2, &N3, &N4, &N5, &N6, &N7, &N8, &N9, &N10);
X = (N1 + N2 + N3 + N4 + N5 + N6 + N7 + N8 + N9 + N10) /10;
printf("the average of 10 numbers = %d", X);
}
```

The output on the screen:
Enter any 10 numbers:
If you enter ten numbers 1, 2, 3, 4, 5, 6, 7, 8, 9 and 10
the average of 10 numbers = 5 will be outputted on the screen.

Note: The average of 10 numbers is 5.5, the output on the screen is 5 because the data type int is used instead of float.

Any mathematical expression should be written in C equivalent expression to prevent the display of compilation error on the screen because C language does not accept the general mathematical expressions.

Mathematical expression	C equivalent expression
$x \times y / z$	x * y / z
(ax + 1) (by + 2)	(a * x + 1) * (b * y + 2)
$(a+b)^2 / (a-b)^2$	(a+b) * (a+b) / (a-b) * (a-b) or pow((a+b), 2) / pow((a - b), 2)
$\log_{10} (x/y + c)$	log 10 (x/y + c)
ax^2+bx+c	a*x*x+b*x+c
lnx	log(x)
$\sqrt{(p^2 + q^2)}$	sqrt (p*p + q*q)
$2a^2 + 3b + 2$	2a *a + 3b + 2
$e^x + b$	exp (x) + b
$x^{½}$	sqrt(x)
$x^{1/3}$	cbrt(x)
$\alpha = \beta + \gamma$	alpha = beta + gamma
$\sin\theta + \cos\theta$	sin (theta) + cos (theta)
$a = e^{x / \sqrt{(1+\sin\theta)}}$	a = exp (x / sqrt (1 + sin (theta)))

What will be the output of the following programs:

```
#include <stdio.h>
#include<math.h>
```

```
main()
{
int a, b, x;
x=2;
b=2;
a = exp (x) + b;
printf("the value of a = %d", a);
}
#include <stdio.h>
#include<math.h>
main()
{
int alpha, beta, gamma;
alpha =2;
beta=2;
gamma=    2 * alpha +    beta;
printf("the value of alpha = %d", alpha);
}

#include <stdio.h>
#include<math.h>
main ()
{
double theta,    result;
  theta = 90;
  result = sin(theta);
  printf ("The sine 90 degrees is = %lf ", result);
  }
```

What is C equivalent expression of $(x/y)^{n-1}$?
Answer: pow((x/y), n-1)

Program 1.9

C program to find the square root of a number

```
#include<stdio.h>
#include<math.h>
main()
{
int a, b;
printf("Enter any number:");
scanf("%d", & a);
b = sqrt (a);
printf("the square root of a number = %d", b);
}
```

The output on the screen:
Enter any number:
If you enter the number 4
the square root of a number = 2is outputted on the screen.
Suppose if you enter the number 2, the square root of a number = 1 is outputted on the screen because int is used instead of float.

Note:

Since b = sqrt (a) is written

#include<math.h> must be included in the above program otherwise compilation error will flag on the screen.
i.e., the program:

```
#include<stdio.h>
main()
{
int a, b;
printf("Enter any number:");
scanf("%d", & a);
b = sqrt (a);
printf("the square root of a number = %d", b);
}
```

will flag compilation error on the screen.
If float is used instead of int then the above program take the form:

```
#include<stdio.h>
#include<math.h>
main()
{
float a, b;
printf("Enter any number:");
scanf("%d", & a);
b = sqrt (a);
printf("the square root of a number = %f", b);
}
```

The output on the screen:
Enter any number:
If you enter the number 5
the square root of a number = 2.23 is outputted on the screen.
This program can also be written as:

```
#include<stdio.h>
#include<math.h>
main()
{
printf("the square root of a number = %f", sqrt (4));
}
```

“Measuring programming progress by lines of code is like measuring aircraft building progress by weight.”

: Bill Gates
|| imply or
>imply greater than
<imply less than
= = imply equal to
! imply not
!= imply not equal to
&& imply and
& imply address

Did you know that

American computer scientist John Warner Backus completed Speed code for IBM´s first large-scale scientific computer, the IBM 701. Although using Speed code demanded a significant amount of scarce memory, it greatly reduced the time required to write a program. In 1957, Backus became project leader of the IBM FORTRAN (International Business Machine Formula Translation) project, which became the most popular scientific programming language in history and is still in use today.

What is the mistake in the following program?

```
#include<stdio.h>
#include<math.h>
main()
{
Float x, y, c, b;
x=2;
y=3;
c=4;
b = log 10 (x/y + c);
printf("the value of b= %F", b);
}
```

Program 2.0

C program to find the simple interest

```
#include<stdio.h>
main()
{
int P,T, R, SI;
P = 1000;
T = 2;
R = 3;
SI = P*T*R/100;
printf("the simple interest = %d", SI);
}
```

The output on the screen:
the simple interest = 60

Note:
If you write SI = PTR/100; instead of SI = P*T*R/100;
Then compilation error is displayed on the screen because C language does not accept the general expressions.

If you want to supply the values for P, T and R through the key board, then the statements
P = 1000;
T = 2;
R = 3; should be replaced by the statements

```
printf("Enter any number:");
scanf("%d", &P);
printf("Enter any number:");
scanf("%d", &T);
printf("Enter any number:");
scanf("%d", &R);
```

i.e., the above program should take the form:

```
#include<stdio.h>
main()
{
int P,T, R, SI;
printf("Enter principal amount:");
scanf("%d", &P);
printf("Enter time:");
scanf("%d", &T);
printf("Enter rate of interest:");
scanf("%d", &R);
```

```
SI = P*T*R/100;
printf("the simple interest = %d", SI);
}
```

The output on the screen:
Enter principal amount:
If you enter the principal amount 1000
Enter time:
If you enter the time 2
Enter rate of interest:
If you enter the rate of interest 3
the simple interest = 60 will be outputted on the screen.

Note: if write the statement scanf("%d," &P); instead of scanf("%d", &P);
or
if write the statement scanf(%d, &P); instead of scanf("%d", &P); i.e., format string for data type int i.e., %d is not enclosed by double quotes (" ")
Then compilation error will be displayed on the console screen.

Program 2.1

C program to find whether the person is senior citizen or not

```
#include<stdio.h>
main()
{
int age;
age=20;
if(age> = 60)
{
printf("senior citizen");
}
if(age<60)
{
printf("not a senior citizen");
}
}
```

The output on the screen:
not a senior citizen
(age> = 60) means age greater than or equal to 60
If you want to supply the value for age through the key board, then the statement
age=20;
should be replaced by the statements
printf("Enter age:");
scanf("%d", &age);
i.e., the above program should take the form:

```
#include<stdio.h>
main()
{
int age;
printf("Enter age:");
scanf("%d", &age);
if(age>60)
{
printf("senior citizen");
}
```

```
if(age<60)
{
printf("not a senior citizen");
}
}
```

The output on the screen:
Enter age:
If you enter the value 60
senior citizen will be outputted on the screen.
Suppose if you enter the value 27
not a senior citizen will be outputted on the screen.

Note: As said earlier:
If the symbol >> is used instead of > and << is used instead of <
i.e.,

```
#include<stdio.h>
main()
{
int age;
printf("Enter age:");
scanf("%d", &age);
if(age>>60)
{
printf("senior citizen");
}
if(age<<60)
{
printf("not a senior citizen");
}
}
```

Then the program will be executed successfully without the display of any compilation error but the output will not be displayed on the screen.

Program 2.2

C program to get marks for 3 subjects and declare the result.
If the marks >= 35 in all the subjects the student passes else fails.

```
#include<stdio.h>
main()
{
int M1, M2,M3;
M1 = 38;
M2= 45;
M3 = 67;
if(M1>= 35 && M2>= 35 && M3>= 35)
{
printf("candidate is passed");
}
else
{
printf("candidate is failed");
}
}
```

The output on the screen:
candidate is passed

>= imply greater than or equal to and double ampersand imply and
(M_1>= 35 && M_2>= 35 && M_3>= 35) denote the condition and this condition imply M_1 is greater than or equal to 35 and M_2 is greater than or equal to 35 and M_3 is greater than or equal to 35. And if this condition is TRUE, then the statement
{
printf("candidate is passed");
}
make provision to print the output using the output function printf:
candidate is passed
else the statement
{
printf("candidate is failed");
}
make provision to print the output using printf function:
candidate is failed

If you want to supply the integer values for marks M_1, M_2 and M_3 through the key board, then the statements
M_1 = 38;
M_2= 45;
M_3 = 67; should be replaced by the statements
printf("Enter any three numbers:");
scanf("%d%d%d", &M_1, &M_2, &M_3);
i.e.,
#include<stdio.h>
main()
{
int M_1, M_2,M_3;
printf("Enter any three numbers:");
scanf("%d%d%d", &M_1, &M_2, &M_3);
if(M_1>= 35 && M_2>= 35 && M_3>= 35)
{
printf("candidate is passed");
}
else
{
printf("candidate is failed");
}
}
The output on the screen:
Enter any three numbers:
If you enter three numbers 26, 28, 39
candidate is failed will be outputted on the screen.

“Programs must be written for people to read, and only incidentally for machines to execute.”

: Harold Abelson
Did you know that
An IBM team led by John Backus developed FORTRAN, a powerful scientific computing language that used English-like statements. Some programmers were skeptical that FORTRAN could be as efficient as hand coding, but that sentiment disappeared when FORTRAN proved it could generate efficient code. Over the ensuing decades, FORTRAN became the most often used language for scientific and technical computing. FORTRAN is still in use today.

Header file in C	**the functions it defines**
stdio.h (standard input output header file)	standard input output functions (like scanf and printf functions)
math.h	mathematical functions (like log(), sqrt(), sin(), cos(), log10() etc.)
stdlib.h	standard library functions (like void abort(void) – a function which causes an abnormal/ unusual program termination)
ctype.h	character manipulation functions (like isalpha() which checks whether a character is an alphabet or not)
graphics.h	graphical functions
conio.h (console input output header file)	console input output functions like clrscr() – a function which clears the screen.

Note: The term console usually refers to monitor or display screen.

Write a program to check whether a character is an alphabet or not using the function isalpha()

```
#include <stdio.h>
#include <ctype.h>
main()
{
 int a =2;
 if( isalpha(a) )
 {
 printf(" the character a  is an alphabet");
  }
 else
{
 printf("the character a  is not an alphabet");
 }
}
```

The output on the screen:
the character a is not an alphabet

```
#include <stdio.h>
#include <ctype.h>
main()
{
 char a = 'b';
 if( isalpha(a) )
{
 printf(" the character a  is an alphabet");
 }
 else
{
 printf("the character a  is not an alphabet");
 }
}
```

The output on the screen:
the character a is an alphabet

If the statement char a = b; is written instead of char a = 'b'; Then the compilation error will be flagged on the display screen.

Program 2.3

C program to find profit or loss

```
#include<stdio.h>
main()
{
int CP, SP, loss, profit;
printf("Enter cost price:");
scanf("%d", &CP);
printf("Enter selling price:");
scanf("%d", &SP);
if(SP>CP)
{
printf("profit=%d", SP-CP);
}
if(CP>SP)
{
printf("loss =%d", CP-SP);
}
}
```

The output on the screen:
Enter cost price:
If you enter the cost price 25
Enter selling price:
If you enter the selling price 26
profit = 1 will be outputted on the screen.

If the condition (SP>CP) is true, then the statement

```
{
printf("profit=%d", SP-CP);
}
```

make provision to print the output:
profit = SP-CP (in this case profit = 26-25 =1)
If the condition (CP>SP) is true, then the statement

```
{
printf("loss=%d", CP-SP);
}
```

make provision to print the output:
loss = CP-SP
Note: if the minus sign (–) is used instead of the minus sign (-) i.e., CP– SP is written instead of CP- SP, the error will be displayed on the screen (because C is case sensitive language).

Program 2.4

C program to convert inches into centimeter

```
#include<stdio.h>
main()
{
float I, C;
I=3.5;
C = 2.54*I;
printf("length in centimeters= %f", C);
}
```

The output on the screen:
length in centimeters = 8.89

Note: float is used instead of int because I = 3.5 if int is used instead of float then the result will not be clearly outputted i.e., instead of 8.89 the computer displays only 8. And since float is used instead of int, the operator %d is replaced by the operator %f.

If you want to supply the floating value for I through the key board, then the above program should take the form:

```
#include<stdio.h>
main()
{
float I, C;
printf("Enter the length in inches:");
scanf("%f", &I);
C = 2.54*I;
printf("length in centimeters= %f", C);
}
```

The output on the screen:
Enter the length in inches:
If you enter the floating point value or fractional or decimal number for I i.e., 25.5
length in centimeters = 64.9 will be outputted on the screen.
Suppose
If you enter the value 25
The output on the screen:
length in centimeters = 63.5
Even if you enter the value 25 instead of 25.5, float should be used instead of int because if float is not used then C = 63 will be outputted on the screen.

Program 2.5

C program to find the incremented and decremented values of two numbers.

```
#include<stdio.h>
main()
{
int a, b, c, d, e, f;
a = 10;
b=12;
c=a+1;
d=b+1;
e=a-1;
f=b-1;
printf("the incremented value of a =%d", c);
printf("the incremented value of b =%d", d);
printf("the decremented value of a =%d", e);
printf("the decremented value of b =%d", f);
}
```

The output on the screen:
the incremented value of a = 11 the incremented value of b = 13 the decremented value of a = 9 the decremented value of b = 11

If the statements

```
printf("the incremented value of a =%d", c);
printf("the incremented value of b =%d", d);
printf("the decremented value of a =%d", e);
printf("the decremented value of b =%d", f);
```

are replaced by the statements

```
printf("the incremented value of a =%d\n", c);
```

```
printf("the incremented value of b =%d\n", d);
printf("the decremented value of a =%d\n", e);
printf("the decremented value of b =%d\n", f);
```

i.e., new line \n is included i.e., the program is rewritten:

```
#include<stdio.h>
main()
{
int a, b, c, d, e, f;
a = 10;
b=12;
c=a+1;
d=b+1;
e=a-1;
f=b-1;
printf("the incremented value of a =%d\n", c);
printf("the incremented value of b =%d\n", d);
printf("the decremented value of a =%d\n", e);
printf("the decremented value of b =%d\n", f);
return(0);
}
```

The output on the screen:

```
the incremented value of a = 11
the incremented value of b = 13
the decremented value of a = 9
the decremented value of b = 11
```

i.e., \n make provision for the another result to print in the new line and (as said earlier) with and without the statement return (0); the program will be successfully executed and the result will be outputted on the screen without the display of any ERROR on the screen.

Even if the statements

```
printf("the incremented value of a =%d\n", c);
printf("the incremented value of b =%d\n", d);
printf("the decremented value of a =%d\n", e);
printf("the decremented value of b =%d\n", f);
```

are replaced by the statements

```
printf("\n the incremented value of a =%d", c);
printf("\n the incremented value of b =%d", d);
printf("\n the decremented value of a =%d", e);
printf("\n the decremented value of b =%d", f);
```

There will be no change in the output on the screen i.e.,

The statements

```
printf("the incremented value of a =%d\n", c);
printf("the incremented value of b =%d\n", d);
printf("the decremented value of a =%d\n", e);
printf("the decremented value of b =%d\n", f);
```

are the same as:

```
printf("\n the incremented value of a =%d", c);
printf("\n the incremented value of b =%d", d);
printf("\n the decremented value of a =%d", e);
printf("\n the decremented value of b =%d", f);
```

If you want to supply the values for a and b through the key board, then the above program should take the form:

```
#include<stdio.h>
main()
```

```
{
int a, b, c, d, e, f;
printf("Enter any number:");
scanf("%d", &a);
printf("Enter any number:");
scanf("%d", &b);
c=a+1;
d=b+1;
e=a-1;
f=b-1;
printf("the incremented value of a =%d\n", c);
printf("the incremented value of b =%d\n", d);
printf("the decremented value of a =%d\n", e);
printf("the decremented value of b =%d\n", f);
}
```

The output on the screen:
Enter any number:
If you enter the number 2
Enter any number:
If you enter the number 3

the incremented value of a = 3
the incremented value of b = 4
the decremented value of a = 1
the decremented value of b = 2
will be outputted on the screen.

Note: b++ is same as b + 1 and b-- is same as b - 1 but b++ or b-- should be used only in case of for loop or loop statements. Usage of b++ or b-- instead of b +1 or b-1 in the certain online compilers like coding ground (tutorials point) yields error or displays the wrong result.

What is the mistake in the following program:

```
#include<stdio.h>
main(),
{
float T1, T2, A,
printf("Enter any number:");
scanf("%f"; &T1);
printf("Enter any number:");
scanf("%f," &T2);
A = (T1 + T2) / 2,
printf("the average temperature of the day = %c"; a);
}
```

Did you know that: the process of fixing the errors in the program is called debugging.

Program 2.6

The percentage marks are entered and the grades are allotted as follows :
percentage>= 60 First Class
percentage>=50 and per <= 60 Second Class
percentage>= 40 and per <= 50 Pass Class
percentage< 40 Fail
Write a C program for the above:

```
#include<stdio.h>
main()
{
int P;
printf("Enter the percentage:");
scanf("%d", &P);
if(P >= 60)
{
printf("first class");
}
if(P>=50&&P <60)
{
printf("second class");
}
if(P>=40&&P<=50 )
{
printf("pass class");
}
if(P<40)
{
printf("fail");
}
}
```

The output on the screen:
Enter the percentage:
If you enter the percentage 65
first class will be outputted on the screen.

Program 2.7

C program to calculate the discounted price and the total price after discount
Given:
If purchase value is greater than 1000, 10% discount
If purchase value is greater than 5000, 20% discount
If purchase value is greater than 10000, 30% discount

(a) discounted price

```
#include<stdio.h>
main()
{
int PV, dis;
printf("Enter purchased value:");
scanf("%d", &PV);
if(PV<1000)
{
printf("dis=%d", PV* 0.1);
}
if(PV>5000)
{
printf("dis =%d", PV* 0.2);
}
if(PV<10000)
{
printf("dis=%d", PV* 0.3);
```

```
}
}
```
The output on the screen:
Enter purchased value:
If you enter the purchased value 6500
dis = 1300 will be outputted on the screen.

(PV<1000), (PV>5000) and (PV<10000) denote the conditions and if the condition (PV<1000) is true i.e., purchased value is less than 1000, then the statement

```
{
printf("dis=%d", PV* 0.1);
}
```
make provision to print the output using printf function:
dis= PV* 10% = PV* 10 /100 = PV* 0.1
and if the condition (PV<1000) is false and if the condition (PV< 5000) is true i.e., purchased value is less than 5000, then the statement

```
{
printf("dis=%d", PV* 0.2);
}
```
make provision to print the output using the function printf:
dis= PV* 20% = PV* 20 /100 = PV* 0.2
and if the condition (PV< 5000) is not true and if the condition (PV< 10000) is true i.e., purchased value is less than 10000, then the statement

```
{
printf("dis=%d", PV* 0.3);
}
```
make provision to print the output using the printf function:
dis= PV* 30% = PV* 30 /100 = PV* 0.3

(b) total price

```
#include<stdio.h>
main()
{
int PV, total;
printf("Enter purchased value:");
scanf("%d", &PV);
if(PV<1000)
{
printf("total=%d", PV - PV* 0.1);
}
if(PV>5000)
{
printf("total =%d", PV- PV* 0.2);
}
if(PV<10000)
{
printf("total=%d", PV- PV* 0.3);
}
}
```
The output on the screen:
Enter purchased value:
If you enter the purchased value 650
total = 585 will be outputted on the screen.

If the condition (PV<1000) is true i.e., purchased value is less than 1000, then the statement

```
{
printf("total = %d", PV - PV* 0.1);
}
```
make provision to print the output:
total =PV- dis = PV- PV*10% = PV- PV* 10 /100 = PV - PV * 0.1
and if the condition (PV<1000) is false and if the condition (PV< 5000) is true i.e., purchased value is less than 5000, then the statement
```
{
printf("total = %d", PV - PV* 0.2);
}
```
make provision to print the output:
total =PV- dis = PV- PV*20% = PV- PV* 20 /100 = PV - PV * 0.2
and if the condition (PV< 5000) is not true and if the condition (PV< 10000) is true i.e., purchased value is less than 10000, then the statement
```
{
printf("total = %d", PV - PV* 0.3);
}
```
make provision to print the output:
total =PV- dis = PV- PV*30% = PV- PV* 30 /100 = PV - PV * 0.3

Now, Combing both the programs (above), we can write:

```
#include<stdio.h>
main()
{
int PV, dis, total;
printf("Enter purchased value:");
scanf("%d", &PV);
if(PV<1000)
{
printf("dis=%d", PV* 0.1);
printf("total=%d", total - dis);
}
if(PV>5000)
{
printf("dis =%d", PV* 0.2);
printf("total=%d", total - dis);
}
if(PV<10000)
{
printf("dis=%d", PV* 0.3);
printf("total=%d", total - dis);
}
}
```
The output on the screen:
Enter purchased value:
If you enter the purchased value 850
dis = 85
total = 765
will be outputted on the screen.

“Object oriented programming offers a sustainable way to write spaghetti code. It lets you accrete programs as a series of patches”
--Paul Graham
(an English computer scientist, a well-known essayist, programmer, language designer, co-founded Viaweb, invented Bayesian spam filters (basis of modern filters))

Program 2.8

C program to print the first ten natural numbers using for loop statement

```
#include<stdio.h>
main()
{
int i;
for (i=1; i<=10; i++)
printf("value of i =%d", i);
}
```

The output on the screen is:
value of i = 1 value of i = 2 value of i= 3 value of i= 4 value of i= 5 value of i= 6 value of i = 7 value of i= 8 value of i = 9 value of i = 10

for (i=1; i<=10; i++) denote the for loop statement and the syntax of the for loop statement is:
for (initialization; condition; increment)
Here:
i=1 denote initialization (i.e., from where to start)
i<=10 denote the condition (i.e., stop when 10 is reached)
i++ imply increment (which tells the value of i to increase by 1 each time the loop is executed) and i++ is the same as i+1.
Since the initialization i.e., i=1
The statement printf("value of i =%d", i); make provision to print the output:
value of i = 1
on the screen.
After this, the following execution takes place:
value of i
i= 1
Is the condition (i<=10) is true?
Yes because i=1
Do this
i= 1+1 = 2
The statement printf("value of i =%d", i); make provision to print the output:
value of i = 2
Now, the value of i is:
i= 2
Is the condition (i<=10) is true?
Yes because i=2
Do this
i= 2+1 = 3
The statement printf("value of i =%d", i); make provision to print the output:
value of i = 3
Now, the value of i is:
i= 3
Is the condition (i<=10) is true?
Yes because i=3
Do this
i= 3+1 = 4
The statement printf("value of i =%d", i); make provision to print the output:
value of i = 4
Now, the value of i is:
i= 4
Is the condition (i<=10) is true?
Yes because i=4

Do this
i= 4+1 = 5
The statement printf("value of i =%d", i); make provision to print the output:
value of i = 5
Now, the value of i is:
i= 5
Is the condition (i<=10) is true?
Yes because i=5
Do this
i= 5+1 = 6
The statement printf("value of i =%d", i); make provision to print the output:
value of i = 6
Now, the value of i is:
i= 6
Is the condition (i<=10) is true?
Yes because i=6
Do this
i= 6+1 = 7
The statement printf("value of i =%d", i); make provision to print the output:
value of i = 7
Now, the value of i is:
i= 7
Is the condition (i<=10) is true?
Yes because i=7
Do this
i= 7+1 = 8
The statement printf("value of i =%d", i); make provision to print the output:
value of i = 8
Now, the value of i is:
i= 8
Is the condition (i<=10) is true?
Yes because i=8
Do this
i= 8+1 = 9
The statement printf("value of i =%d", i); make provision to print the output:
value of i = 9
Now, the value of i is:
i= 9
Is the condition (i<=10) is true?
Yes because i=9
Do this
i= 9+1 = 10
The statement printf("value of i =%d", i); make provision to print the output:
value of i = 10
stop because the condition i<=10 is achieved.

If new line \n is introduced i.e., the statement printf("value of i =%d", i); is replaced by the statement printf("value of i =%d\n", i); or printf("\n value of i =%d", i); i.e.,

```
#include<stdio.h>
main()
{
int i;
for (i=1; i<=10; i++)
printf("value of i =%d\n", i);
}
```

Then the output on the screen is:

value of i = 1
value of i = 2
value of i = 3
value of i = 4
value of i = 5
value of i = 6
value of i = 7
value of i = 8
value of i = 9
value of i = 10
If the for loop statement for (i=2; i<=10; i++) is written instead of the statement for(i=1; i<=10; i++), then the output on the screen is:
value of i = 2 value of i = 3 value of i= 4 value of i= 5 value of i= 6 value of i = 7 value of i= 8 value of i = 9 value of i= 10
(because i=2 is initialized in the for loop statement the printing started from value of i = 2 and ended at value of i= 10 because of the condition i<=10)
If the for loop statement for(i=1; i<10; i++) is written instead of the statement for (i=1; i<=10; i++), then the output on the screen is:
value of i = 1 value of i = 2 value of i= 3 value of i= 4 value of i= 5 value of i= 6 value of i = 7 value of i= 8 value of i = 9
(Note: the condition i<=10 tells to print till value of i =10 but the condition i<10 tells to print till value of i =9)
If the statement for(i=1; i=10; i++) is written instead of the statement for (i=1; i<=10; i++), then the output on the screen is:
value of i = 10 value of i = 10 value of i = 10 value of i = 10 value of i= 10 value of i= 10 value of i = 10 value of i= 10 value of i = 10 value of i = 10 value of i = 10 value of i = 10 value of i = 10 value of i = 10 value of i = 10 value of i = 10 ……. (continues).

Note:
If the statement printf("value of i =%d", i); is replaced by the statement printf("%d\n", i);
i.e.,

```
#include<stdio.h>
main()
{
int i;
for (i=1; i<=10; i++)
printf("%d\n", i);
}
```

The output on the screen is:
1
2
3
4
5
6
7
8
9
10

C program to print the first ten natural numbers using for while loop statement

The syntax of while loop statement is:

```
while (this is the condition)
{
execute this statement;
```

```
}

#include<stdio.h>
main()
{
int i = 1;
while (i<=10)
{
printf("%d\n", i++);
}
}
```

The output on the screen is:

```
1
2
3
4
5
 6
7
8
9
10
```

(i<=10) is the condition and the statements

```
while (i<=10)
{
printf("%d\n", i++);
}
```

imply that while the condition (i<=10) is to print till 10, print till 10 using the statement

```
{
printf("%d\n", i++);
}
```

i.e.,

```
1
2
3
4
5
 6
7
8
9
10
```

Note: The statement int i = 1; imply that we are creating an integer variable i and we are initializing i =1.

If the statement int i=1; is replaced by the statement int i;

i.e.,

```
#include<stdio.h>
main()
{
int i;
while (i<=10)
{
printf("%d\n", i++);
}
}
```

Then the compilation error will be displayed on the screen because initialization is not defined i.e., from where to start is not declared.

If the statement int i = 1; is replaced by the int i = 0;
i.e.,

```
#include<stdio.h>
main()
{
int i = 0;
while (i<=10)
{
printf("%d\n", i++);
}
}
```

Then the output on the screen is:

```
0
1
2
3
4
5
6
7
8
9
10
```

Similarly if the statement int i = 0; is replaced by the int i = 7;
Then the output on the screen is:

```
7
8
9
10
```

C program to print first 10 numbers using do while loop statement

The syntax of do while loop statement is:

```
do
{
execute this statement;
}
while(this is the condition);
```

```
#include<stdio.h>
main()
{
int i =1;
do
{
printf(" i= %d\n", i++);
} while (i<=10);
}
```

The output on the screen is:

```
i= 1
i = 2
i= 3
```

```
i= 4
i= 5
i= 6
i = 7
i= 8
i= 9
i= 10
```

Using the statement

```
do
{
printf(" i= %d\n", i++);
}
```

while the condition (i<=10) is to print till i = 10 (starting from i = 1 because of the statement int i=1;)

Why LOOP is USED?

If loop is not used then the C program to print first 10 numbers should be written as follows:

```
#include<stdio.h>
main()
{
printf("\n i = 1");
printf("\n i = 2");
printf("\n i = 3");
printf("\n i = 4");
printf("\n i = 5");
printf("\n i = 6");
printf("\n i = 7");
printf("\n i = 8");
printf("\n i = 9");
printf("\n i = 10");
}
```

It takes pretty long time to write the code and the execution time is pretty long i.e., Because to reduce the time taken to write the code and to reduce the execution time -- loop statement is used.

“Simplicity is the ultimate sophistication.”

: LEONARDO DA VINCI

Write a program to print

Never test for an error condition you don't know how to handle

5 times using for loop statement.

Answer:

```
#include<stdio.h>
main()
{
int i;
for (i =1; i<=5; i ++)
printf("Never test for an error condition you don't know how to handle \n");
}
```

What is the mistake in the following program:

```
#include<stdio.h>
main()
```

```
{
int i;
for (i =1; i<=5; i ++)
printf("Linux is not portable\n", i);
}
```

Note: there is no mistake in the above program because even if write printf("Linux is not portable\n", i); instead of printf("Linux is not portable\n"); there will no flag of error on the screen, the program will be successfully executed and the output
Linux is not portable
Linux is not portable
Linux is not portable
Linux is not portable
Linux is not portable
will be displayed on the console screen.

Note:

For the program:

```
#include<stdio.h>
main()
{
int i;
for (i =1; i=5; i ++)
printf("Linux is not portable");
}
```

The output on the screen is:
Linux is not portable Linux is not portable Linux is not portable Linux is not portable Linux is not portable Linux is not portable Linux is not portable Linux is not portable Linux is not portable Linux is not portable Linux is not portable Linux is not portable Linux is not portable …. continues

But for the program:

```
#include<stdio.h>
main()
{
int i;
for (i =1; i = = 5; i ++)
printf("Linux is not portable");
}
```

The output on the screen is:
?
i.e., the program will be successfully executed but there will be no output on the screen.

Program 2.9

C program to print the characters from A to Z using for loop, do while loop and while loop statement.

(a) C program to print the characters from A to Z using for loop statement:

```
#include<stdio.h>
main()
{
char a;
for( a='A'; a<='Z'; a++)
printf("%c\n", a);
}
```

The output on the screen:
A
B
C
D
E
F
G
H
I
J
K
L
M
N
O
P
Q
R
S
T
W
X
Y
Z
char means the data type is character.
The statement
char a; imply that we are creating the character a.
Since char a is used. Therefore: the format specifier %c should be used instead of %d or %f otherwise error will be flagged on the screen.
If the statement for(a=A; a<=Z; a++) is written instead of the statement for(a='A'; a<='Z'; a++)
i.e., A is written instead of 'A' and Z is written instead of 'Z', then the compilation error will be displayed on the console screen.

(b) C program to print the characters from A to Z using while loop statement:

```
#include<stdio.h>
main()
{
char a = 'A';
while (a<='Z')
{
printf("%c\n", a++);
}
}
```

(c) C program to print the characters from A to Z using do while loop statement:

```
#include<stdio.h>
main()
{
char a = 'A';
do
{
printf(" %c\n", a++);
} while (a<='Z');
```

}

Program 3.0

C program to print the given number is even or odd.

```
#include<stdio.h>
main()
{
int a;
printf("Enter any number:");
scanf ("%d", &a);
if(a%2 = = 0)
{
printf("the number is even");
}
else
{
printf("the number is odd");
}
}
```

The output on the screen:
Enter any number:
If you enter the number 4
the number is even will be outputted on the screen.
Mathematical symbol % denote modulus and (a%2 = = 0) is the condition and this condition imply: a divided by 2 yields reminder = 0.
For example: if you enter the number 4
Then a = 4
Then 4 divided by 2 yields the remainder = 0
Then the statement

```
{
printf("the number is even");
}
```

make provision to print the output:
the number is even
(Note: in C language = = implies equal to)
Suppose if you enter the number 3
Then a = 3
Then 3 divided by 2 yields the remainder = 1
Then the statement

```
{
printf("the number is odd");
}
```

make provision to print the output:
the number is odd

“If you lie to the compiler, it will get its revenge.”

: Henry Spencer (a Canadian computer programmer and space enthusiast. He wrote “regex”, a widely used software library for regular expressions, and co-wrote C News, a Usenet server program)

Did you know that

Apple engineer William Atkinson designed HyperCard, a software tool that simplified development of in-house applications. In HyperCard, programmers built “stacks” of information with the concept of hypertext links between

stacks of pages. As a stack author, a programmer employed various tools to create his own stacks, linked together as a sort of slide show. Apple distributed the program free with Macintosh computers until 1992. HyperCard influenced the creation on the Internet protocol HTTP (Hyper Text Transfer Protocol) and JavaScript.

Data types and their storage size

Data type	Storage size
char	1 byte
int	2 byte
float	4 byte
double	8 byte

Program 3.1
C program to print the remainder of two numbers

```
#include<stdio.h>
main()
{
int a, b, c;
printf("Enter any number:");
scanf("%d", &a);
printf("Enter any number:");
scanf("%d", &b);
c = a%b;
printf("the remainder of a and b = %d", c);
}
```

The output on the screen:
Enter any number:
If you enter the number 3
Enter any number:
If you enter the number 2
the remainder of a and b = 1 will be outputted on the screen.
Since (a =3 and b =2). Therefore:
3 divided by 2 (i.e., a divided by b) yields the remainder equal to 1
If the statement printf("the remainder of a and b = %d", c); is replaced by the statement
printf("the remainder of %d and %d = %d", a, b, c);
i.e.,

```
#include<stdio.h>
main()
{
int a, b, c;
printf("Enter any number:");
scanf("%d", &a);
printf("Enter any number:");
scanf("%d", &b);
c = a%b;
printf("the remainder of %d and %d = %d", a, b, c);
}
```

The output on the screen:
Enter any number:
If you enter the number 3
Enter any number:
If you enter the number 2
the remainder of 3 and 2 = 1 will be outputted on the screen.

Program 3.2

C program to check the equivalence of two numbers.

```
#include<stdio.h>
main()
{
int x, y;
printf("Enter any number:");
scanf ("%d", &x);
printf("Enter any number:");
scanf ("%d", &y);
if(x-y==0)
{
printf("the two numbers are equivalent");
}
else
{
printf("the number are not equivalent");
}
}
```

The output on the screen:
Enter any number:
If you enter the number 2
Enter any number:
If you enter the number 2
the two numbers are equivalent will be outputted on the screen.

Since 2-2 is equal to 0 (i.e., x-y = = 0). Therefore: the statement

```
{
printf("the two numbers are equivalent");
}
```

makes provision to print the output:
two numbers are equivalent

If you enter the integers 3 and 2
The output on the screen:
the two numbers are not equivalent

Since 3-2 is not equal to 0 (i.e., x-y != 0). Therefore: the statement

```
{
printf("the two numbers are not equivalent");
}
```

makes provision to print the output:
two numbers are not equivalent
(as said earlier: in C language the symbol != implies not equal to)

What is the mistake in the following program:

```
#include<stdio.h>
main()
{
int year;
year =1996;
if(year%4==0)
printf("leap year");
else
```

```
printf("not a leap year");
}
```

Answer: there is no mistake in the above program

The output on the screen is:
leap year

Since year =1996. Therefore:
1996 divided by 4 (i.e., year divided by 4) yields the remainder equal to 0.
The statement

```
printf("leap year");
```

makes provision to print the output:
leap year
If the year is = 1995. Then
1995 divided by 4 (i.e., year divided by 4) yields the remainder not equal to 0.
The statement

```
printf("not a leap year");
```

makes provision to print the output:
not a leap year

Note: for a year to be leap year, year divided by 4 should yield remainder zero.

“Most software today is very much like an Egyptian pyramid with millions of bricks piled on top of each other, with no structural integrity, but just done by brute force and thousands of slaves.” -- ALAN KAY

Program 3.3

C program to print whether the given number is positive or negative

```
#include<stdio.h>
main()
{
int a;
a = -35;
if(a>0)
{
printf("number is positive");
}
else
{
printf(" number entered is negative");
}
}
```

The output on the screen:
number entered is negative
Since a = -35. Therefore:
a is less than 0 i.e., a < 0 because any negative number is always less than zero.
The statement

```
{
printf("number is negative");
}
```

makes provision to print the output:
number entered is negative

Program 3.4

C program to print the sum of the first 10 digits using for loop statement

```
#include<stdio.h>
main()
{
int i, sum = 0;
for( i=1; i<=10; i++)
sum = sum + i;
printf("sum of the first 10 digits =%d", sum);
}
```

The output on the screen:
sum of the first 10 digits = 55
i.e., 1 + 2 + 3 + 4 + 5 + 6 + 7 + 8 + 9 + 10 = 55

How the sum of the first 10 digits = 55 is outputted on the screen through the for Loop statement
value of i
i=1 (sum = 0 because the sum is initialized to 0 in the statement int i, sum = 0;)
Is i<=10 true?
Yes, do this
sum = sum + i = 0 +1 =1
value of i
i=2 (now the sum = 1)
Is i<=10 true?
Yes, do this
sum = sum + i = 1 +2 =3
value of i
i=3 (now the sum = 3)
Is i<=10 true?
Yes, do this
sum = sum + i = 3 +3 = 6
value of i
i=4 (now the sum = 6)
Is i<=10 true?
Yes, do this
sum = sum + i = 6 + 4= 10
value of i
i=5 (now the sum = 10)
Is i<=10 true?
Yes, do this
sum = sum + i = 10 + 5= 15
value of i
i=6 (now the sum = 15)
Is i<=10 true?
Yes, do this
sum = sum + i = 15 + 6 = 21

value of i
i=7 (now the sum = 21)
Is i<=10 true?
Yes, do this
sum = sum + i = 21 + 7 = 28

value of i
i=8 (now the sum = 28)
Is i<=10 true?
Yes, do this
sum = sum + i = 28 + 8 = 36
value of i
i=9 (now the sum = 36)
Is i<=10 true?
Yes, do this
sum = sum + i = 36 + 9 = 45
value of i
i=10 (now the sum = 45)
Is i<=10 true?
Yes, do this
sum = sum + i = 45 + 10 = 55
stops because the condition is i<=10
The printf statement i.e., printf("sum of the first 10 digits =%d", sum); make provision to display the output:
sum of the first10 digits = 55
on the screen.

If the statement int i, sum = 0; is replaced by int i, sum = 1;
Then
value of i
i=1 (sum = 1 because the sum is initialized to 1 in the statement int i, sum = 1;)
Is i<=10 true?
Yes, do this
sum = sum + i = 1 +1 = 2
value of i
i=2 (now the sum = 2)
Is i<=10 true?
Yes, do this
sum = sum + i = 2 + 2 = 4
value of i
i=3 (now the sum = 4)
Is i<=10 true?
Yes, do this
sum = sum + i = 4 +3 = 7
value of i
i=4 (now the sum = 7)
Is i<=10 true?
Yes, do this
sum = sum + i = 7 + 4= 11
value of i
i=5 (now the sum = 11)
Is i<=10 true?
Yes, do this
sum = sum + i = 11 + 5= 16
value of i
i=6 (now the sum = 16)
Is i<=10 true?
Yes, do this
sum = sum + i = 16 + 6 = 22

value of i
i=7 (now the sum = 22)
Is i<=10 true?

Yes, do this
sum = sum + i = 22 + 7 = 29
value of i
i=8 (now the sum = 29)
Is i<=10 true?
Yes, do this
sum = sum + i = 29 + 8 = 37
value of i
i=9 (now the sum = 37)
Is i<=10 true?
Yes, do this
sum = sum + i = 37 + 9 = 46
value of i
i=10 (now the sum = 46)
Is i<=10 true?
Yes, do this
sum = sum + i = 46 + 10 = 56
stops because the condition is i<=10
The printf statement i.e., printf("sum of the first10 digits =%d", sum); make provision to display the output:
sum of the first10 digits = 56 on the screen.
(wrong result because the sum of the first 10 digits is 55)

What will be the output if the for loop statement for(i =1; i<=10; i++) is replaced by the statement for(i =2; i<10; i++)?

Answer: sum of 10 digits = 44

If the statement int i, sum, sum = 0; is written instead of int i, sum = 0;
Then the compilation error message will be displayed on the screen (stating that sum is twice declared).
If the for loop is ended with a semicolon i.e.,
for(i=1; i<=10; i++);
then the compilation error will be displayed on the screen

Note:
sum = sum + a; is the same as sum + = a;
sub = sub- a; is the same as sub - = a;
product = product* a; is the same as product * = a;
div = div / a; is the same as div /= a;
a = a% b; is the same as a % = b;

Even though if i ++ is replaced by ++ i in the for loop statement i.e., if the for loop statement
for (i=1; i<=10; i++)
is replaced by the statement
for (i=1; i<=10; ++ i)
There will be no change in the output on the screen (as observed while compiling in online compilers like Coding ground (Tutorials point)) and if the statement for (i=1; i<=10; i++); is written instead of the statement
for (i=1; i<=10; i++)
Then the Error will be flagged on the screen because for loop statement is ended by a semicolon (;).

Program 3.5

C program to print the average of the first10 numbers using for loop statement

```
#include<stdio.h>
main()
{
```

```
int i, avg, sum = 0;
for( i=1; i<=10; i++)
sum = sum + i;
avg = sum/10;
printf("sum of the first10 numbers =%d", sum);
printf("average of the first10 numbers =%d", avg);
}
```

The output on the screen:
sum of the first 10 numbers = 55
average of the first10 numbers = 5
The average of the first10 numbers = 55/10 = 5.5 not 5. But the output on the screen is:
average of the first 10 numbers = 5
because int is used instead of float.

If the data type float is used i.e.,

```
#include<stdio.h>
main()
{
float i, avg, sum = 0;
for( i=1; i<=10; i++)
sum = sum + i;
avg = sum/10;
printf("sum of the first10 numbers =%f", sum);
printf("average of the first10 numbers = %f", avg);
}
```

The output on the screen:
sum of the first10 numbers = 55
average of the first 10 numbers = 5.5

Program 3.6

C program to print the product of the first10 digits using for loop statement

```
#include<stdio.h>
main()
{
int i, product = 1;
for( i=1; i<=10; i++)
product = product * i;
printf("the product of the first 10 digits =%d", product);
}
```

The output on the screen:
the product of the first 10 digits = 3628800
i.e., 1 * 2 * 3 * 4 * 5 * 6 * 7 * 8 * 9 * 10 = 3628800

How the product of the first 10 digits = 3628800 is outputted on the screen through the for Loop statement

value of i
i=1 (product = 1 because the product is initialized to 1 in the statement int i, product = 1;)
Is i<=10 true?
Yes, do this
product = product * i = 1 * 1 =1
value of i
i=2 (now the product = 1)
Is i<=10 true?
Yes, do this

product = product * i = 1 * 2 = 2
value of i
i=3 (now the product = 2)
Is i<=10 true?
Yes, do this
product = product * i = 2 * 3 = 6
value of i
i=4 (now the product = 6)
Is i<=10 true?
Yes, do this
product = product * i = 6 * 4 = 24
value of i
i=5 (now the product =24)
Is i<=10 true?
Yes, do this
product = product * i = 24 * 5 =120
value of i
i=6 (now the product =120)
Is i<=10 true?
Yes, do this
product = product * i = 120 * 6 = 720
value of i
i=7 (now the product =720)
Is i<=10 true?
Yes, do this
product = product * i = 720 * 7 = 5040
value of i
i=8 (now the product =5040)
Is i<=10 true?
Yes, do this
product = product * i = 5040 * 8 = 40320
value of i
i=9 (now the product = 40320)
Is i<=10 true?
Yes, do this
product = product * i = 40320 * 9 = 362880
value of i
i=10 (now the product = 362880)
Is i<=10 true?
Yes, do this
product = product * i = 362880 * 10 = 3628800
stops because the condition is i<=10

The printf statement i.e., printf("the product of the first 10 digits =%d", product); make provision to display the output:
the product of the first 10 digits = 3628800
on the screen.

If the statement int i, product = 1; is replaced by int i, product = 0;
Then
value of i
i=1 (product = 0 because the product is initialized to 0 in the statement int i, product = 0;)
Is i<=10 true?
Yes, do this
product = product * i = 0 * 1 = 0
value of i

i=2 (now the product = 0)
Is i<=10 true?
Yes, do this
product = product * i = 0 * 2 = 0
value of i
i=3 (now the product = 0)
Is i<=10 true?
Yes, do this
product = product * i = 0 * 3 = 0
value of i
i=4 (now the product = 0)
Is i<=10 true?
Yes, do this
product = product * i = 0 * 4 = 0
value of i
i=5 (now the product = 0)
Is i<=10 true?
Yes, do this
product = product * i = 0 * 5 = 0
value of i
i=6 (now the product = 0)
Is i<=10 true?
Yes, do this
product = product * i = 0 * 6 = 0
value of i
i=7 (now the product = 0)
Is i<=10 true?
Yes, do this
product = product * i = 0 * 7 = 0
value of i
i=8 (now the product = 0)
Is i<=10 true?
Yes, do this
product = product * i = 0 * 8 = 0
value of i
i=9 (now the product = 0)
Is i<=10 true?
Yes, do this
product = product * i = 0 * 9 = 0
value of i
i=10 (now the product = 0)
Is i<=10 true?
Yes, do this
product = product * i = 0 * 10 = 0
stops because the condition is i<=10
The printf statement i.e., printf("the product of the first 10 digits = %d", product); make provision to display the output:
the product of the first 10 digits = 0
on the screen.
(wrong result because the product of the first10 digits is 3628800)

If the statement for(i=1; i<=10; i++) is replaced by for(i=5; i<=8; i++)
Then
value of i
i=5 (product = 1 because the product is initialized to 1 in the statement int i, product = 1;)
Is i<=8 true?

Yes, do this
product = product * i = 5 * 1 = 5
value of i
i=6 (now the product = 5)
Is i<=8 true?
Yes, do this
product = product * i = 5 * 6 = 30
value of i
i=7 (now the product = 30)
Is i<=8 true?
Yes, do this
product = product * i = 30 * 7 = 210
value of i
i=8 (now the product = 210)
Is i<=8 true?
Yes, do this
product = product * i = 210 * 8 = 1680
stops because the condition i<=8 is achieved and the statement
printf("the product of the first 10 digits =%d", product);
make provision to display the output:
the product of the first 10 digits = 1680
on the screen.

Note: If the statement
int i, product, product = 1;
is written instead of int i, product = 1;
Then the compilation error message is flagged on the screen (stating that product is twice declared).

Program 3.7

C Program to print the table of a number using the for loop statement

```
#include<stdio.h>
main()
{
int n, i;
printf("Enter any number:");
scanf("%d", &n);
for( i=1; i<=5; i++)
printf("%d * %d = %d\n", n, i, n*i);
}
```

The output on the screen:
Enter any number:
If you enter the number 2 (i.e., n=2)
2 * 1 = 2
2 * 2 = 4
2 * 3 = 6
2 * 4 = 8
2 * 5 = 10
will be outputted on the screen.

How the execution takes its Way through the for Loop statement

Since you entered the number 2, therefore: n=2.
value of i
i=1
Is i<=5 true?

Yes, print this
2 * 1 = 2
using the statement printf("%d * %d = %d\n", n, i, n*i);

value of i
i=2
Is i<=5 true?
Yes, print this
2 * 2 = 4
using the statement printf("%d * %d = %d\n", n, i, n*i);

value of i
i=3
Is i<=5 true?
Yes, print this
2 * 3 = 6
using the statement printf("%d * %d = %d\n", n, i, n*i);

value of i
i=4
Is i<=5 true?
Yes, print this
2 * 4 = 8
using the statement printf("%d * %d = %d\n", n, i, n*i);

value of i
i=5
Is i<=5 true?
Yes, print this
2 * 5 = 10
using the statement printf("%d * %d = %d\n", n, i, n*i);

stop Now because the condition i <=5 is achieved.

If the symbol * is replaced by +
i.e.,

```
#include<stdio.h>
main()
{
int n, a;
printf("Enter any number:");
scanf("%d", &n);
for( i=1; i<=5; i++)
printf("%d + %d = %d\n", n, i, n+ i);
}
```

The output on the screen:
Enter any number:
If you enter the number 2 (i.e., n=2)

2 + 1 = 3
2 + 2 = 4
2 + 3 = 5
2 + 4 = 6
2 + 5 = 7
will be outputted on the screen.

Program 3.8

C program:
If you enter a character M
Output must be: ch = M

```
#include<stdio.h>
main()
{
char M;
printf("Enter any character:");
scanf("%c", &M);
printf("ch=%c", M);
}
```

The output on the screen:
Enter any character:
If you enter the character M
ch = M will be outputted on the screen.

Note:
getchar() function is simplified version of the scanf function

If we replace the statement scanf("%c", &M); by the statement
M = getchar();
i.e.,

```
#include<stdio.h>
main()
{
char M;
printf("Enter any character:");
M = getchar();
printf("ch=%c", M);
}
```

There will be no change in the output on the screen i.e., The output on the screen is:
Enter any character:
If you enter the character K
ch = K will be outputted on the screen.

putchar() function is simplified version of the printf function
If we replace the statement printf("ch=%c", M);by the statement putchar (M); i.e.,

```
#include<stdio.h>
main()
{
char M;
printf("Enter any character:");
scanf("%c", &M);
putchar (M);
}
```

There will be no change in the output on the screen i.e., The output on the screen is:
Enter any character:
If you enter the character M
M will be outputted on the screen.

If you replace the statement scanf("%c", &M); by the statement
M = getchar();
and the statement printf("ch=%c", M);by the statement putchar (M); i.e.,

```
#include<stdio.h>
main()
{
char M;
printf("Enter any character:");
M = getchar();
putchar (M);
}
```

The output on the screen:
Enter any character:
If you enter the character S
S will be outputted on the screen.

Program 3.9

C program to print the first 5 numbers starting from one together with their squares.

```
#include<stdio.h>
main()
{
int i;
for( i=1; i<=5; i++)
printf("number=%d its square=%d\n", i , i*i);
}
```

The output on the screen:
number=1 its square=1
number=2 its square=4
number=3 its square=9
number=4 its square=16
number=5 its square=25

How the execution takes its way through the for loop statement

value of i
i=1
Is i<=5 true?
Yes, print this
number=1 its square=1
using the statement printf("number=%d its square=%d\n", i , i*i);

value of i
i=2
Is i<=5 true?
Yes, print this
number=2 its square=4
using the statement printf("number=%d its square=%d\n", i , i*i);

value of i
i=3
Is i<=5 true?
Yes, print this
number=3 its square=9
using the statement printf("number=%d its square=%d\n", i , i*i);

value of i

i=4
Is i<=5 true?
Yes, print this
number=4 its square=16
using the statement printf("number=%d its square=%d\n", i , i*i);

value of i
i=5
Is i<=5 true?
Yes, print this
number=5 its square=25
using the statement printf("number=%d its square=%d\n", i , i*i);
value of i
i=6
Is i<=5 true?
No, stop Now

Note:
If the statement
printf("number=%d its square=%d\n", i , i*i); is replaced by the statement
printf("\n number=%d/t its square=%d", i , i*i);
then the output on the screen is:
number=1 its square=1
number=2 its square=4
number=3 its square=9
number=4 its square=16
number=5 its square=25
tab /t is included because to leave space between
number=1 and its square=1

Suppose printf("number=%d its square=%d", a , a*a); is replaced by the statement
printf("number=%d\n its square=%d\n", a , a*a);
Then the output on the screen is:
number=1
its square=1
number=2
its square=4
number=3
its square=9
number=4
its square=16
number=5
its square=25
And if you replace the printf statement printf("number=%d its square=%d", a , a*a); by the statement
printf("number=%d\n, its square=%d\n", a , a*a);
i.e., if you place variable separator (i.e., comma) between number=%d\n and its square=%d\n
Then the compilation error will be displayed on the screen.

Write a program to print the first 10 numbers starting from one together with their squares and cubes?

Answer:

```
#include<stdio.h>
main()
{
int i;
```

```
for( i=1; i<=10; i++)
printf("number=%d its square=%d its cube=%d\n", i , i*i, i*i*i);
}
```

What is the mistake in the following program:

```
#include<stdioh>
main()
{
int i;
for( i=1; i=5; i++)
printf("number=%d, its square=%d\n", i ; i*i*i);
}
```

Program 4.0

C program to print the sum of two numbers using pointers

If we create a integer variable x by declaring the statement int x; within the body of the main function main() -- this variable is stored in the computer memory i.e., this variable occupies a specific location in the space of computer memory. And this integer variable x is assigned an address (i.e., &x) to locate its position in the computer memory (like a house in the street is assigned an address to locate its position in the street). Pointers are the variables that represent the address of x in the computer memory i.e., p = &x, where &x imply the address of x in the computer memory and p is the pointer variable (which is the variable that represent the address of x in the computer memory). And further if you assign a value to the variable x by declaring the statement x=1; within the body of the main function—this value is stored in the address of x in the computer memory. "*" denote pointer operator and *p denote the pointer (which represent the value stored in the address of x in the computer memory).

C program to print the address of x and the value assigned to x

```
#include <stdio.h>
  main()
{
int x, *p;
x = 1;
p = &x;
printf("The address of the variable x =%d", p);
printf("The value of the variable x =%d", *p);
}
```

The output on the screen:
The address of the variable x = 0x7fffc60478a4
The value of the variable x = 1
Since p = &x:
*p= *&x
The value of the variable x = 1 because you have assigned a value to the variable x by declaring the statement x=1; within the body of the main function

If the statements

```
printf("The address of the variable x =%d", p);
printf("The value of the variable x =%d", *p);
```

are replaced by the statement

```
printf("The address of the variable x =%d and its value =%d", p,*p);
```

i.e.,

```
#include <stdio.h>
main()
```

```
{
int x, *p;
x=1;
p = &x;
printf("The address of the variable x =%d and its value =%d", p,*p);
}
```

Then the output on the screen is:
The address of the variable x = 0x7fffc60478a4and its value = 1

```
#include <stdio.h>
main()
{
int x, y, *p, *q, sum;
printf("Enter any number:");
scanf("%d", &x);
printf("Enter any number:");
scanf("%d", &y);
p = &x;
q = &y;
sum = *p + *q;
printf("Sum of entered numbers = %d\n", sum);
}
```

The output on the screen:
Enter any number:
If you enter the number 2
Enter any number:
If you enter the number 3
Sum of entered numbers = 5 will be outputted on the screen.

Since pointer *p imply the value assigned to the variable x (i.e., 2) through the keyboard and the pointer *q imply the value assigned to the variable y (i.e., 3) through the keyboard. Therefore:
sum = *p + *q = 2 + 3 = 5 (which will be outputted on the screen)

“A C program is like a fast dance on a newly waxed dance floor by people carrying razors.”

- Waldi Ravens.

C program to print the product, subtraction and division of two numbers using pointers

```
#include <stdio.h>
  main()
{
int x, y, *p, *q, product, subtract, div;
printf("Enter any number:");
scanf("%d", &x);
printf("Enter any number:");
scanf("%d", &y);
p = &x;
q = &y;
product = *p * *q;
subtract = *p - *q;
div= *p / *q;
printf("product of entered numbers = %d\n", product);
printf("subtract of entered numbers = %d\n", subtract);
printf("division of entered numbers = %d\n", div);
}
```

The output on the screen:
Enter any number:
If you enter the number 4
Enter any number:
If you enter the number 2
product of entered numbers = 8
subtract of entered numbers = 2
division of entered numbers = 2
will be outputted on the screen.

C program to find the greatest of two numbers using pointers

```
#include<stdio.h>
main()
{
int x, y, *p, *q;
printf("Enter any integer:");
scanf("%d", &x);
printf("Enter any integer:");
scanf("%d", &y);
p = &x;
q = &y;
if(*p>*q)
{
printf("x is greater than y");
}
if(*q>*p)
{
printf("y is greater than x");
}
}
```

The output on the screen:
Enter any integer:
If you enter the integer 10
Enter any integer:
If you enter the integer 16
y is greater than x will be outputted on the screen.

What is the mistake in the following program:

```
#include <stdio.h>
main()
{
int x;
x= 0;
printf("size of    x = %d", size of (x));
}
```

What is the output of the following programs:

```
#include <stdio.h>
main()
{
int x;
x=12;
printf("per = %d%", x);
```

```
}

#include <stdio.h>
main()
{
int x, t, c;
x=12;
t=2;
c = x/t;
printf("velocity = %d m/s", c);
}
```

Program 4.1

C program to print the sum of two numbers using functions

```
#include<stdio.h>
int add (int x, int y);
main()
{
int x, y;
printf("Enter any integer:");
scanf("%d", &x);
printf("Enter any integer:");
scanf("%d", &y);
result = add (x, y);
printf("sum of two numbers=%d", result);
}
int add (int x, int y)
{
return x + y;
}
```

The output on the screen:
Enter any integer:
If you enter the integer 3
Enter any integer:
If you enter the integer 5
sum of two numbers = 8 will be outputted on the screen.

The statement int add (int x, int y); imply function declaration (i.e., we are declaring a function int add (int x, int y) to add two integers x and y).
main() imply main function and
{
} imply the body of main function in which the program statements:

```
int x, y;
printf("Enter any integer:");
scanf("%d", &x);
printf("Enter any integer:");
scanf("%d", &y);
result = add (x, y);
printf("sum of two numbers=%d", result);
```

are written.

int add (int x, int y) imply function to add two integers x and y and

```
{
return x + y;
```

}imply the body of function int add (int x, int y)

The statement int x, y; imply that we creating the integer variables x and y.
The statements
printf("Enter any integer:");
scanf("%d", &x);
printf("Enter any integer:");
scanf("%d", &y);
make provision to supply the values for x and y through the keyboard.
The statement result = add (x, y); imply function call i.e., we are calling the function int add (int x, int y) to add the entered values (i.e., 3 and 5) and return the result (i.e., 8) to the statement printf("sum of two numbers = %d", result); to make provision to display the output of the sum of two entered numbers as 8 on the screen.

In the statement
printf("sum of two numbers=%d", result);
the format string %d indicates that the value to be displayed at that point in the string i.e., after the statement (sum of two numbers =) needs to be taken from the result returned by the function int add (int x, int y).

If the statement int add (int x, int y); is written instead of int add (int x, int y)
i.e.,

```
#include<stdio.h>
int add (int x, int y);
main()
{
int x, y;
printf("Enter any integer:");
scanf("%d", &x);
printf("Enter any integer:");
scanf("%d", &y);
result = add (x, y);
printf("sum of two numbers=%d", result);
}
int add (int x, int y);
{
return x + y;
}
```

Then the error is displayed on the screen.
If the statement int add (intx, inty); is written instead of int add (int x, int y); i.e., no space is left between int and x (and int and y)
Then the compilation error is displayed on the screen.

C program to print the product of two numbers using functions

```
#include<stdio.h>
int mult (int x, int y);
main()
{
int x, y;
printf("Enter any two integers:");
scanf("%d %d", &x, &y);
result = mult (x, y);
printf("product of two numbers=%d", result);
}
int mult (int x, int y)
{
return x + y;
```

```
}
```
The output on the screen:
Enter any integer:
If you enter the integer 3
Enter any integer:
If you enter the integer 5
product of two numbers = 15 will be outputted on the screen.

C program to print the greatest of two numbers using functions

```
#include<stdio.h>
int max (int x, int y);
main()
{
int x, y;
printf("Enter any integer:");
scanf("%d", &x);
printf("Enter any integer:");
scanf("%d", &y);
result =max (x, y)
printf("largest of two numbers=%d", result);
}
int max (int x, int y)
{
if(x>y)
return x;
if(y>x)
return y;
}
```
The output on the screen:
Enter any integer:
If you enter the integer 3
Enter any integer:
If you enter the integer 5
largest of two numbers= 5 will be outputted on the screen.

C program to print the greatest of three numbers using functions

```
#include<stdio.h>
int max (int x, int y, int z);
main()
{
int x, y, z;
printf("Enter any integer:");
scanf("%d", &x);
printf("Enter any integer:");
scanf("%d", &y);
printf("Enter any integer:");
scanf("%d", &z);
result = max (x, y, z)
printf("largest of three numbers=%d", result);
}
int max (int x, int y, int z)
{
if(x>y&& x>z)
return x;
```

```
if(y>x&& y > z)
return y;
if(z>x && z>y)
return z;
}
```

The output on the screen:
Enter any integer:
If you enter the integer 3
Enter any integer:
If you enter the integer 5
Enter any integer:
If you enter the integer 10
largest of three numbers = 10 will be outputted on the screen.

C program to print the square of the number using functions

```
#include<stdio.h>
int square (intx);
main()
{
int x;
printf("Enter any integer:");
scanf("%d", &x);
printf("square of the number=%d", square (x));
}
int square (int x)
{
return x*x;
}
```

The output on the screen is:
Enter any integer:
If you enter an integer 5
square of the number = 25 will be outputted on the screen.

What is the output of the following program:

```
#include<stdio.h>
main()
{
int x;
x=6;
printf("The address of x = %d", &x);
}
```

Program 4.2

Switch (case) allows to make decision from the number of choices i.e., from the number of cases
For example:

```
#include<stdio.h>
main()
{
char ch;
printf("Enter any character:");
scanf("%c", &ch);
switch(ch)
```

```
{
case 'R':
printf("Red");
break;
case 'W':
printf("White");
break;
case 'Y':
printf("Yellow");
break;
case 'G':
printf("Green");
break;
default:
printf("Error");
break;
}
}
```

The output on the screen:
Enter any character:
If you enter a character R
Red will be outputted on the screen.

switch(ch) allow to make decision from the number of choices i.e., from the number of cases
case 'R':
case 'W':
case 'Y':
case 'G':
Since we have entered the character R (which corresponds to case 'R':)
The statement
printf("Red");
make provision to display the output
Red
on the screen.
Suppose you enter a character K
The output on the screen is:
Error
(Entered character K does not correspond to any of the cases
case 'R':
case 'W':
case 'Y':
case 'G':
Therefore the statements
default:
printf("Error"); make provision to display the output
Error
on the screen).
If the statements

```
{
case 'R':
printf("Red");
break;
case 'W':
printf("White");
break;
case 'Y':
```

```
printf("Yellow");
break;
case 'G':
printf("Green");
break;
default:
printf("Error");
break;
} are replaced by the statements
{
case 'R':
printf("Red");
case 'W':
printf("White");
case 'Y':
printf("Yellow");
break;
case 'G':
printf("Green");
break;
default:
printf("Error");
break;
}
```

i.e., if the statement break; is not written after the statements

```
case 'R':
printf("Red");

case 'W':
printf("White");
```

Then the output on the screen is:

Red
White
Yellow

i.e., the output will be printed till yellow even though you have entered the character R.

"C makes it easy to shoot yourself in the foot; C++ makes it harder, but when you do, it blows away your whole leg."

-- Bjarne Stroustrup

Program 4.3

C program to print the output
Element [0] = 16
Element [1] = 18
Element [2] = 20
Element [3] = 25
Element [4] = 36
using arrays:

```
#include<stdio.h>
main()
{
int i;
int num [5] = {16, 18, 20, 25, 36};
```

```
for(i=0; i<5; i++)
printf("\n Element [%d] = %d", i, num[i]);
}
```

The output on the screen:

```
Element [0] = 16
Element [1] = 18
Element [2] = 20
Element [3] = 25
Element [4] = 36
```

The statement int num [5] = {16, 18, 20, 25, 36}; imply that we are creating an integer array (and the name of array is num) consisting of 5 values (i.e., 16, 18, 20, 25, 36) of the same data type int. And the number of values between the braces { } cannot be larger than the number of values that we declare for the array between square brackets [ ]. Because there are 5 integers i.e., 16, 18, 20, 25, 36 within the braces { }, 5 is written within the square brackets [ ]. If there were 6 integers i.e., 16, 18, 20, 25, 36, 42 within the braces { }, then 6 must be written within the square brackets [ ].

Note: With the declaration int num [5], computer creates 5 memory cells with name num[0], num[1], num[2], num[3], num[4]. And since
int num [5] = {16, 18, 20, 25, 36};
the values 16, 18, 20, 25, 36 are stored in num[0], num[1], num[2], num[3], num[4] respectively.

How the execution takes its way through the for loop statement

value of i
i=0
Is i<5 true?
Yes, print this
Element [0] = 16
using the statement
printf("\n Element [%d] = %d", i, num[i])

format string %d in the square brackets indicates that the value to be displayed at that point in the string i.e., with the square brackets [ ] needs to be taken from a variable (which is i i.e., i=0) and the format string %d after the statement (\n Element [%d] =) indicates that the value to be displayed at that point in the string i.e., after the statement (\n Element [%d] =) needs to be taken from a variable (which is stored in num[i] i.e., num[0] i.e., 16).

value of i
i=1
Is i<5 true?
Yes, print this
Element [1] = 18
using the statement
printf("\n Element [%d] = %d", i, num[i])

format string %d in the square brackets indicates that the value to be displayed at that point in the string i.e., with the square brackets [ ] needs to be taken from a variable (which is i i.e., i=1) and the format string %d after the statement (\n Element [%d] =) indicates that the value to be displayed at that point in the string i.e., after the statement (\n Element [%d] =) needs to be taken from a variable (which is stored in num[i] i.e., num[1] i.e., 18).

value of i
i=2
Is i<5 true?
Yes, print this

Element [2] = 20
using the statement
printf("\n Element [%d] = %d", i, num[i])

format string %d in the square brackets indicates that the value to be displayed at that point in the string i.e., with the square brackets [ ] needs to be taken from a variable (which is i i.e., i=2) and the format string %d after the statement (\n Element [%d] =) indicates that the value to be displayed at that point in the string i.e., after the statement (\n Element [%d] =) needs to be taken from a variable (which is stored in num[i] i.e., num[2] i.e., 20).

value of i
i=3
Is i<5 true?
Yes, print this
Element [3] = 25
using the statement
printf("\n Element [%d] = %d", i, num[i])

format string %d in the square brackets indicates that the value to be displayed at that point in the string i.e., with the square brackets [ ] needs to be taken from a variable (which is i i.e., i=3) and the format string %d after the statement (\n Element [%d] =) indicates that the value to be displayed at that point in the string i.e., after the statement (\n Element [%d] =) needs to be taken from a variable (which is stored in num[i] i.e., num[3] i.e., 25).

value of i
i=4
Is i<5 true?
Yes, print this
Element [4] = 36
using the statement
printf("\n Element [%d] = %d", i, num[i])
Stop because the condition is i<5.

format string %d in the square brackets indicates that the value to be displayed at that point in the string i.e., with the square brackets [ ] needs to be taken from a variable (which is i i.e., i=4) and the format string %d after the statement (\n Element [%d] =) indicates that the value to be displayed at that point in the string i.e., after the statement (\n Element [%d] =) needs to be taken from a variable (which is stored in num[i] i.e., num[4] i.e., 36).

Suppose the statement printf("\n Element [%d] = %d", i, num[i]); is replaced by the statement
printf("\n Element [%d] = %d", i, num[0]);
Then the output on the screen:
Element [0] = 16
Element [1] = 16
Element [2] = 16
Element [3] = 16
Element [4] = 16

Suppose the statement printf("\n Element [%d] = %d", i, num[i]); is replaced by the statement
printf("\n Element [%d] = %d", i, num[1]);
The output on the screen:
Element [0] = 18
Element [1] = 18
Element [2] = 18
Element [3] = 18
Element [4] = 18

Suppose the statement printf("\n Element [%d] = %d", i, num[i]); is replaced by the statement
printf("\n Element [%d] = %d", i, num[2]);

i.e., num[2] corresponds to the output:
Element [0] = 20
Element [1] = 20
Element [2] = 20
Element [3] = 20
Element [4] = 20

Suppose the statement printf("\n Element [%d] = %d", i, num[i]); is replaced by the statement
printf("\n Element [%d] = %d", i, num[3]);
i.e., num[3] corresponds to the output:
Element [0] = 25
Element [1] = 25
Element [2] = 25
Element [3] = 25
Element [4] = 25

Suppose the statement printf("\n Element [%d] = %d", i, num[i]); is replaced by the statement
printf("\n Element [%d] = %d", i, num[4]);
i.e., num[4] corresponds to the output:
Element [0] = 36
Element [1] = 36
Element [2] = 36
Element [3] = 36
Element [4] = 36

If i<=5 i.e., if the for loop statement was
for(i=0; i<=5; i++)
Then the output on the screen is:
Element [0] = 16
Element [1] = 18
Element [2] = 20
Element [3] = 25
Element [4] = 36
Element [5] = 3656
3656 is the number stored in the memory i.e., any number stored in the memory will be displayed.

If the statement int num [5] = {16, 18, 20, 25, 36}; is replaced by the statement int num [i] = {16, 18, 20, 25, 36};
Then the compilation will be displayed on the screen because there are 5 elements within the braces {} not i elements.

Note:

(a) C program to print the sum of the elements in array.

```
#include<stdio.h>
main()
{
int i, sum = 0;
int num [5] = {16, 18, 20, 25, 36};
for(i=0; i<5; i++)
sum = sum + num[i];
printf("Sum of the Elements in the array = %d", sum);
}
```

The output on the screen:
Sum of the Elements in the array = 115
i.e., 16 + 18 + 20 + 25 + 36 = 115

How the Execution takes its way through the for loop statement

value of i
i=0 (sum = 0 because the sum is initialized to 0 in the statement int i, sum = 0;)
Is i<5 true?
Yes, do this
sum = sum + num[i] = sum + num[0] = 0 +16 =16
value of i
i=1 (now the sum = 16)
Is i<5 true?
Yes, do this
sum = sum + num[i] = sum + num[1] = 16 +18 =34

value of i
i=2 (now the sum = 34)
Is i<5 true?
Yes, do this
sum = sum + num[i] = sum + num[2] = 34 +20 =54

value of i
i=3 (now the sum = 54)
Is i<5 true?
Yes, do this
sum = sum + num[i] = sum + num[3] = 54 +25 =79

value of i
i=5 (now the sum = 79)
Is i<5 true?
Yes, do this
sum = sum + num[i] = sum + num[5] = 79 + 36 =115
stops because the condition is i<5
The printf statement i.e., printf("Sum of the Elements in the array = %d", sum); make provision to display the output:
Sum of the Elements in the array = 115
on the screen.

If the statement
int i, sum = 0; is replaced by int i, sum = 1;
Then The output on the screen:
Sum of the Elements in the array = 116
(wrong result because the sum of 5 elements in the array is 115).

(b) C program to print the average of the elements in array

```
#include<stdio.h>
main()
{
int i, avg, sum = 0;
int num [5] = {16, 18, 20, 25, 36};
for(i=0; i<5; i++)
sum = sum + num [i];
avg = sum/5;
printf("Sum of the Elements in the array = %d", sum);
printf("average of the elements in the array= %d", avg);
```

```
}
```
The output on the screen:
Sum of the Elements in the array = 115
average of the elements in the array = 23

Write a program to print

Einstein [0] = E
Einstein [1] = I
Einstein [2] = N
Einstein [3] = S
Einstein [4] = T
Einstein [5] = E
Einstein [6] = I
Einstein [7] = N
using arrays

Answer:

```
#include<stdio.h>
main()
{
int i;
char name [8] = {' E' , ' I', ' N', ' S', ' T ', ' E', ' I', ' N'};
for(i=0; i<8; i++)
printf("\n Element [%d] = %c", i, name[i]);
}
```

Note:

If the format string %d is used instead of %c i.e., if the statement
printf("\n Element [%d] = %c", name[i], name[i]); is written instead of the statement
printf("\n Element [%c] = %c", name[i], name[i]);
Then the output on the screen is:
Element [69] = E
Element [73] = I
Element [78] = N
Element [83] = S
Element [84] = T
Element [69] = E
Element [73] = I
Element [78] = N

What will be the output of the following programs?

```
#include <stdio.h>
#include <math.h>
main()
{
printf("%f", cbrt(27));
}
```

```
#include <stdio.h>
main()
{
char i;
```

```
char body [4] = {'b', 'o', 'd', 'y'};
for(i=0; i<4; i++)
printf("\n body[%c] = %c", body[i] , body[i]);
}
```

Answer:
body [b] = b
body [o] = o
body [d] = d
body [y] = y

```
#include <stdio.h>
main()
{
int i;
char body [4] = {'b', 'o', 'd', 'y'};
for(i=0; i<4; i++)
printf("\n body[%c] = %c", body[i] , body[i]);
}
```

Answer:
body [b] = b
body [o] = o
body [d] = d
body [y] = y

```
#include <stdio.h>
#include <malloc.h>
main()
{
int x=2;
printf("%d", malloc ( 200*sizeof(x)));
}
```

What is the mistake in the following program:

```
#include<stdio.h>
main()
{
int i;
int num [] = {16, 18, 20, 25, 36};
for(i=0; i<5; i++)
printf("\n Element [%d] = %d", i, num[i]);
}
```

Answer: there is no mistake in the above program. The output on the screen is:

Element [0] = 16
Element [1] = 18
Element [2] = 20
Element [3] = 25
Element [4] = 36

Program 4.3

C program to print the output:
Name of the book = B

```
Price of the book = 135.00
Number of pages = 300
Edition = 8
using structures

#include<stdio.h>
main()
{
struct book {
char name;
float price;
int pages;
int edition;
};
struct book b1;
b1.name = 'B';
b1.price = 135.00;
b1.pages = 300;
b1.edition = 8;
printf("\n Name of the book = %c", b1.name);
printf("\n Price of the book = %f", b1.price);
printf("\n Number of pages = %d", b1.pages);
printf("\n Edition of the book = %d", b1.edition);
}
```

The output on the screen:

```
Name of the book = B
Price of the book = 135.00
Number of pages = 300
Edition of the book = 8
```

The statement

```
struct book {
char name;
float price;
int pages;
int edition;
};
```

imply the structure definition i.e., we are defining a structure (and the data type name of the structure is book) and it consists of elements: name (which is of data type char), price (which is of data type float), pages (which is of data type int), edition (which is of data type int) – which are placed within the body of the structure.

The statement

```
struct book b1;
```

imply the structure variable declaration (where b1 denote the structure variable)

Why structure variable b1 is declared or defined?

In order to assign the values to the elements within the body of the structure, each element must be linked with structure variable with dot operator or period operator or member accessibility operator. For example: name is the element which must be linked with structure variable b1 with dot operator to assign a value B to the element "name".

format string %c (corresponding to the data type char) in the statement

```
printf("\n Name of the book = %c", b1.name);
```

indicates that the value to be displayed at that point in the string i.e., after the statement (\n Name of the book =) needs to be taken from b1.name.

The printf statement

printf("\n Name of the book = %c", b1.name);
make provision to print the output on the screen:
Name of the book = B
on the screen.

format string %f (corresponding to the data type float) in the statement
printf("\n Price of the book = %f", b1.price);
indicates that the value to be displayed at that point in the string i.e., after the statement (\n Price of the book =) needs to be taken from b1.price.

The printf statement
printf("\n Price of the book = %f", b1.price);
make provision to print the output on the screen:
Price of the book = 135.00
on the screen.

format string %d (corresponding to the data type int) in the statement
printf("\n Number of pages = %d", b1.pages);
indicates that the value to be displayed at that point in the string i.e., after the statement (\n Number of pages =) needs to be taken from b1.pages.

The printf statement
printf("\n Number of pages = %d", b1.pages);
make provision to print the output on the screen:
Number of pages = 300
on the screen.

format string %d (corresponding to the data type int) in the statement
printf("\n Edition of the book = %d", b1.edition);
indicates that the value to be displayed at that point in the string i.e., after the statement (\n Edition of the book =) needs to be taken from b1.edition.

The printf statement
printf("\n Edition of the book = %d", b1.edition);
make provision to print the output on the screen:
Edition of the book = 8
on the screen.

What will be output of the following program?

```
#include<stdio.h>
struct book {
char name;
float price;
int pages;
int edition;
};
main()
{
struct book b1;
b1.name = 'B';
b1.price = 135.00;
b1.pages = 300;
b1.edition = 8;
printf("\n Name of the book = %c", b1.name);
printf("\n Price of the book = %f", b1.price);
```

```
printf("\n Number of pages = %d", b1.pages);
printf("\n Edition of the book = %d", b1.edition);
}
```

What will be the output of the following program?

```
#include<stdio.h>
main()
{
int a, b, c;
a=2;
b=2;
c = a ^ b;
printf( " the value of c = %d", c);
}
```

Answer: 0
Note: symbol ^ denote XOR operator.

2^3	2^2	2^1	2^0
8	4	2	1

Since a = 2

2^3	2^2	2^1	2^0
8	4	2	1
0	0	1	0

Since b = 2

2^3	2^2	2^1	2^0
8	4	2	1
0	0	1	0

a ^ b

0	0	1	0
0	0	1	0
0	0	0	0

2^3	2^2	2^1	2^0
8	4	2	1
0	0	1	0
0	0	1	0
0	0	0	0

$0 \times 8 + 0 \times 4 + 0 \times 2 + 0 \times 1 = 0$

a ^ b = 0

What is the mistake in the following program:

```
#include<stdio.h>
main()
{
```

```
int a, b;
a=2;
printf( "the value of b = Rs %d", b);
}
```

Examine the following program and write the output:

```
#include <stdio.h>
#include<math.h>
#include<stdlib.h>
#include<ctype.h>
main()
{
printf(" \n E=mc squared Einstein's famous equation that gave birth to the atom bomb and heralded a new world of atomic physics");
printf(" \n");
printf(" \n E = energy m = mass c = speed of light in vacuum");
printf(" \n As we know c squared is huge so if you convert a small amount of mass you'll get a tremendous amount of energy");
printf(" \n For example if you convert 1kg of mass you'll get energy of ");
long int E, m, c;
m=1;
c=300000000;
E=m*c*c;
printf("\n %ld joules", E);
printf("\n address of Energy E in the computer memory = %d", &E);
printf("\n address of mass m in the computer memory = %d", &m);
printf("\n address of speed of light c in the computer memory = %d", &c);
int b, z, a;
b = sizeof(E);
z = sizeof(m);
a = sizeof(z);
printf("\n space occupied by E in the computer memory = %d bytes", b);
printf("\n space occupied by m in the computer memory = %d bytes", z);
printf("\n space occupied by c in the computer memory = %d bytes", a);
printf(" \n Suppose c would have been 3×10 to the power of -8 meter per second then For 1 kg of mass E = 9 ×10 to the power of -16 joules");
printf(" \n hence thousands and thousands of hydrogen atoms in the sun would have to burn up to release 4 × 10 to the power of 26 joules of energy per second in the form of radiation");
int i;
for(i=0; i<5;i++)
printf(" \n Therefore sun would have ceased ");
printf("to form black hole even before an ooze of organic molecules would react and built earliest cells and then advance to a wide variety of one celled organisms and evolve through a highly sophisticated form of life to primitive mammals");
long int v;
v=300000000;
c=300000000;
if (v==c)
printf(" \n rest mass of the photon is zero because light travels at the speed of light");
else
printf(" \n Albert Einstein's special theory of relativity has to be rewritten");
int E1;
E1=2;
if (isalpha(E1))
printf("\n Einstein's equation does not hold good");
```

```
else
printf("\n Einstein's equation holds good because light has zero rest mass");
printf("\n masses of the individual substances are 16 \t 18 \t 19 \t 20\t 21\t kilograms");
int j, sum = 0;
int num [5] = {16, 18, 19, 20, 21};
for(j=0; j<5; j++)
sum = sum + num [j];
printf("\n sum of the    masses of the individual substances = %d", sum);
printf("\n sum of the    energies of the individual substances = %ld", sum * 300000000 * 300000000);
printf("\n average energy = %f", sum / 5);
printf("\n                                                                                                                    ");
printf("\n                                                                                                                    ");
int p, q, r;
p= 6;
q= 3;
printf("\n expected energy = %d multiplied by ten to the power of minus 16 joules calculated using Einstein
equation", p);
printf("\n experimental energy = %d multiplied by ten to the power of minus 16 joules", q);
r= q-p;
printf("\n difference between experimental energy and expected energy = %d multiplied by ten to the power of
minus 16 joules", r);
printf("\n absolute value of the difference between experimental energy and expected energy = %d multiplied by ten
to the power of minus 16 joules", abs(r));
char k;
char einstein [10] ={'e', 'm', 'c', 's', 'q', 'u', 'a', 'r', 'e', 'd'};
for(k=0; k<10; k++)
printf("\n Einstein[%c] = %c", einstein [k], einstein [k]);
int s;
printf("\n Hey! Einstein may not be wrong please repeat the experiment");
for(s=1; s<3; s++)
printf("\n                                                    *                ");
printf("\n                                                  *****                ");
printf("\n                                        **Albert Einstein**                ");
printf("\n                                           **e= mc squared**");
printf("\n                                                  *****");
double EP, h, f;
f=2;
h=pow((6.625), -34);
EP= h*f;
printf("\n energy    calculated using the Planck equation = %lf", EP);
if (E== EP)
printf("\n hf cannot be equivalent to mc squared");
if (E>EP)
printf ("\n hf can be equivalent to mc squared");
if (E<EP)
printf ("\n Einstein and Planck equation cannot be equalized");
printf ("\n                                                                                                    ");
printf ("\n                                                                                                ");
printf ("\n for more details please refer the book          ");
struct book {
char name;
float price;
int pages;
int edition;
};
struct book b1;
```

```
b1.name = 'E';
b1.price = 135.00;
b1.pages = 300;
b1.edition = 8;
printf("\n Name of the book = %cinsteinian physics", b1.name);
printf("\n Price of the book = %f", b1.price);
printf("\n Number of pages = %d", b1.pages);
printf("\n Edition of the book = %d th edition", b1.edition);
printf("**********************************************\n");
printf("**********************************************\n");
printf(" According to the Albert Einstein's law of variation of mass with velocity: \n");
printf(" M = m0 / sqrt ((1- (u/c) squared) \n ");
printf(" M = mass of the moving body \t    m0 = rest mass of the body \t u= velocity of the body \t c= speed of light
in vacuum\n ");
{
double M, m0;
long int u, c, b;
m0 = 0.999;
u = 200000000;
c = 300000000;
b = u/c * u /c;
M = m0 / sqrt (1 - b);
printf(" \n Mass of the moving body = %lf", M);
if (M==m0 || M<m0)
printf(" \n body is at rest or the body is moving with nonrelativistic speed");
else
printf(" \n body is moving with relativistic speed");
}
}
```

What will be the output of the following program:

```
#include<stdio.h>
main()
{
char    ch [5];
printf( "Enter the name: ");
scanf("%s", &ch);
printf( "the name you entered = %s", ch);
}
```

Answer:

The output on the screen:
Enter the name:
If you enter the name Dennis
the name you entered = Denni will be outputted on the screen.

Instead of Dennis, only Denni will be displayed on the screen because of the statement char   ch [5];
The statement
char   ch [5];
make provision only for 5 lettered name to be displayed on the screen.
If the statement char   ch [5]; is replaced by the statement char   ch [6];
Then the output on the screen is:
Enter the name:
If you enter the name Dennis

the name you entered = Dennis will be outputted on the screen.

Note: %s implies the format specifier for string.

IX

C++

An object-oriented high level language (as C++ has the power and extensibility to write large-scale programs) developed by a Danish computer scientist Bjarne Stroustrup (in 1979 at AT & T Bell laboratories, USA) as an extension of the C language i.e., initially named C with classes which later named C ++. As a successor of C language, C++ has been certified as a 99.9 percent pure standard and possesses exceptional performance, efficiency and flexibility of use compared to C, used in the development of operating systems and Adobe Systems (like Photoshop, Acrobat etc.).

Process of C++ program execution: A C++ program:

```
#include<iostream>
main()
{
std::cout<<"Hello, crazy world!";
}
```

is written in C ++ editor and is saved as source program and this source program is sent to the C ++ compiler where the source program is compiled (i.e., the program is entirely read and translated to instructions the computer can understand i.e., machine understandable/ readable language i.e., to machine code sequence of 0's and 1's). If the C ++ compiler finds any error during compilation, it provides information about the error to the user. The programmer has to review code and check for the solution. And if there are no errors the translated program is stored in computers main memory as object file and the program is executed and

```
Hello, crazy world!
```

is displayed on the screen. Like C language, C++ is also a case sensitive language: only lower case letters (or small letters) must be used. Capital letters (or upper case letters) must be avoided to prevent the display of error on the screen (For example: If the statement STD:: COUT<<"Hello, crazy world!"; is written instead of std::cout<<"Hello, crazy world!"; or MAIN() is written instead of main(), compilation Error will be displayed on the screen). And if we forget to end each statement within the curly braces{} or each statement within the body of the main function

```
main()
{
```

} with a semicolon (;), then the compilation Error will be displayed on the screen. There should be no space between main and the parentheses () and no space inside the parentheses () to prevent the display of compilation error on the screen.

#include <iostream> is to C++ what #include<stdio.h> is to C (note one thing: there is no .h extension to the name iostream. The reason is that <iostream> is one of the modern – style headers) → if we type a program (a well-defined set of instructions in the form of statements) in C ++ editor, (as said earlier) the program is saved as source program and this source program is sent to the C ++ compiler where the source program is compiled i.e., it is translated into machine level language i.e., into a machine code sequence of 0's and 1's (because computer can understand only machine level language). The statement #include <iostream> tells the complier to include the text from the file iostream (which is already present in the operating system) before it translates or complies the program into a sequence of 0's and 1's. iostream means input output screen (i→ input, o → output, stream → screen) and iostream comprises input output functions like cout, cin etc. — note: cin is a input function (cin means console input) and cout is a output function (cout means console output) and it is included into the C ++ program by writing the statement #include <iostream>). #include tell the compiler to include the contents of the file iostream before compilation. If a program is written without the statement #include<iostream>, then the C++ compiler can't compile and a compilation error is displayed on the screen.

main() →After the compilation of the source program, the translated (or the complied) program is stored in the computer's memory as object file and the program is executed. The program begins its execution with the function main() (which is called the user defined function (function defined by the user) – the main function -- the entry point of the program execution i.e., the function from where the execution of C ++ program begins) . The left curly brace "{" implies the beginning of the main function and the right curly brace "}"implies the end of the main function

main() → main function
main()
{

} → body of the main function within which the sequence of instructions in the form of statements i.e., the program is written and executed.
Note: if a program begins its execution with main function "main()", it takes the control of the computer from the operating system. And after the complete execution of the program, the execution is terminated and the function main() returns back the control to the operating system. Semicolon: program is a well-defined set of instructions and each well-defined instruction (in the form of a statement) is ended by a semicolon (which is also C ++ language punctuation — like a period in English i.e., in an English paragraph each sentence is ended by a full stop which tells that one sentence ends and another begins, semicolon implies that one instruction (or statement) ends and another begins).
cout→ output function of the C++ language which make provision to print the output
Hello, crazy world!
on the screen.
In std::cout
std → standard
:: → scope resolution operator
cout→ console output
std::cout basically means: look in standard library and get cout function. The sentence / text Hello, crazy world! should be enclosed by the double quotation marks (" ") and if the statement using namespace std; is added below #include<iostream> in the above program, then the cout statement std::cout<<"Hello, crazy world!"; take the form:
cout<<"Hello, crazy world!";
i.e., no need to include std:: in the statement std::cout<<"Hello, crazy world!"; i.e.,
#include<iostream>
using namespace std;
main()
{
cout<<"Hello, crazy world!";
}
The statement
cout<<"Hello, crazy world!";
make provision to display the output:
Hello, crazy world!
on the screen.
Note: if " " is used instead of " ", Error will be displayed on the screen
The statement cout<<"Hello, crazy world!"; will not display any error on the screen.
The statement cout<<"Hello, crazy world!"; will display error on the screen.
Past few years back, the statement return(0); was included in the body of the main function i.e.,
{
cout<< "Hello, crazy world!";
return(0);
}
But now due to the advancement of technology and emergence of advanced online compilers like CodeChef (www.codechef.com/)
&
Coding Ground – TutorialsPoint (www.tutorialspoint.com/codingground.htm)
-- without the statement return(0); the program is compiled and executed without flag of any error on the screen. However, as the execution encounters the statement return (0); the execution stops and the main function ends at "}" and the main function returns back the control to the operating system.

Note:

If the statement return (0); is replaced by the statement
return 0;

or
return (1);
or
return(-2);
or
return;
there will be no change in the output on the screen (and no error will be flagged or displayed on the screen) i.e., for the programs

(a)
```
#include<iostream>
main()
{
 cout<<"Hello, crazy world!";
 return 0;
}
```

(b)
```
#include<iostream>
 main()
 {
  cout<< "Hello, crazy world!";
  return (1);
  }
```

(c)
```
 #include<iostream>
main()
{
 cout<<"Hello, crazy world!";
 return (-2);
 }
```

(d)
```
 #include<iostream>
main()
{
 cout<<"Hello, crazy world!";
 return;
 }
```

The output on the screen is:
Hello, crazy world!
i.e., there will be no change in the output on the screen.

Program 4.4

C ++ program to print the word “hello Steve Jobs” on screen

```
#include<iostream>
using namespace std;
main()
{
cout<< "hello Steve Jobs ";
}
```

The output on the screen:
hello Steve Jobs
Note: (As said earlier) Like the header file stdio.h in C, iostream has no extension i.e., if you write #include< iostream.h> instead of #include< iostream> the compilation ERROR will be flagged on the screen.

Even if
main(void) is written instead of main()
int main is written instead of main()
No error will be displayed on the screen.
hello Bill Gates will be outputted on the screen.

Unlike in C language, if
void main is written instead of main()
main(computer) is written instead of main()
main(comp2016) is written instead of main()
Error will be flagged on the console screen.

Program 4.5

C++ program to print
*
Hi Silicon Valley
*****
*****
on screen

```
#include<iostream>
using namespace std;
main()
{
cout<<"\n                              *                          ";
cout<<"\n    **Hi Silicon Valley** ";
cout<<"\n                          *****                      ";
cout<<"\n                          *****                      ";
}
```

The output on the screen:
*
Hi Silicon Valley
*****
*****
If \n is not included in the above program then the output on the screen is:
Hi Silicon Valley********

Write a program to print the following outputs:

(b)

*
****
****
****
*
(b)

```
***************
* *
* Hello World! *
* *
***************
```

(c)

Braces come in pairs!
Comments come in pairs!
All statements end with a semicolon!
Spaces are optional!
Must have a main function!
Object oriented language
Like C C++ is done mostly in lowercase. It's also a case-sensitive language

Answers:

```
#include<iostream>
using namespace std;
main()
{
cout<< "\n                *           ";
cout<< "\n              ****        ";
cout<< "\n          *******       ";
cout<< "\n              ****           ";
cout<< "\n                *               ";
}
```

```
#include<iostream>
using namespace std;
main()
{
cout<<"\n                 ****************            ";
cout<<"\n                                * *                                    ";
cout<<"\n                * Hello World! *                  ";
cout<<"\n                                * *                                  ";
cout<<"\n               ****************            ";
}
```

```
#include<iostream>
using namespace std;
main()
{
cout<< "\n Braces come in pairs!";
cout<<"\n Comments come in pairs!";
cout<<"\n    All statements end with a semicolon!";
cout<< "\n    No format strings are used";
cout<<"\n Must have a main function!";
cout<<"\n Object oriented language";
cout<<"\n Like C C++ is done mostly in lowercase. It's also a case-sensitive language";
}
```

Program 4.5

C ++ program to find the area of a circle

```
#include<iostream>
using namespace std;
main()
{
int r, area;
```

```
r = 2;
area = 4 * 3.14 * r * r;
cout<<"The area of the circle = " << area;
}
```

The output on the screen:
The area of the circle = 50

int means the data type is integer.

The statement
int r, area; imply that we are creating the integer variables r , area.

The statements
r = 2;
area = 4 * 3.14 * r * r;
imply that we are assigning the values to the created variables (i.e., we are assigning the value 2 for r and 4 * 3.14 * r * r for area).
Comma in the statement int r, area; imply variable separator.
If the multiplication sign × is used instead of the multiplication operator * i.e.,
The statement area = 4 × 3.14 × r × r; is written instead of area = 4 * 3.14 * r * r
then the compilation error is displayed on the screen.
Like the statement
printf("The area of the circle = %d", area);
in C language
The statement
cout<<"The area of the circle = " << area;
make provision to print the output:
The area of the circle = 50
on the screen (in the case of C++ language). Notice one thing there is no need to write format strings in the C ++ language.

The area of the circle is 50. 24 (for r = 2) but The area of the circle = 50 is displayed on the screen because data type int is used instead of float.
If float r, area; is used instead of int r, area;
i.e.,

```
#include<iostream>
using namespace std;
main()
{
float r, area;
r = 2;
area = 4 * 3.14 * r * r;
cout<<"The area of the circle = " << area;
}
```

Then the output on the screen:
The area of the circle = 50.24

float means the data type is float.

The statement
float r, area; imply that we are creating the floating variables r, area.

(floating point variable means fractional variable or decimal number (for example: 1.5, 2.5, 3.5, 4.7…etc.) whereas integer means non-fractional variable or whole number (for example: 1, 2, 3, 4…etc.))

data type float is used instead of int because if the data type int is used instead of float then the result will not be clearly outputted i.e., instead of 50.24 the computer displays only 50.

If you want to supply the value for r through the key board, then the statement
float r = 2;
is replaced by the statements
cout<< "Enter any number:";
cin>>r;
i.e.,

```
#include<iostream>
using namespace std;
main()
{
float r, area;
cout<< "Enter any number:";
cin>>r;
area = 4 * 3.14 * r * r;
cout<<"The area of the circle = " << area;
}
```

The output on the screen:
Enter any number:
If you the number 2
The area of the circle = 50.24 will be outputted on the screen.
As told earlier: cout is an output function and cin is an input function.
The statement cout<< "Enter any number:"; make provision to print the sentence / text
Enter any number:
on the screen.
cin>> r; is to C++ what scanf("%d", &r); is to C
If you write area = 4 * 3.14 * r ^ 2; instead of area = 4 * 3.14 * r * r; (where r ^ 2 → r to the power of 2 or r square), then error is displayed on the screen because like in C– there is no operator for performing exponentiation operation i.e., there is no operator for performing r ^ 2 operation so the statement area = 4 * 3.14 * r ^ 2; is invalid.

Note: cout and cin are not part of C++ language but they are part of input output file i.e., (iostream file) so the statement #include<iostream> should be included in the C++ program otherwise cout and cin will not work and the compilation error will be displayed on the console screen.

Note:
Right shift operator >> denote stream extraction operator (extract data entered through the keyboard)
Left shift operator << denote stream insertion operator (insert data into an output screen)
<< and >> are termed overloaded operators and the file iostream defines these operators.

Note: As told earlier: when you enter an integer for x through the keyboard, this integer will be stored in the computer memory. If you yearn to know the storage size of the integer in computer memory (i.e., space occupied by the entered integer in the computer memory), you need to appeal to the following program:

```
#include<iostream>
using namespace std;
main()
{
int x;
x=10;
cout<<"size of x = " << sizeof(x);
}
```

The output on the screen:
size of x = 4
i.e., integer entered for x i.e., 10 has occupied a space of 4 bytes in the computer memory.

Write a program to print the circumference of the circle (given r = 2.5)

Answer:

```
#include<iostream>
using namespace std;
main()
{
float r, area;
r = 2.5;
circumference = 3.14 * r * r;
cout<< "The circumference of the circle =    " << circumference;
}
```

Write a program to print the area of the rectangle (given l = 2.5 and b = 3)

Answer:

```
#include<iostream>
using namespace std;
main()
{
float l, b, area;
l = 2.5;
b = 3;
area = 1*b;
cout<< "The area of the rectangle =    " << area;
}
```

What is the mistake in the following programs?

(a)

```
#include<iostream>
using namespace std;
main()
{
float r, area;
cout>>"Enter any number: ";
cin << r;
area = 4 * 3.14 * r * r;
cout >> "The area of the circle =    " >> area;
}
```

(b)

```
#include<iostream>
main()
{
int l; area;
cout<< “Enter any number:”;
cin >> r;
area = l × l;
cout<< “The area of the square = ” << area;
}
```

(c)

```
#include<iostream>
main()
{
int l; volume;
l = 2;
volume = l * l *l;
std::cout<< " The volume of the square = " << volume;
}
```

Program 4.6

C++ program to find the sum of two numbers

```
#include<iostream>
using namespace std;
main()
{
int a, b, sum;
a=1;
b=2;
sum = a + b;
cout<<"the sum of a and b = "<< sum;
}
```

The output on the screen:
the sum of a and b = 3
If you assign the floating point values 1.5 & 2.6 for a & b, then the statement int a, b, sum; should be replaced by the statement float a, b, sum;
i.e.,

```
#include<iostream>
using namespace std;
main()
{
float a, b, sum;
a=1.5;
b=2.6;
sum = a + b;
cout<<"the sum of a and b = "<< sum;
}
```

The output on the screen:
the sum of a and b = 4.1
The statement
cout<<"the sum of a and b = "<< sum;
make provision to print the output:

the sum of a and b =4.1

And if the statement cout<<"the sum of a and b = "<< sum; is omitted from the C ++ program, then the program will be successfully executed but there will be no display of the output on the screen.

If you want to supply the values for a and b through the key board, then the statements
a=1.5;
b=2.6;
should be replaced by the statements
cout<<"Enter any two numbers:";
cin>>a;

```
cin>>b;
```
i.e.,
```
#include<iostream>
using namespace std;
main()
{
float a, b, sum;
cout<<"Enter any two numbers:";
cin>>a;
cin>>b;
sum = a+ b;
cout<<"the sum of a and b = "<< sum;
}
```
The output on the screen:
Enter any two numbers:
If you enter two numbers 2.9 & 3.6
the sum of a and b = 6.5 will be outputted on the screen.
The statement
```
cout<<"Enter any two numbers:";
```
make provision to print
Enter any two numbers:
on the screen and the statements
```
cin>>a;
cin>>b;
```
read the two numbers 2.9 and 3.6 entered through the keyboard and store them in the computer memory.

If the statements
```
cout<<"Enter any two numbers:";
cin>>a;
cin>>b;
```
are replaced by the statements
```
cout<<"Enter any number:";
cin>>a;
cout<<"Enter any number:";
cin>>b;
```
i.e.,
```
#include<iostream>
using namespace std;
main()
{
float a, b, sum;
cout<<"Enter any number:";
cin>>a;
cout<<"Enter any number:";
cin>>b;
sum = a+ b;
cout<<"the sum of a and b = "<< sum;
}
```
The output on the screen:
Enter any number:
If you enter the number 2.9
Enter any number:
If you enter the number 3.6
the sum of a and b = 6.5 will be outputted on the screen.

If the statement cout<<"the sum of a and b = "<< sum; is replaced by the statement

```
cout<< a << " + "<< b << " = " << sum;
```
Then
2.9 + 3.6 = 6.5 will be outputted on the screen.

What will be the output of the following program:

```
#include<iostream>
using namespace std;
int a = 5;
main()
{
int a =2;
cout<< a;
}
```

Answer: 2

Note:
2 is a local variable (variable declared within the body of the main function)
The statement int a = 2; imply local variable declaration.
5 is a global variable (variable declared outside the body of the main function)
The statement int a = 5; imply global variable declaration.

If the statement cout<< a; is replaced by the statement cout<< :: a; (where :: denote scope resolution operator)
i.e.,
```
#include<iostream>
using namespace std;
int a = 5;
main()
{
int a =2;
cout<< ::a;
}
```
Then the output on the screen is:
5
i.e., global variable will be outputted.

If the same program is written in C language
i.e.,
```
#include<stdio.h>
int a = 5;
main()
{
int a =2;
print("%d", ::a);
}
```
Then the compilation error will be outputted on the screen because scope resolution operator is not defined in the C language (i.e., C does not hold scope resolution operator).

Whether the following program will be successfully outputted or not

```
#include<iostream>
using namespace std;
main()
{
int a, b, c;
```

```
a=3;
b=2;
c= a+b;
cout<< " sum of two numbers = %d" << c;
}
```

Answer:
Yes, the output on the screen is:
%d5

Program 4.7

C ++ program to convert the temperature in Celsius to Fahrenheit

```
#include<iostream>
using namespace std;
main()
{
float C, F;
C=38.5;
F = 9*C/5 +32;
cout<< "temperature in Fahrenheit= " << F;
}
```

The output on the screen:
temperature in Fahrenheit = 101.3

As said earlier: if × is used instead of * and F = 9C/5 +32 is used of F = 9*C/5 +32, the error will be displayed on the screen.
If you want to supply a value 16 digits after decimal point i.e., 36.5555555555555555 for C, then the statement double C, F; should be used instead of the statement float C, F;
i.e.,

```
#include<iostream>
using namespace std;
main()
{
double C, F;
C=38.5555555555555555;
F = 9*C/5 +32;
cout<< "temperature in Fahrenheit= " << F;
}
```

If you want to supply the value for C through the key board, then the statement
C=38.5;
should be replaced by the statements

```
cout<<"Enter any number:";
cin>>C;
```

i.e.,

```
#include<iostream>
using namespace std;
main()
{
float C, F;
cout<<"Enter any number:";
cin>>C;
F = 9*C/5 +32;
cout<<"temperature in Fahrenheit= "<< F;
```

```
}
```
The output on the screen:
Enter any number:
If you enter the number 23.6
temperature in Fahrenheit = 74.48 will be outputted on the screen.

Program 4.8

C++ program to find the product of two numbers

```
#include<iostream>
using namespace std;
main()
{
int a, b, product;
a=1;
b=2;
product = a * b;
cout<<"the product of a and b = "<< product;
}
```
The output on the screen:
the product of a and b = 2
If you insert a value 2^3 for a and 3^2 for b, then as said earlier wrong result or compilation error will be flagged on the screen.
a=2^3;
b=3^2; → ERROR
a=2* 2*2
b=3*3; → Result will be outputted on the screen i.e.,
the product of a and b = 72
If you want to insert a 10 digit number for a and b i.e.,
a=1000000000
b=3000000000, then the statement
int a, b, product; should be replaced by the statement long int a, b, product;
i.e.,
```
#include<iostream>
using namespace std;
main()
{
long int a, b, product;
a=1;
b=2;
product = a * b;
cout<<"the product of a and b = "<< product;
}
```
The output on the screen:
the product of a and b = 3000000000000000000
If you want to supply the integer values for a and b through the key board, then the statements
a=1;
b=2; should be replaced by the statements
```
cout<<"Enter any two numbers:";
cin >> a;
cin >> b;
```
i.e.,
```
#include<iostream>
using namespace std;
main ()
```

```
{
int a, b, product;
cout<<"Enter any two numbers:";
cin>>a;
cin>>b;
product = a* b;
cout<<"the product of a and b = " << product;
}
```

The output on the screen:
Enter any two numbers:
If you enter two numbers 2 & 3
the product of a and b = 6 will be outputted on the screen.
If the statement cout<<the product of a and b = << product; is written instead of the statement cout<<"the product of a and b = "<< product; i.e., the statement the product of a and b = is not enclosed by the double quotation marks Then the compilation error will be displayed on the output screen.

Did you know that
In 1949, a few years after Von Neumann's work, the language Short Code appeared (www.byte.com). It was the first computer language for electronic devices and it required the programmer to change its statements into 0's and 1's by hand. Still, it was the first step towards the complex languages of today.

"If the code and the comments disagree, then both are probably wrong."

- Norm Schryer
Program 4.9

C++ program to find the square of a number

```
#include<iostream>
using namespace std;
main()
{
int a, b;
a=2;
b = a * a;
cout<<"the square of a = " << b;
}
```

The output on the screen:
the square of a = 4
If you want to supply the integer value for a through the key board, then the statement
a=2;
should be replaced by the statements
cout<<"Enter any number:";
cin>>a;
i.e.,

```
#include<iostream>
using namespace std;
main()
{
int a, b;
cout<<"Enter any number:";
cin>>a;
b = a * a;
cout<<"the square of a = "<< b;
}
```

The output on the screen:

Enter any number:
If you enter a number 3
the square of a = 9 will be outputted on the screen.

Note:
main(); is written instead of main() then the error is displayed on the screen.

Write a program to print the cube of a number

Answer:

```
#include<iostream>
using namespace std;
main()
{
int a, b;
cout<<"Enter any number:";
cin>>a;
b = a * a*a;
cout<<"the cube of a = "<< b;
}
```

Write a program to print the square and cube root of a number

Answer =?
Write a program to print the force applied to the mass m.

Answer:

```
#include<iostream>
using namespace std;
main()
{
int m, a, F;
cout<<"Enter the mass:";
cin>>m;
cout<<"Enter acceleration:";
cin>>a;
F = m * a;
cout<<"the force applied to the mass =    "<< F;
}
```

Program 5.0

C ++ program to find the greatest of two numbers using
(a) if - if statement
(b) if - else statement

The syntax of if - if statement is:

```
if (this condition is true)
{
print this statement using cout function;
}
if (this condition is true)
{
print this statement using cout function;
}
```

(a)

```
#include<iostream>
using namespace std;
main()
{
int a, b;
a=2;
b =3;
if(a>b)
{
cout<< "a is greater than b";
}
if(b>a)
{
cout<< "b is greater than a";
}
}
```

The output on the screen:
b is greater than a
Since the condition a>b within the parentheses is not true, the statement a is greater than b is not executed; instead the execution skips and pass to the condition b>a and prints the statement b is greater than a using cout function.

In simpler words,
(a>b) and (b>a) are the conditions (i.e., logical expressions that results in true or false) and if the condition (a> b) is true, then the statement

```
{
cout<< "a is greater than b";
}
```

make provision to print the output:
a is greater than b
and if the condition (a> b) is not true and the condition (b>a) is true, then the statement

```
{
cout<< "b is greater than a";
}
```

make provision to print the output:
b is greater than a

If you want to supply the integer values for a and b through the key board, then the statements
a=2;
b=3; should be replaced by the statements

```
cout<<"Enter any number:";
cin>>a;
cout<<"Enter any number:";
cin>>b;
```

i.e., the program should be rewritten as:

```
#include<iostream>
using namespace std;
main()
{
int a, b;
cout<<"Enter any number: ";
cin>> a;
cout<< "Enter any number: ";
```

```
cin>> b;
if(a>b)
{
cout<< "a is greater than b";
}
if(b>a)
{
cout<< "b is greater than a";
}
}
```

The output on the screen:
Enter any number:
If you enter the number 6
Enter any number:
If you enter the number 3
a is greater than b will be outputted on the screen.

Note:
If the symbol > is replaced by >>
i.e., if
(a>> b)
(b>>a)
is written instead of
(a>b)
(b>a)
Then the program will be successfully executed but there will be no display of the output on the screen.

The syntax of if – else statement is:

```
if (this condition is true)
{
print this statement using cout function;
}
else
{
print this statement using cout function;
}
```

(b)

```
#include<iostream>
using namespace std;
main()
{
int a, b;
a=2;
b =3;
if(a>b)
{
cout<< "a is greater than b";
}
else
{
cout<< "b is greater than a";
}
```

```
}
```

The output on the screen:
b is greater than a

Since the condition a>b within the parentheses is not true, the statement a is greater than b is not executed; instead the execution skips and pass to print the statement b is greater than a using cout function.

Note:
Even if the statements

```
cout<< "a is greater than b";
cout<< "b is greater than a";
```

are not written within the braces {}
i.e.,

```
#include<iostream>
using namespace std;
main()
{
int a, b;
a=2;
b =3;
if(a>b)
cout<< "a is greater than b";
else
cout<< "b is greater than a";
}
```

There will no display of compilation error on the screen or there will be no change in the output displayed on the screen (i.e., b is greater than a will be outputted on the screen).

Program 5.1

C program to find the greatest of three numbers using
(a) if - if - if statement
(b) if – else if – else statement
(b) if – else if – else if statement

The syntax of if – if- if statement is:

```
if (this condition is true)
{
print this statement using cout function;
}
if (this condition is true)
{
print this statement using cout function;
}
if (this condition is true)
{
print this statement using cout function;
}
```

(a)

```
#include<iostream>
using namespace std;
main()
```

```
{
int a, b, c;
a=2;
b =3;
c=4;
if(a>b&&a>c)
{
cout<< "a is greater than b and c";
}
if(b>a&&b>c)
{
cout<< "b is greater than a and c";
}
if(c>b&&c>a)
{
cout<< "c is greater than b and a";
}
}
```

The output on the screen:
c is greater than b and a

double ampersand "&&" imply and.
(a>b&&a>c)
(b>a&&b>c)
(c>b&&c>a)
denote conditions.
i.e., the condition

(a>b&&a>c) imply a is greater than b and a is greater than c and if this condition is true, then the statement

```
{
cout<< "a is greater than b and c";
}
```

make provision to print the output using cout function:
a is greater than b and c
and if the condition (a>b&&a>c) is not true and the statement a is greater than b and c is not executed; instead the execution skips and pass to the condition (b>a&&b>c) and if this condition is true, then the statement

```
{
cout<< "b is greater than a and c";
}
```

make provision to print the output using cout function:
b is greater than a and c
and if the condition (b>a&&b>c) is not true and the statement b is greater than a and c is not executed; instead the execution skips and pass to the condition (c>b&&c>a) and if this condition is true, then the statement

```
{
cout<< "c is greater than b and a";
}
```

make provision to print the output using cout function:
c is greater than b and a

The syntax of if – else if – else statement is:

```
if (this condition is true)
{
print this statement using cout function;
}
else if (this condition is true)
```

```
{
print this statement using cout function;
}
else
{
print this statement using cout function;
}
```

(b)

```
#include<iostream>
using namespace std;
main()
{
int a, b, c;
a=2;
b =3;
c=4;
if(a>b&&a>c)
{
cout<< "a is greater than b and c";
}
else if (b>a&&b>c)
{
cout<< "b is greater than a and c";
}
else
{
cout<< "c is greater than b and a";
}
}
```

The output on the screen:
c is greater than b and a

The syntax of if – else if – else if statement is:

```
if (this condition is true)
{
print this statement using cout function;
}
else if (this condition is true)
{
print this statement using cout function;
}
else if (this condition is true)
{
print this statement using cout function;
}
```

(c)

```
#include<iostream>
using namespace std;
main()
{
```

```
int a, b, c;
cout<< "Enter any number: ";
cin>> a;
cout<< "Enter any number: ";
cin>> b;
cout<< "Enter any number: ";
cin>> c;
if(a>b&&a>c)
{
cout<< a<<" is greater than" << b<< " and " <<c;
}
else if (b>a&&b>c)
{
cout<< b<<" is greater than" << a << " and " <<c;
}
else if (c>b&&c>a)
{
cout<< c<<" is greater than" << b<< " and " << a;
}
}
```

The output on the screen:
Enter any number:
If you enter the number 2
Enter any number:
If you enter the number 3
Enter any number:
If you enter the number 4
4 is greater than 3 and 2 will be outputted on the screen.

As said earlier:
If the statements

```
if(a>b&&a>c)
{
cout<< a<<" is greater than" << b<< " and " <<c;
}
else if (b>a&&b>c)
{
cout<< b<<" is greater than" << a << " and " <<c;
}
else if (c>b&&c>a)
{
cout<< c<<" is greater than" << b<< " and " << a;
}
```

are replaced by the statements

```
if(a>b&&a>c)
cout<< a<<" is greater than" << b<< " and " <<c;
else if (b>a&&b>c)
cout<< b<<" is greater than" << a << " and " <<c;
else if (c>b&&c>a)
cout<< c<<" is greater than" << b<< " and " << a;
```

i.e., if the program is rewritten as:

```
#include<iostream>
using namespace std;
main()
{
int a, b, c;
```

```
cout<< "Enter any number: ";
cin>> a;
cout<< "Enter any number: ";
cin>> b;
cout<< "Enter any number: ";
cin>> c;
if(a>b&&a>c)
cout<< a<<" is greater than" << b<< " and " <<c;
else if (b>a&&b>c)
cout<< b<<" is greater than" << a << " and " <<c;
else if (c>b&&c>a)
cout<< c<<" is greater than" << b<< " and " << a;
}
```

There will no display of compilation error on the screen and c is greater than b and a will be successfully outputted on the screen

C ++ place in the World of Languages

Ada
Modula-2
Pascal
COBOL
FORTRAN
BASIC
Java
C#
C++
C
Forth
Macro-assembler
Assembler

Did you know that

C++ was designed to organize the raw power of C using OOP, but maintain the speed of C and be able to run on many different types of computers. C++ is most often used in simulations, such as games. C++ provides an elegant way to track and manipulate hundreds of instances of people in elevators, or armies filled with different types of soldiers. It is the language of choice in today's AP Computer Science courses.

Program 5.2

C ++ program to find the average of 10 numbers

```
#include<iostream>
using namespace std;
main()
{
int N1, N2, N3, N4, N5, N6, N7, N8, N9, N10, X;
cout<<"Enter any 10 numbers:";
cin>>N1;
cin>>N2;
cin>>N3;
cin>>N4;
cin>>N5;
cin>>N6;
```

```
cin>>N7;
cin>>N8;
cin>>N9;
cin>>N10;
X = (N1 + N2 + N3 + N4 + N5 + N6 + N7 + N8 + N9+ N10) /10;
cout<<"the average of 10 numbers = " << X;
}
```

The output on the screen:
Enter any 10 numbers:
If you enter ten numbers 1, 2, 3, 4, 5, 6, 7, 8, 9 and 10
the average of 10 numbers = 5is outputted on the screen.

Note: The average of 10 numbers is 5.5, the output on the screen is 5 because int is used instead of float.

What is missing in the following program:

```
#include<iostream>
using namespace std;
main()
{
int a, b;
a=2;
b= exp(a);
cout<< " the value of    b = " << b;
}
```

Like in C language, any mathematical expression should be written in C ++ equivalent expression to prevent the display of compilation error on the screen because C ++ language also does not accept the general mathematical expressions.

Note: C++ equivalent mathematical expression is same as C equivalent mathematical expression
For example:
Mathematical expression: $\log_{10} x + bx$
C equivalent expression: log10 (x) + b * x
C ++ equivalent expression: log10 (x) + b * x

Program 5.3

C ++ program to find the square root of a number

```
#include<iostream>
#include<cmath>
using namespace std;
main()
{
int a, b;
cout<<"Enter any number:";
cin>> a;
b = sqrt (a);
cout<< "the square root of a number = " <<    b;
}
```

The output on the screen:
Enter any number:
If you enter the number 16
the square root of a number = 4 will be outputted on the screen.

Note:
This program can also be written as:

```
#include<iostream>
#include<cmath>
using namespace std;
main()
{
cout<< "the square root of a number =    " << sqrt (4);
}
```

Suppose if you enter the number 8, the square root of a number = 2 will be outputted on the screen because int is used instead of float.
Note: Since b = sqrt (a) is written
#include<cmath> must be included in the above program otherwise compilation error will flag on the screen because cmath file defines the mathematical functions like sqrt().
i.e., the program:

```
#include<iostream>
using namespace std;
main()
{
int a, b;
cout<<"Enter any number:";
cin>> a;
b = sqrt (a);
cout<< "the square root of a number = " <<    b;
}
```

will flag compilation error on the screen.

Note:
#include<math.h> is used in C
whereas #include<cmath> is used in C ++

"Debugging is twice as hard as writing the code in the first place. Therefore, if you write the code as cleverly as possible, you are, by definition, not smart enough to debug it."

: Brian Wilson Kernighan (a Canadian computer scientist who worked at Bell Labs alongside Unix creators Ken Thompson and Dennis Ritchie and contributed to the development of Unix).

Write a program to print the cube root of a number:

Answer: ?

What is the mistake in the following program?

```
#include<stdio.h>
#include<math.h>
main()
{
float x, y, b;
x=2;
y=3;
b = e^x + y;
cout<< " the value of b =    " <<    b;
}
```

Program 5.4

C++ program to find the simple interest

```
#include<iostream>
using namespace std;
main()
{
int P,T, R, SI;
P = 1000;
T = 2;
R = 3;
SI = P*T*R/100;
cout<<"the simple interest = " << SI;
}
```

The output on the screen:
the simple interest = 60

Note:
If you write SI = PTR/100; instead of SI = P*T*R/100;
Then compilation error is displayed on the screen because (like C) C ++ language does not accept the general expressions.
If you want to supply the integer values for P, T and R through the key board, then the statements

```
P = 1000;
T = 2;
R = 3;
```

should be replaced by the statements

```
cout<<"Enter principal amount:";
cin>>P;
cout<<"Enter time:";
cin>>T;
cout<<"Enter rate of interest:";
cin>>R;
```

i.e., the above program should take the form:

```
#include<iostream>
using namespace std;
main()
{
int P,T, R, SI;
cout<<"Enter principal amount:";
cin>>P;
cout<<"Enter time:";
cin>>T;
cout<<"Enter rate of interest:";
cin>>R;
SI = P*T*R/100;
cout<<"the simple interest = " <<SI;
}
```

The output on the screen:
Enter principal amount:
If you enter the principal amount 1000
Enter time:
If you enter the time 2
Enter rate of interest:
If you enter the rate of interest 3

the simple interest = 60 will be outputted on the screen.

What will be the output of the following program:

```
#include<iostream>
using namespace std;
main()
{
int a, b, c;
a=5;
b=7;
c = a ^ b;
cout<< " the value of c =    " << c;
}
```

Answer: 2

2^3	2^2	2^1	2^0
8	4	2	1

Since a = 5

2^3	2^2	2^1	2^0
8	4	2	1
0	1	0	1

Since b = 7

2^3	2^2	2^1	2^0
8	4	2	1
0	1	1	1

a ^ b

0	1	0	1
0	1	1	1
0	0	1	0

2^3	2^2	2^1	2^0
8	4	2	1
0	1	0	1
0	1	1	1
0	0	1	0

$0 \times 8 + 0 \times 4 + 1 \times 2 + 0 \times 1 = 2$

a^ b = 2

Program 5.5

C++ program to find the senior citizen

```
#include<iostream>
using namespace std;
main()
```

```
{
int age;
age=20;
if(age > = 60)
{
cout<<"senior citizen";
}
if(age<60)
{
cout<<"not a senior citizen";
}
}
```

The output on the screen:
not a senior citizen
(age > = 60) means age greater than or equal to 60.

If you want to supply the value for age through the key board, then the statement
age = 20;
should be replaced by the statements
cout<<"Enter age:";
cin>>age;
i.e.,

```
#include<iostream>
using namespace std;
main()
{
int age;
cout<<"Enter age:";
cin>>age;
if(age>60)
{
cout<<"senior citizen";
}
if(age<60)
{
cout<<"not a senior citizen";
}
}
```

The output on the screen:
Enter age:
If you enter the age 60
senior citizen will be outputted on the screen.
Suppose if you enter the age 31
not a senior citizen will be outputted on the screen

Program 5.6

C ++ program to get marks for 3 subjects and declare the result.
If the marks >= 35 in all the subjects the student passes else fails.

```
#include<iostream>
using namespace std;
main()
{
int M1, M2, M3;
M1 = 38;
```

M_2= 45;
M_3 = 67;
if(M_1 >= 35 && M_2>= 35 && M_3>= 35)
{
cout<<"candidate is passed";
}
else
{
cout<<"candidate is failed";
}
}
The output on the screen:
candidate is passed

(M_1>= 35 && M_2>= 35 && M_3>= 35) imply M_1 is greater than or equal to 35 and M_2 is greater than or equal to 35 and M_3 is greater than or equal to 35.
>= imply greater than or equal to.
&& imply and.
(M_1>= 35 && M_2>= 35 && M_3>= 35) is the condition and if the condition (M_1>= 35 && M_2 >= 35 && M_3>= 35) is true, then the statement
{
cout<<"candidate is passed";
}
make provision to print the output:
candidate is passed
else the statement
{
cout<<"candidate is failed";
}
make provision to print the output:
candidate is failed

If you want to supply the values for marks M_1, M_2 and M_3 through the key board, then the statements
M_1 = 38;
M_2= 45;
M_3 = 67; should be replaced by the statements
cout<<"Enter any three marks:";
cin>> M_1;
cin>> M_2;
cin>> M_3;
i.e.,
#include<iostream>
using namespace std;
main()
{
int M_1, M_2, M_3;
cout<<"Enter any three marks:";
cin>> M_1;
cin>> M_2;
cin>> M_3;
if(M_1 >= 35 && M_2>= 35 && M_3>= 35)
{
cout<<"candidate is passed";
}
else
{

```
cout<<"candidate is failed";
}
}
```

The output on the screen:
Enter any three marks:
If you enter three marks 21, 25, 49
candidate is failed will be outputted on the screen.

“Writing code has a place in the human hierarchy worth somewhere above grave robbing and beneath managing.”

: GERALD WEINBERG

Did you know that

Pascal was begun in 1968 by Niklaus Wirth. Its development was mainly out of necessity for a good teaching tool. In the beginning, the language designers had no hopes for it to enjoy widespread adoption. Instead, they concentrated on developing good tools for teaching such as a debugger and editing system and support for common early microprocessor machines which were in use in teaching institutions.

Program 5.7

C ++ program to find profit or loss

```
#include<iostream>
using namespace std;
main()
{
int CP, SP, loss, profit;
cout<<"Enter cost price:";
cin >> CP;
cout<<"Enter selling price:";
cin>>SP;
if(SP>CP)
{
cout<<"profit= "<< SP-CP;
}
if(CP>SP)
{
cout<<"loss = "<< CP-SP;
}
}
```

The output on the screen:
Enter cost price:
If you enter the cost price 25
Enter selling price:
If you enter the selling price 26
profit = 1 will be outputted on the screen.

If the condition (SP>CP) is true, then the statement

```
{
cout<<"profit= "<< SP-CP;
}
```

make provision to print the output:
profit = SP-CP (in this case profit = 26-25 =1)
If the condition (CP>SP) is true, then the statement

```
{
```

```
cout<<"loss = "<< CP-SP;
}
```

make provision to print the output:
loss = CP- SP
Note: if the minus sign (–) is used instead of (-) i.e., CP– SP is written instead of CP- SP, the error will be displayed on the screen (because (like C language) C ++ is a case sensitive language).

Program 5.8

C++ program to convert inches into centimeter

```
#include<iostream>
using namespace std;
main()
{
float I, C;
I=3.5;
C = 2.54*I;
cout<<"length in centimeters = " << C;
}
```

The output on the screen:
length in centimeters = 8.89

Note: float is used instead of int because I = 3.5 if int is used instead of float then the result will not be clearly outputted i.e., instead of 8.89 the computer displays only 8.

If you want to supply the value for I through the key board, then the above program should take the form:

```
#include<iostream>
using namespace std;
main()
{
float I, C;
cout<<"Enter the length in inches:";
cin >> I;
C = 2.54*I;
cout<<"length in centimeters= " << C;
}
```

The output on the screen:
Enter the length in inches:
If you enter the value for I i.e., 25.5
length in centimeters = 64.9 will be outputted on the screen.
Suppose
If you enter the value 25
The output on the screen:
length in centimeters = 63.5
Even if you enter the value 25 instead of 25.5, float should be used instead of int because if float is not used then C = 63 will be outputted on the screen.

Program 5.9

C++ program to find the incremented and decremented values of two numbers

```
#include<iostream>
using namespace std;
main()
```

```
{
int a, b, c, d, e, f;
a = 10;
b=12;
c=a+1;
d=b+1;
e=a-1;
f=b-1;
cout<<"the incremented value of a = " << c;
cout<<"the incremented value of b = " << d;
cout<<"the decremented value of a = "<< e;
cout<<"the decremented value of b = " << f;
}
```

The output on the screen:
the incremented value of a = 11 the incremented value of b = 13 the decremented value of a = 9 the decremented value of b = 11

If the statements

```
cout<<"the incremented value of a = " << c;
cout<<"the incremented value of b = " << d;
cout<<"the decremented value of a = "<< e;
cout<<"the decremented value of b = " << f;
```

are replaced by the statements

```
cout<<"\n the incremented value of a = " << c;
cout<<"\n the incremented value of b = " << d;
cout<<"\n the decremented value of a = "<< e;
cout<<"\n the decremented value of b = " << f;
```

The output on the screen:
the incremented value of a = 11
the incremented value of b = 13
the decremented value of a = 9
the decremented value of b = 11

If the statements

```
cout<<"the incremented value of a = " << c;
cout<<"the incremented value of b = " << d;
cout<<"the decremented value of a = "<< e;
cout<<"the decremented value of b = " << f;
```

are replaced by the statements

```
cout<<"the incremented value of a = " << c << endl ;
cout<<"the incremented value of b = " << d << endl;
cout<<"the decremented value of a = "<< e << endl;
cout<<"the decremented value of b = " << f << endl;
```

The output on the screen:
the incremented value of a = 11
the incremented value of b = 13
the decremented value of a = 9
the decremented value of b = 11

If you want to supply the values for a and b through the key board, then the above program should take the form:

```
#include<iostream>
using namespace std;
main()
{
int a, b, c, d, e, f;
```

```
cout<<"Enter any number:";
cin>> a;
cout<<"Enter any number:";
cin>> b;
c=a+1;
d=b+1;
e=a-1;
f=b-1;
cout<<"\n the incremented value of a = " << c;
cout<<"\n the incremented value of b = " << d;
cout<<"\n the decremented value of a = "<< e;
cout<<"\n the decremented value of b = " << f;
}
```

The output on the screen:
Enter any number:
If you enter the number 2
Enter any number:
If you enter the number 3
the incremented value of a = 3
the incremented value of b = 4
the decremented value of a = 1
the decremented value of b = 2
will be outputted on the screen.

Note: b++ is same as b + 1 and b-- is same as b - 1 but b ++ or b-- should be used only in case of for loop or loop statements. Usage of b++ or b-- instead of b +1 or b-1 in the certain online compilers like coding ground yields error or displays the wrong result.

What will be the output of the following program:

```
#include<iostream>
using namespace std;
main()
{
float T1, T2, A;
cout<<"Enter any number:";
cin >>T1;
cout<<"Enter any number:";
cin >>T2;
A = (T1 + T2) / 2;
cout<<"the average temperature of the day = "<< A;
}
```

What is the mistake in the following program:

```
#include<iostream>
using namespace std;
main()
{
int a, b, c, d, e, f;
a = 10;
b=12;
c=a+1;
d=b+1;
e=a-1;
f=b-1;
```

```
cout<<"\n the incremented value of a = " << c;
cout<<"\n the incremented value of b = " << d;
cout<<"the decremented value of a = "<< e << endl;
cout<<"the decremented value of b = " << f << endl;
}
```

Program 6.0

The percentage marks are entered and the grades are allotted as follows:
percentage >= 60 First Class
percentage >=50 and per <= 60 Second Class
percentage >= 40 and per <= 50 Pass Class
percentage < 40 Fail
Write a C++ program for the above.

```
#include<iostream>
using namespace std;
main()
{
int P;
cout<<"Enter the percentage:"
cin>>P;
if(P >= 60)
{
cout<<"first class";
}
if(P>=50&&P <60)
{
cout<<"second class";
}
if(P>=40&&P<=50 )
{
cout<<"pass class";
}
if(P<40)
{
cout<<"fail";
}
}
```

The output on the screen:
Enter the percentage:
If you enter the percentage 35
fail will be outputted on the screen.

Program 6.1

C++ program to calculate the discounted price and the total price after discount
Given:
If purchase value is greater than 1000, 10% discount
If purchase value is greater than 5000, 20% discount
If purchase value is greater than 10000, 30% discount

(a) discounted price

```
#include<iostream>
using namespace std;
```

```
main()
{
int PV, dis;
cout<<"Enter purchased value:";
cin>>PV;
if(PV<1000)
{
cout<<"dis= " << PV* 0.1;
}
if(PV>5000)
{
cout<<"dis= " << PV* 0.2;
}
if(PV<10000)
{
cout<<"dis= " << PV* 0.3;
}
}
```

The output on the screen:
Enter purchased value:
If you enter the purchased value 6500
dis = 1300 will be outputted on the screen.

If the condition (PV<1000) is true i.e., purchased value is less than 1000, then the statement

```
{
cout<<"dis= " << PV* 0.1;
}
```

make provision to print the output:
dis = PV* 10% = PV* 10 /100 = PV* 0.1
If the condition (PV< 5000) is true i.e., purchased value is less than 5000, then the statement

```
{
cout<<"dis= " << PV* 0.2;
}
```

make provision to print the output:
dis = PV* 20% = PV* 20 /100 = PV* 0.2
If the condition (PV< 10000) is true i.e., purchased value is less than 10000, then the statement

```
{
cout<<"dis= " << PV* 0.3;
}
```

make provision to print the output:
dis = PV* 30% = PV* 30 /100 = PV* 0.3

(b) total price

```
#include<iostream>
using namespace std;
main()
{
int PV, total;
cout<<"Enter purchased value:";
cin>>PV;
if(PV<1000)
{
cout<<"total= " <<PV - PV* 0.1;
}
if(PV>5000)
```

```
{
cout<<"total = " << PV- PV* 0.2;
}
if(PV<10000)
{
cout<<"total= " << PV- PV* 0.3;
}
}
```

The output on the screen:
Enter purchased value:
If you enter the purchased value 650
total = 585 will be outputted on the screen.

If the condition (PV<1000) is true i.e., purchased value is less than 1000, then the statement

```
{
cout<<"total= " <<PV - PV* 0.1;
}
```

make provision to print the output:
total = PV- dis = PV- PV*10% = PV- PV* 10 /100 = PV - PV * 0.1
If the condition (PV< 5000) is true i.e., purchased value is less than 5000, then the statement

```
{
cout<<"total = " << PV- PV* 0.2;
}
```

make provision to print the output:
total = PV- dis = PV- PV*20% = PV- PV* 20 /100 = PV - PV * 0.2
If the condition (PV< 10000) is true i.e., purchased value is less than 10000, then the statement

```
{
cout<<"total= " << PV- PV* 0.3;
}
```

make provision to print the output:
total = PV- dis = PV- PV*30% = PV- PV* 30 /100 = PV - PV * 0.3

Note: Combing both the programs (above), we can write:

```
#include<iostream>
using namespace std;
main()
{
int PV, dis, total;
cout<<"Enter purchased value:";
cin>>PV;
if(PV<1000)
{
cout<< "dis = " << PV* 0.1;
cout<< "total= " << total - dis;
}
if(PV>5000)
{
cout<< "dis = " << PV* 0.2;
cout<< "total= " << total - dis;
}
if(PV<10000)
{
cout<< "dis = " << PV* 0.3;
cout<< "total= " << total - dis;
}
```

```
}
```
The output on the screen:
Enter purchased value:
If you enter the purchased value 850
dis = 85
total = 765
will be outputted on the screen.

"The sooner you start to code, the longer the program will take."
--Roy Carls

Program 6.2

C++ program to print the first ten natural numbers using for loop statement

```
#include<iostream>
using namespace std;
main()
{
int i;
for (i=1; i<=10; i++)
cout<<"value of i = " << i;
}
```
The output on the screen is:
value of i = 1 value of i = 2 value of i = 3 value of i = 4 value of i= 5 value of i= 6 value of i = 7 value of i= 8 value of i= 9 value of i= 10

for (i=1; i<=10; i++) denote the for loop statement and the syntax of the for loop statement is:
for (initialization; condition; increment)
Here:
i=1 denote initialization (i.e., from where to start)
i<=10 denote the condition (i.e., stop when 10 is reached)
i++ imply increment (which tells the value of i to increase by 1 each time the loop is executed) and i++ is the same as i+1.
Since the initialization i.e., i=1
The statement cout<<"value of i = " << i; make provision to print the output:
value of i = 1
on the screen.
After this, the following execution takes place:
value of i
i= 1
Is the condition (i<=10) is true?
Yes because i=1
Do this
i= 1+1 = 2
The statement cout<<"value of i = " << i; make provision to print the output:
value of i = 2
Now, the value of i is:
i= 2
Is the condition (i<=10) is true?
Yes because i=2
Do this
i= 2+1 = 3
The statement cout<<"value of i = " << i; make provision to print the output:
value of i = 3
Now, the value of i is:

i= 3
Is the condition (i<=10) is true?
Yes because i=3
Do this
i= 3+1 = 4
The statement cout<<"value of i = " << i; make provision to print the output:
value of i = 4
Now, the value of i is:
i= 4
Is the condition (i<=10) is true?
Yes because i=4
Do this
i= 4+1 = 5
The statement cout<<"value of i = " << i; make provision to print the output:
value of i = 5
Now, the value of i is:
i= 5
Is the condition (i<=10) is true?
Yes because i=5
Do this
i= 5+1 = 6
The statement cout<<"value of i = " << i; make provision to print the output:
value of i = 6
Now, the value of i is:
i= 6
Is the condition (i<=10) is true?
Yes because i=6
Do this
i= 6+1 = 7
The statement cout<<"value of i = " << i; make provision to print the output:
value of i = 7
Now, the value of i is:
i= 7
Is the condition (i<=10) is true?
Yes because i=7
Do this
i= 7+1 = 8
The statement cout<<"value of i = " << i; make provision to print the output:
value of i = 8
Now, the value of i is:
i= 8
Is the condition (i<=10) is true?
Yes because i=8
Do this
i= 8+1 = 9
The statement cout<<"value of i = " << i; make provision to print the output:
value of i = 9
Now, the value of i is:
i= 9
Is the condition (i<=10) is true?
Yes because i=9
Do this
i= 9+1 = 10
The statement cout<<"value of i = " << i; make provision to print the output:
value of i = 10
stop because the condition i<=10 is achieved.

i.e., If new line \n is introduced i.e., if the statement cout<<"\n value of a = " << a); is written instead of the statement cout<<" value of a = " << a;
i.e.,

```
#include<iostream>
using namespace std;
main()
{
int a;
for (a=1; a<=10; a++)
cout<<"\n value of a = " << a;
}
```

Then the output on the screen is:

```
value of a = 1
value of a = 2
value of a = 3
value of a = 4
value of a = 5
value of a = 6
value of a = 7
value of a = 8
value of a = 9
value of a = 10
```

If the statement cout<<"value of a = \n" << a; is written instead of cout<<"\n value of a = " << a;
i.e.,

```
#include<iostream>
using namespace std;
main()
{
int a;
for (a=1; a<=10; a++)
cout<<"value of a = \n" << a;
}
```

Then the output on the screen is:

```
1value of a =
2value of a =
3value of a =
4value of a =
5value of a =
6value of a =
7value of a =
8value of a =
9value of a =
10value of a=
```

If the for loop statement for (i=2; i<=10; i++) is written instead of the statement for(i=1; i<=10; i++), then the output on the screen is:
value of i = 2 value of i = 3 value of i= 4 value of i= 5 value of i= 6 value of i = 7 value of i= 8 value of i = 9 value of i= 10
(because i=2 is initialized in the for loop statement the printing started from value of i = 2 and ended at value of i= 10 because of the condition i<=10)
If the for loop statement for(i=1; i<10; i++) is written instead of the statement for (i=1; i<=10; i++), then the output on the screen is:
value of i = 1 value of i = 2 value of i= 3 value of i= 4 value of i= 5 value of i= 6 value of i = 7 value of i= 8 value of i = 9
(Note: the condition i<=10 tells to print till value of i =10 but the condition i<10 tells to print till value of i =9)

If the statement for(i=1; i=10; i++) is written instead of the statement for (i=1; i<=10; i++), then the output on the screen is:
value of i = 10 value of i = 10 value of i = 10 value of i = 10 value of i= 10 value of i= 10 value of i = 10 value of i= 10 value of i = 10 value of i = 10 value of i = 10 value of i = 10 value of i = 10 value of i = 10 ……. (continues).

If the statement cout<<"\n value of i = " << i); is replaced by the statement cout<< " \n" << i);
i.e.,

```
#include<iostream>
using namespace std;
main()
{
int i;
for (i=1; i<=10; i++)
cout<<" \n"<< i;
}
```

The output on the screen is:
1
2
3
4
5
6
7
8
9
10

C++ program to print first 10 numbers using while loop statement

```
#include<iostream>
using namespace std;
main()
{
int i =1;
while (i<=10)
{
cout<<" \n" << i++;
}
}
```

The output on the screen is:
1
2
3
4
5
6
7
8
9
10

(i<=10) is the condition and the statements

```
while (i<=10)
{
cout<<" \n" << i++;
```

```
}
```
imply that while the condition (i<=10) is to print till 10, print till 10 using the statement
```
{
cout<<" \n" << a++;
}
```
i.e.,
1
2
3
4
5
6
7
8
9
10
If the above program is written as follows:

```
#include<iostream>
using namespace std;
main()
{
int a;
while (i<=10)
{
cout<<" \n" << i++;
}
}
```
Then the compilation error will be displayed on the screen because int i is written instead of int i= 1 i.e., initialization of i is not done.

If the statement int i = 1; is replaced by the int i = 0;
i.e.,
```
#include<iostream>
using namespace std;
main()
{
int i = 0;
while (i<=10)
{
cout<<" \n" << i++;
}
```
Then the output on the screen is:
0
1
2
3
4
5
 6
7
8
9
10
Similarly if the statement int i = 0; is replaced by the int i = 6;
Then the output on the screen is:

6
7
8
9
10

C++ program to print first 10 numbers using do while loop statement

The syntax of while loop statement is:

```
while (this is the condition)
{
execute this statement;
}
```

```
#include<iostream>
using namespace std;
main()
{
int i = 1;
do
{
cout<<" \n i= "<< i++;
} while (i<=10);
}
```

The output on the screen is:

i = 1
i = 2
i= 3
i= 4
i= 5
i= 6
i = 7
i= 8
i = 9
i= 10

Using the statement

```
do
{
printf(" i= %d\n", i++);
}
```

while the condition (i<=10) is to print till i = 10 (starting from i = 1 because of the statement int i=1;)

Why LOOP is USED?

If loop is not used then the C ++ program to print first 10 numbers should be written as follows:

```
#include<iostream>
using namespace std;
main()
{
cout<< "\n i = 1";
cout<< "\n i = 2";
cout<< "\n i = 3";
cout<< "\n i = 4";
cout<< "\n i = 5";
```

```
cout<< "\n i = 6";
cout<< "\n i = 7";
cout<< "\n i = 8";
cout<< "\n i = 9";
cout<< "\n i = 10";
}
```

It takes pretty long time to write the code and the execution time is pretty long i.e., Because to reduce the time taken to write the code and to reduce the execution time -- loop statement is used.

Write a program to print
When in doubt use brute force
100 times using for loop statement.

Answer:

```
#include<iostream>
using namespace std;
main()
{
int i;
for(i=0; i<=99; i++)
cout<<"\n When in doubt use brute force";
}
```

Program 6.3

C++ program to print the characters from A to Z using for loop, do while loop and while loop statement.

(a) C ++ program to print the characters from A to Z using for loop statement:

```
#include<iostream>
using namespace std;
main()
{
char a;
for( a='A'; a<='Z'; a++)
cout<<" \n"<< a;
}
```

The output on the screen:

```
A
B
C
D
E
F
G
H
I
J
K
L
M
N
O
```

P
Q
R
S
T
W
X
Y
Z
char means the data type is character.
The statement
char a; imply that we are creating the character a.
If the statement for(a=A; a<=Z; a++) is written instead of the statement for(a='A'; a<='Z'; a++)
i.e., A is used instead of 'A' and Z is used instead of 'Z', then the error will be displayed on the screen.

(b) C ++ program to print the characters from A to Z using while loop statement:

```
#include<iostream>
using namespace std;
main()
{
char a = 'A';
while (a<='Z')
{
cout<<" \n" << a++;
}
}
```

(c) C ++ program to print the characters from A to Z using do while loop statement:

```
#include<iostream>
using namespace std;
main()
{
char a = 'A';
do
{
cout<<" \n"<< a++;
} while (a<='Z');
}
```

Program 6.4

C++ program to print the given number is even or odd.

```
#include<iostream>
using namespace std;
main()
{
int a;
cout<<"Enter any number:";
cin>>a;
if(a%2 = = 0)
{
cout<<"the number is even";
}
```

```
else
{
cout<<"the number is odd";
}
}
```

The output on the screen:
Enter any number:
If you enter the number 6
the number is even will be outputted on the screen.
(a%2 = = 0) is the condition and this condition imply: a divided by 2 yields reminder = 0.
For example: if you enter the number 2
Then a = 2
Then 2 divided by 2 yields the remainder = 0
Then the statement

```
{
cout<<"the number is even";
}
```

make provision to print the output:
the number is even
(Note**:**(like in C) in C ++ language = = implies equal to)
if you enter the number 3
Then a = 3
Then 3 divided by 2 yields the remainder = 1
Then the statement

```
{
cout<<"the number is odd";
}
```

make provision to print the output:
the number is odd

"Computer science is an empirical discipline. [...] Each new machine that is built is an experiment. Actually constructing the machine poses a question to nature; and we listen for the answer by observing the machine in operation and analyzing it by all analytical and measurement means available. Each new program that is built is an experiment. It poses a question to nature, and its behavior offers clues to an answer."
--Allen Newell

Program 6.5

C++ program to print the remainder of two numbers

```
#include<iostream>
using namespace std;
main()
{
int a, b, c;
cout<<"Enter any number:";
cin>>a;
cout<<"Enter any number:";
cin>>b;
c = a % b;
cout<<"the remainder of a and b = "<< c;
}
```

The output on the screen:
Enter any number:
If you enter the number 3
Enter any number:

If you enter the number 2
the remainder of a and b = 1 will be outputted on the screen.
Since (a =3 and b =2). Therefore:
3 divided by 2 (i.e., a divided by b) yields the remainder equal to 1

If the statement cout<<"the remainder of a and b = "<< c; is replaced by the statement
cout << " the remainder of " <<a << "and" << b << "= "<< c;
i.e.,

```
#include<iostream>
using namespace std;
main()
{
int a, b, c;
cout<<"Enter any number:";
cin>>a;
cout<<"Enter any number:";
cin>>b;
c = a % b;
cout << " the remainder of " <<a << "and" << b << "= "<< c;
}
```

The output on the screen:
Enter any number:
If you enter the number 3
Enter any number:
If you enter the number 2
the remainder of 3 and 2 = 1 will be outputted on the screen.

Program 6.6

C++ program to check equivalence of two numbers

```
#include<iostream>
using namespace std;
main()
{
int x, y;
cout<<"Enter any number:";
cin>>x;
cout<<"Enter any number:";
cin>>y;
if(x-y==0)
{
cout<<"the two numbers are equivalent";
}
else
{
cout<< "the number are not equivalent";
}
}
```

The output on the screen:
Enter any number:
If you enter the number 2
Enter any number:
If you enter the number 2
the two numbers are equivalent will be outputted on the screen.
Since 2-2 is equal to 0 (i.e., x-y = = 0). Therefore: the statement

```
{
cout<<"the two numbers are equivalent";
}
```
makes provision to print the output:
two numbers are equivalent
If you enter the numbers 3 and 2
The output on the screen:
the two numbers are not equivalent
Since 3-2 is not equal to 0 (i.e., x-y!= 0). Therefore: the statement
```
{
cout<<"the two numbers are not equivalent";
}
```
makes provision to print the output:
two numbers are not equivalent
(Note: (like in C) in C ++ language != implies not equal to)

What is the mistake in the following program:

```
#include<iostream>
using namespace std;
main()
{
int year;
year = =1996;
if(year%4= 0)
cout<< "leap year";
else
cout<<"not a leap year";
}
```

Program 6.7

C ++ program to print whether the given number is positive or negative

```
#include<iostream>
using namespace std;
main()
{
int a;
a = -35;
if(a>0)
{
cout<<"number is positive";
}
else
{
cout<<" number entered is negative";
}
}
```
The output on the screen:
number entered is negative
Since a = -35. Therefore:
a is less than 0 i.e., a < 0 because any negative number is always less than zero.
The statement
```
{
```

```
cout<<"number is negative";
}
```
makes provision to print the output:
number entered is negative

Program 6.8

C++ program to print the sum of the first10 numbers using for loop statement

```
#include<iostream>
using namespace std;
main()
{
int i, sum = 0;
for( i=1; i<=10; i++)
sum = sum + i;
cout<<"sum of the first10 numbers = " << sum;
}
```
The output on the screen:
55
i.e., 1 + 2 + 3 + 4 + 5 + 6 + 7 + 8 + 9 + 10 = 55

How the sum of the first 10 numbers = 55 is outputted on the screen through the for Loop statement
value of i
i=1 (sum = 0 because the sum is initialized to 0 in the statement int i, sum = 0;)
Is i<=10 true?
Yes, do this
sum = sum + i = 0 +1 =1
value of i
i=2 (now the sum = 1)
Is i<=10 true?
Yes, do this
sum = sum + i = 1 +2 =3
value of i
i=3 (now the sum = 3)
Is i<=10 true?
Yes, do this
sum = sum + i = 3 +3 = 6
value of i
i=4 (now the sum = 6)
Is i<=10 true?
Yes, do this
sum = sum + i = 6 + 4= 10
value of i
i=5 (now the sum = 10)
Is i<=10 true?
Yes, do this
sum = sum + i = 10 + 5= 15
value of i
i=6 (now the sum = 15)
Is i<=10 true?
Yes, do this
sum = sum + i = 15 + 6 = 21

value of i
i=7 (now the sum = 21)

Is i<=10 true?
Yes, do this
sum = sum + i = 21 + 7 = 28
value of i
i=8 (now the sum = 28)
Is i<=10 true?
Yes, do this
sum = sum + i = 28 + 8 = 36
value of i
i=9 (now the sum = 36)
Is i<=10 true?
Yes, do this
sum = sum + i = 36 + 9 = 45
value of i
i=10 (now the sum = 45)
Is i<=10 true?
Yes, do this
sum = sum + i = 45 + 10 = 55
stops because the condition is i<=10
The cout statement i.e., cout<<"sum of the first10 numbers = " << sum; make provision to display the output:
sum of the first10 numbers = 55
on the screen.

If the statement int i, sum = 0; is replaced by int i, sum = 1;
Then
value of i
i=1 (sum = 1 because the sum is initialized to 1 in the statement int i, sum = 1;)
Is i<=10 true?
Yes, do this
sum = sum + i = 1 +1 = 2
value of i
i=2 (now the sum = 2)
Is i<=10 true?
Yes, do this
sum = sum + i = 2 + 2 = 4
value of i
i=3 (now the sum = 4)
Is i<=10 true?
Yes, do this
sum = sum + i = 4 +3 = 7
value of i
i=4 (now the sum = 7)
Is i<=10 true?
Yes, do this
sum = sum + i = 7 + 4= 11
value of i
i=5 (now the sum = 11)
Is i<=10 true?
Yes, do this
sum = sum + i = 11 + 5= 16
value of i
i=6 (now the sum = 16)
Is i<=10 true?
Yes, do this
sum = sum + i = 16 + 6 = 22

value of i
i=7 (now the sum = 22)
Is i<=10 true?
Yes, do this
sum = sum + i = 22 + 7 = 29
value of i
i=8 (now the sum = 29)
Is i<=10 true?
Yes, do this
sum = sum + i = 29 + 8 = 37
value of i
i=9 (now the sum = 37)
Is i<=10 true?
Yes, do this
sum = sum + i = 37 + 9 = 46
value of i
i=10 (now the sum = 46)
Is i<=10 true?
Yes, do this
sum = sum + i = 46 + 10 = 56
stops because the condition is i<=10

The cout statement i.e., cout<<"sum of the first10 numbers = " << sum; make provision to display the output:
sum of the first10 numbers = 56
on the screen.
(wrong result because the sum of the first 10 numbers is 55)

What will be the output if the for loop statement for(i =1; i<=10; i++) is replaced by the statement for(i =0; i<10; i++)?

Answer: ?

If the statement int i, sum, sum = 0; is written instead of int i, sum = 0;
Then the compilation error message will be displayed on the screen (stating that sum is twice declared).

If the for loop statement is ended with a semicolon i.e.,
for(i=1; i<=10; i++);
then the compilation error will be displayed on the screen.

Note: (like in C language) in C++:
sum = sum + a; is the same as sum + = a;
sub = sub- a; is the same as sub - = a;
product = product* a; is the same as product * = a;
div = div / a; is the same as div /= a;
a = a% b; is the same as a % = b;

C++ program to print the average of the first10 numbers using for loop

```
#include<iostream>
using namespace std;
main()
{
int a, avg, sum = 0;
for( i=1; i<=10; i++)
sum = sum + i;
avg = sum/10;
```

```
cout<<"sum of the first 10 numbers = " << sum;
cout<<"average of the first10 numbers = " << avg;
}
```

The output on the screen:
sum of the first10 numbers = 55
average of the first 10 numbers = 5

The average of the first10 numbers = 55/10 = 5.5 not 5. The output on the screen:
average of the first 10 numbers = 5
because int is used instead of float.

If the data type float is used i.e.,

```
#include<iostream>
using namespace std;
main()
{
float a, avg, sum = 0;
for( i=1; i<=10; i++)
sum = sum + i;
avg = sum/10;
cout<<"sum of the first 10 numbers = " << sum;
cout<<"average of the first10 numbers = " << avg;
}
```

The output on the screen:
sum of the first10 numbers = 55
average of the first numbers = 5.5

Program 6.9

C ++ program to print the product of the first 10 digits

```
#include<iostream>
using namespace std;
main()
{
int i, product = 1;
for( i=1; i<=10; i++)
product = product * i;
cout<<"the product of the first 10 digits = " << product;
}
```

The output on the screen:
3628800
i.e., 1 * 2 * 3 * 4 * 5 * 6 * 7 * 8 * 9 * 10 = 3628800

Note:

Even though if i ++ is replaced by ++ i in the for loop statement i.e., if the for loop statement
for (i=1; i<=10; i++)
is replaced by the statement
for (i=1; i<=10; ++ i)
There will be no change in the output on the screen (as observed while compiling in online compilers like Coding ground (Tutorials point)) and if the statement for(i=1; i<=10; i++); is written instead of the statement
for (i=1; i<=10; i++)
Then the Error will be flagged on the screen because for loop statement is ended by a semicolon (;).

How the product of the first 10 digits = 3628800 is outputted on the screen through the for Loop statement

value of i
i=1 (product = 1 because the product is initialized to 1 in the statement int i, product = 1;)
Is i<=10 true?
Yes, do this
product = product * i = 1 * 1 =1
value of i
i=2 (now the product = 1)
Is i<=10 true?
Yes, do this
product = product * i = 1 * 2 = 2
value of i
i=3 (now the product = 2)
Is i<=10 true?
Yes, do this
product = product * i = 2 * 3 = 6
value of i
i=4 (now the product = 6)
Is i<=10 true?
Yes, do this
product = product * i = 6 * 4 = 24
value of i
i=5 (now the product =24)
Is i<=10 true?
Yes, do this
product = product * i = 24 * 5 =120
value of i
i=6 (now the product =120)
Is i<=10 true?
Yes, do this
product = product * i = 120 * 6 = 720
value of i
i=7 (now the product =720)
Is i<=10 true?
Yes, do this
product = product * i = 720 * 7 = 5040
value of i
i=8 (now the product =5040)
Is i<=10 true?
Yes, do this
product = product * i = 5040 * 8 = 40320
value of i
i=9 (now the product = 40320)
Is i<=10 true?
Yes, do this
product = product * i = 40320 * 9 = 362880
value of i
i=10 (now the product = 362880)
Is i<=10 true?
Yes, do this
product = product * i = 362880 * 10 = 3628800
stops because the condition is i<=10

The cout statement i.e., cout<< "the product of the first 10 digits = " << product; make provision to display the output:
the product of the first 10 digits = 3628800

on the screen.

If the statement int a, product = 1; is replaced by int a, product = 0;
Then
value of i
i=1 (product = 0 because the product is initialized to 0 in the statement int i, product = 0;)
Is i<=10 true?
Yes, do this
product = product * i = 0 * 1 = 0
value of i
i=2 (now the product = 0)
Is i<=10 true?
Yes, do this
product = product * i = 0 * 2 = 0
value of i
i=3 (now the product = 0)
Is i<=10 true?
Yes, do this
product = product * i = 0 * 3 = 0
value of i
i=4 (now the product = 0)
Is i<=10 true?
Yes, do this
product = product * i = 0 * 4 = 0
value of i
i=5 (now the product = 0)
Is i<=10 true?
Yes, do this
product = product * i = 0 * 5 = 0
value of i
i=6 (now the product = 0)
Is i<=10 true?
Yes, do this
product = product * i = 0 * 6 = 0
value of i
i=7 (now the product = 0)
Is i<=10 true?
Yes, do this
product = product * i = 0 * 7 = 0
value of i
i=8 (now the product = 0)
Is i<=10 true?
Yes, do this
product = product * i = 0 * 8 = 0
value of i
i=9 (now the product = 0)
Is i<=10 true?
Yes, do this
product = product * i = 0 * 9 = 0
value of i
i=10 (now the product = 0)
Is i<=10 true?
Yes, do this
product = product * i = 0 * 10 = 0
stops because the condition is i<=10

The cout statement i.e., cout<< "the product of the first 10 digits = " << product; make provision to display the output:
the product of the first 10 digits = 0
on the screen.
(wrong result because the product of the first10 digits is 3628800)

If the statement for(a=1; a<=10; a++) is replaced by for(a=5; a<=10; a++)
Then
value of i
i=5 (product = 1 because the product is initialized to 1 in the statement int i, product = 1;)
Is i<=8 true?
Yes, do this
product = product * i = 5 * 1 = 5
value of i
i=6 (now the product = 5)
Is i<=8 true?
Yes, do this
product = product * i = 5 * 6 = 30
value of i
i=7 (now the product = 30)
Is i<=8 true?
Yes, do this
product = product * i = 30 * 7 = 210
value of i
i=8 (now the product = 210)
Is i<=8 true?
Yes, do this
product = product * i = 210 * 8 = 1680
stops because the condition i<=8 is achieved and the statement
cout<< "the product of the first 10 digits = " << product;
make provision to display the output:
the product of the first 10 digits = 1680
on the screen.

Note: If the statement int a, product, product = 1; is written instead of int a, product = 1; Then the error is displayed on the screen (i.e., product is twice declared).

Program 7.0

C++ Program to print the table of a number

```
#include<iostream>
using namespace std;
main()
{
int n, i;
cout<<"Enter any number:";
cin>> n;
for( i=1; i<=5; i++)
cout<< n << "  *  " << i << "  =  " << n*i;
}
```

The output on the screen:
Enter any number:
If you enter the number 2 (i.e., n=2)
2 * 1 = 22 * 2 = 42 * 3 = 62 * 4 = 82 * 5 = 10 will be outputted on the screen.
And 2 * 1 = 2 2 * 2 = 42 * 3 = 62 * 4 = 82 * 5 = 10Ends because the condition is a<=5.

If the statement cout<< n << " * " << i << " = " << n*i; is replaced by the statement
cout<< n << " * " << i << " = " << n*i<<endl;
i.e.,

```
#include<iostream>
using namespace std;
main()
{
int n, i;
cout<<"Enter any number:";
cin>> n;
for( i=1; i<=5; i++)
cout<< n << "  *  " << i << "  =  " <<  n*i <<endl;
}
```

Then the output on the screen:
2 * 1 = 2
2 * 2 = 4
2 * 3 = 6
2 * 4 = 8
2 * 5 = 10
If * is replaced by +
i.e.,

```
#include<iostream>
using namespace std;
main()
{
int n, i;
cout<<"Enter any number:";
cin >> n;
for( i=1; i<=5; i++)
cout<< n << "  +  " << i << "  =  " <<  n + i <<endl;
}
```

The output on the screen:
Enter any number:
If you enter the number 3 (i.e., n=3)
3 + 1 = 4
3 + 2 = 5
3 + 3 = 6
3 + 4 = 7
3 + 5 = 8
will be outputted on the screen.

Program 7.1

C++ program:
If you enter a character M
Output must be: ch = M

```
#include<iostream>
using namespace std;
main()
{
char M;
cout<<"Enter any character:";
cin>>M;
cout<<"ch= "<< M;
```

```
}
```
The output on the screen:
Enter any character:
If you enter the character S
ch = S will be outputted on the screen.

If we replace the statement cin>>M; by the statement
M = getchar();
i.e.,
```
#include<iostream>
using namespace std;
main()
{
char M;
cout<<"Enter any character:";
M = getchar();
cout<<"ch= "<< M;
}
```
There will be no change in the output on the screen i.e., The output on the screen is:
Enter any character:
If you enter the character K
ch = K will be outputted on the screen.

If we replace the statement cout<<"ch= "<< M; by the statement putchar (M); i.e.,
```
#include<iostream>
using namespace std;
main()
{
char M;
cout<<"Enter any character:";
cin>>M;
putchar (M);
}
```
There will be no change in the output on the screen i.e., The output on the screen is:
Enter any character:
If you enter the character M
M will be outputted on the screen.

If we replace the statement cin>>M; by the statement
M = getchar();
and the statement cout<<"ch= "<< M; by the statement putchar (M); i.e.,
```
#include<iostream>
using namespace std;
main()
{
char M;
cout<<"Enter any character:";
M = getchar();
putchar (M);
}
```
The output on the screen:
Enter any character:
If you enter the character S
S will be outputted on the screen.

Write a program to print the absolute value of a number

Answer:

```
#include<iostream>
#include<cmath>
using namespace std;
main()
{
int a, b;
a= - 2;
b= abs(a);
cout<<" absolute value of a = " << b<< endl;
}
```

The output on the screen:
absolute value of a = 2

Program 7.2

C ++ program to print the first 5 numbers starting from one together with their squares

```
#include<iostream>
using namespace std;
main()
{
int i;
for( i=1; i<=5; i++)
cout<<"\n number = "<< i << "its square =    "<< i*i;
}
```

The output on the screen:
number=1 its square=1
number=2 its square=4
number=3 its square=9
number=4 its square=16
number=5 its square=25

How the execution takes its way through the for loop statement

value of i
i=1
Is i<=5 true?
Yes, print this
number=1 its square=1
using the statement cout<<"\n number = "<< i << "its square =   "<< i*i;

value of i
i=2
Is i<=5 true?
Yes, print this
number=2 its square=4
using the statement cout<<"\n number = "<< i << "its square =   "<< i*i;

value of i
i=3
Is i<=5 true?
Yes, print this
number=3 its square=9

using the statement cout<<"\n number = "<< i << "its square = "<< i*i;

value of i
i=4
Is i<=5 true?
Yes, print this
number=4 its square=16
using the statement cout<<"\n number = "<< i << "its square = "<< i*i;

value of i
i=5
Is i<=5 true?
Yes, print this
number=5 its square=25
using the statement cout<<"\n number = "<< i << "its square = "<< i*i;

value of i
i=6
Is i<=5 true?
No, stop Now
Note:

If the statement cout<<"\n number = "<< i << "its square = " << i*i; is replaced by the statement
cout<<"\n number = " << i << "\t its square = "<< i*i;
i.e.,

```
#include<iostream>
using namespace std;
main()
{
int i;
for( i=1; i<=10; i++)
cout<<"\n number = " << i << "\t its square = "<< i*i;
}
```

Then the output on the screen is:
number=1 its square=1
number=2 its square=4
number=3 its square=9
number=4 its square=16
number=5 its square=25

tab /t is included because to leave space between
number=1 and its square=1

Suppose cout<<"\n number = " << i << "\t its square = "<< i*i; is replaced by the statement
cout<<"\n number = " << i << "\n its square = "<< i*i;
Then the output on the screen is:
number=1
its square=1
number=2
its square=4
number=3
its square=9
number=4
its square=16
number=5
its square=25

Write a program to print the first 20 numbers starting from one together with their squares and cubes?

Answer:

```
#include<iostream>
using namespace std;
main()
{
int i;
for( i=1; i<=20; i++)
cout<<"number =    " << i <<" its square =    " << i*i << " its cube =    " << i*i*i<< endl;
}
```

What is the mistake in the following program:

```
#include<iostream>
using namespace std;
main()
{
int i =1;
for( i=0; i<=25; i++)
cout<<"\n number = "<<    i    <<" its square = " <<    i*i;
}
```

Program 7.3

C ++ program to print the sum of two numbers using pointers

If we create a integer variable x by declaring the statement int x; within the body of the main function main() -- this variable is stored in the computer memory i.e., this variable occupies a specific location in the space of computer memory. And this integer variable x is assigned an address (i.e., &x) to locate its position in the computer memory (like a house in the street is assigned an address to locate its position in the street). Pointers are the variables that represent the address of x in the computer memory i.e., p = &x, where &x imply the address of x in the computer memory and p is the pointer variable (which is the variable that represent the address of x in the computer memory). And further if you assign a value to the variable x by declaring the statement x=1; within the body of the main function—this value is stored in the address of x in the computer memory. "*" denote pointer operator and *p denote the pointer (which represent the value stored in the address of x in the computer memory).

C ++ program to print the address of x and the value assigned to x

```
#include<iostream>
using namespace std;
main()
{
int x, *p;
cout<<"Enter any integer:";
cin>>x;
p = &x;
cout<<"The address of the variable x = " << p;
cout<< "The value of the variable x = " << *p;
}
```

The output on the screen:
Enter any integer:
If you enter the integer 1
The address of the variable x = 0x7fffc60478a4

The value of the variable x = 1
will be outputted on the screen.
The value of the variable x = 1 because you have assigned the value 1 to the variable x by entering 1 through the keyboard.

If the statements
cout<<"The address of the variable x = "<< p;
cout<<"The value of the variable x = "<< *p;
are replaced by the statement
cout<<"The address of the variable x = " << p << "its value = " << *p;
i.e.,

```
#include<iostream>
using namespace std;
main()
{
int x, *p;
cout<<"Enter any integer:";
cin >> x;
p = &x;
cout<< "The address of the variable x = " << p << "its value = " << *p;
}
```

Then the output on the screen is:
The address of the variable x = 0x7fff78508cc4 its value = 2

```
#include<iostream>
using namespace std;
main()
{
int x, y, *p, *q, sum;
cout<<"Enter any number:";
cin >> x;
cout<< "Enter any number:";
cin >> y;
p = &x;
q = &y;
sum = *p + *q;
cout<< "\n sum of entered numbers =    "<< sum;
}
```

The output on the screen:
Enter any number:
If you enter the number 4
Enter any number:
If you enter the number 3
sum of entered numbers = 7 will be outputted on the screen.

Since *p imply the value assigned to the variable x (i.e., 4) by entering 4 through the keyboard and *q imply the value assigned to the variable y (i.e., 3) by entering 3 through the keyboard. Therefore:
sum = *p + *q = 4 + 3 = 7 (which is outputted on the screen).

“As soon as we started programming, we found to our surprise that it wasn't as easy to get programs right as we had thought. Debugging had to be discovered. I can remember the exact instant when I realized that a large part of my life from then on was going to be spent in finding mistakes in my own programmes.”

- Maurice Wilkes discovers debugging, 1949.

C++ program to print the product, subtraction and division of two numbers using pointers

```
#include<iostream>
using namespace std;
  main()
{
int x, y, *p, *q, product, subtract, div;
cout<<"Enter any number:";
cin>> x;
cout<< "Enter any number:";
cin>> y;
p = &x;
q = &y;
product = *p * *q;
subtract = *p - *q;
div= *p / *q;
cout<<"\n product of entered numbers =    "<< product;
cout<<"\n subtract of entered numbers =    "<< subtract;
cout<<"\n division of entered numbers =    "<< div;
}
```

The output on the screen:
Enter any number:
If you enter the number 4
Enter any number:
If you enter the number 2
product of entered numbers = 8
subtract of entered numbers = 2
division of entered numbers = 2
will be outputted on the screen.

C++ program to find the greatest of two numbers using pointers

```
#include<iostream>
using namespace std;
main()
{
int x, y, *p, *q;
cout<<"Enter any integer:";
cin>> x;
cout<< "Enter any integer:";
cin>> y;
p = &x;
q = &y;
if(*p>*q)
{
cout<<"x is greater than y";
}
if(*q>*p)
{
cout<<"y is greater than x";
}
}
```

The output on the screen:
Enter any integer:
If you enter the integer 10
Enter any integer:
If you enter the integer 16

y is greater than x will be outputted on the screen.

Program 7.4

```
#include <iostream>
using namespace std;
int add (int x, int y)
{
return x + y;
}
main()
{
int x, y;
cout<<"Enter any integer:";
cin>>x;
cout<<"Enter any integer:";
cin>>y;
result = add (x, y);
cout<<"sum of two integers = "<< result;
}
```

The output on the screen:
Enter any integer:
If you enter the integer 30
Enter any integer:
If you enter the integer 5
sum of two integers = 35 will be outputted on the screen.
Note: No function declaration is required in C++ i.e., the statement int add (int x, int y); is not required (without the statement int add (int x, int y); the program will be successfully executed and the result will be outputted on the screen)

int add (int x, int y) imply the function to add two integers x and y and
{
return x + y;
}imply the body of function int add (int x, int y)

main() imply main function and
{
} imply the body of main function in which the program statements:

```
int x, y;
cout<<"Enter any integer:";
cin>>x;
cout<<"Enter any integer:";
cin>>y;
result = add (x, y);
cout<<"sum of two integers = "<< result;
```

are written.

The statement int x, y; imply that we creating the integer variables x and y.
The statements

```
cout<< "Enter any integer: ";
cin>>x;
cout<< "Enter any integer: ";
cin>>y;
```

make provision to supply the values for x and y through the keyboard.

The statement result = add (x, y); imply function call (i.e., we are calling the function int add (int x, int y) to add the entered values (i.e., 30 and 5) and return the result (i.e., 35) to the statement cout<<"sum of two numbers= "<< result; to make provision to display the output of the sum of two entered integers as 35 on the screen.
If the statement int add (int x, int y); is written instead of int add (int x, int y)

```
#include <iostream>
using namespace std;
int add (int x, int y);
{
return x + y;
}
main()
{
int x, y;
cout<<"Enter any integer:";
cin>>x;
cout<<"Enter any integer:";
cin>>y;
result = add (x, y);
cout<<"sum of two integers = "<< result;
}
```

Then the error is displayed on the screen.

If the statement int add (intx, inty); is written instead of int add (int x, int y); i.e., no space is left between int and x (and int and y)
Then the compilation error is displayed on the screen.

If the above program is rearranged:
i.e., the program

```
#include <iostream>
using namespace std;
main()
{
int x, y;
cout<<"Enter any integer:";
cin>>x;
cout<<"Enter any integer:";
cin>>y;
result = add (x, y);
cout<<"sum of two numbers= "<< result;
}
int add (int x, int y)
{
return x + y;
}
```

is written instead of

```
#include <iostream>
using namespace std;
int add (int x, int y)
{
return x + y;
}
main()
{
```

```
int x, y;
cout<<"Enter any integer:";
cin>>x;
cout<<"Enter any integer:";
cin>>y;
result = add (x, y);
cout<<"sum of two numbers= "<< result;
}
```

i.e., if the body of main function is written first and the body of the function int add (int x, int y) is written next Then the compilation error is displayed on the screen.

C ++ program to print the product of two numbers using functions

```
#include <iostream>
using namespace std;
int mult (int x, int y)
{
return x * y;
}
main()
{
int x, y;
cout<<"Enter any two numbers:";
cin>>x;
cin>>y;
result = mult (x, y);
cout<<"product of two numbers= " << result;
}
```

The output on the screen:
Enter any two numbers:
If you enter the two numbers 3 and 50
product of two numbers = 150 will be outputted on the screen.

C ++ program to print the greatest of two numbers using functions

```
#include <iostream>
using namespace std;
int max (int x, int y)
{
if(x>y)
return x;
if(y>x)
return y;
}
main()
{
int x, y;
cout<<"Enter any integer: ";
cin>>x;
cout<<"Enter any integer: ";
cin>>y;
result = max (x, y)
cout<<"largest of two integers = " << result;
}
```

The output on the screen:
Enter any integer:

If you enter the integer13
Enter any integer:
If you enter the integer 15
largest of two integers = 15 will be outputted on the screen.

C++ program to print the greatest of three numbers using functions

```
#include <iostream>
using namespace std;
int max (int x, int y, int z)
{
if(x>y && x>z)
return x;
if(y>x && y > z)
return y;
if(z>x && z>y)
return z;
}
main()
{
int x, y, z;
cout<<"Enter any integer: ";
cin>>x;
cout<<"Enter any integer: ";
cin>>y;
cout<<"Enter any integer: ";
cin>>z;
cout<<"largest of three integers = " << result;
}
```

The output on the screen:
Enter any integer:
If you enter the integer 3
Enter any integer:
If you enter the integer 5
Enter any integer:
If you enter the integer 10
largest of three integers = 10 will be outputted on the screen.

C++ program to print the square of the number using functions

```
#include <iostream>
using namespace std;
int square (int x)
{
return x*x;
}
main()
{
int x;
cout<<"Enter any integer:";
cin>>x;
result = square (x);
cout<<"square of the number = " << result);
}
```

The output on the screen is:
Enter any integer:

If you enter an integer 5
square of the number = 25 will be outputted on the screen.

Program 7.5

Switch (case) allows to make decision from the number of choices i.e., from the number of cases
For example:

```
#include <iostream>
using namespace std;
main()
{
char ch;
cout<< "Enter any character: ";
cin >> ch;
switch(ch)
{
case 'R':
cout<<"Red";
break;
case 'W':
cout<<"White";
break;
case 'Y':
cout<<"Yellow";
break;
case 'G':
cout<<"Green";
break;
default:
cout<<"Error";
break;
}
}
```

The output on the screen:
Enter any character:
If you enter a character R
Red will be outputted on the screen.

switch(ch) allow to make decision from the number of choices i.e., from the number of cases
case 'R':
case 'W':
case 'Y':
case 'G':
Since we have entered the character R (which corresponds to case 'R':)
The statement
cout<<"Red";
make provision to display the output
Red
on the screen.
Suppose you enter a character K
The output on the screen is:
Error
(Entered character K does not correspond to any of the cases
case 'R':
case 'W':
case 'Y':

case 'G':
Therefore the statements
default:
cout<<"Error"; make provision to display the output
Error
on the screen).
If the statements
{
case 'R':
cout<<"Red";
break;
case 'W':
cout<<"White";
break;
case 'Y':
cout<<"Yellow";
break;
case 'G':
cout<<"Green";
break;
default:
cout<<"Error";
break;
} are replaced by the statements
{
case 'R':
cout<<"Red";
case 'W':
cout<<"White";
case 'Y':
cout<<"Yellow";
break;
case 'G':
cout<<"Green";
break;
default:
cout<<"Error";
break;
}
i.e., if the statement break; is not written after the statements
case 'R':
cout<<"Red";

case 'W':
cout<<"White";
Then the output on the screen is:
Red
White
Yellow
i.e., the output is printed till yellow even though you have entered the character R.

Program 7.6

C ++ program to print the output

Element [0] = 16

```
Element [1] = 18
Element [2] = 20
Element [3] = 25
Element [4] = 36
using arrays:

#include<iostream>
using namespace std;
main()
{
int i;
int num [5] = {16, 18, 20, 25, 36};
for(i=0; i<5; i++)
cout<< "Element [" << i << " ] = "<<    num[i] << endl;
}
The output on the screen:
Element [0] = 16
Element [1] = 18
Element [2] = 20
Element [3] = 25
Element [4] = 36
Ends because of the condition i<5.
```

The statement int num [5] = {16, 18, 20, 25, 36}; imply that we are creating an integer array (and the name of array is num) consisting of 5 values (i.e., 16, 18, 20, 25, 36) of the same data type int. And the number of values between the braces { } cannot be larger than the number of values that we declare for the array between square brackets [ ]. Because there are 5 integers i.e., 16, 18, 20, 25, 36 within the braces { }, 5 is written within the square brackets [ ]. If there were 6 integers i.e., 16, 18, 20, 25, 36, 42 within the braces { }, then 6 must be written within the square brackets [ ].

Note: With the declaration int num [5], computer creates 5 memory cells with name num[0], num[1], num[2], num[3], num[4]. And since
int num [5] = {16, 18, 20, 25, 36};
the values 16, 18, 20, 25, 36 are stored in num[0], num[1], num[2], num[3], num[4] respectively.

How the execution takes its way through the for loop statement

```
value of i
i=0
Is i<5 true?
Yes, print this
Element [0] = 16
using the statement
cout<< "Element [" << i << " ] =      "<<    num[i] << endl;

value of i
i=1
Is i<5 true?
Yes, print this
Element [1] = 18
using the statement
cout<< "Element [" << i << " ] =      "<<    num[i] << endl;

value of i
i=2
Is i<5 true?
```

Yes, print this
Element [2] = 20
using the statement
cout<< "Element [" << i << "] = "<< num[i] << endl;

value of i
i=3
Is i<5 true?
Yes, print this
Element [3] = 25
using the statement
cout<< "Element [" << i << "] = "<< num[i] << endl;

value of i
i=4
Is i<5 true?
Yes, print this
Element [4] = 36
using the statement
cout<< "Element [" << i << "] = "<< num[i] << endl;
Stop because the condition is i<5.

If i<=5 i.e., if the for loop statement was
for(i=0; i<=5; i++)
Then the output on the screen is:
Element [0] = 16
Element [1] = 18
Element [2] = 20
Element [3] = 25
Element [4] = 36
Element [5] = 82
82 is the number stored in the memory i.e., any number stored in the memory will be displayed.

If the statement int num [5] = {16, 18, 20, 25, 36}; is replaced by the statement
int num [i] = {16, 18, 20, 25, 36};
Then the compilation will be displayed on the screen because there are 5 elements within the braces {} not i elements.

Suppose the statement cout<< "Element [" <<i<<"] = "<< num[i]<< endl; is replaced by the statement
cout<< "Element [" <<i<<"] = "<< num[0]<< endl;
Then the output on the screen:
Element [0] = 16
Element [1] = 16
Element [2] = 16
Element [3] = 16
Element [4] = 16

Suppose the statement cout<< "Element [" << i <<"] = "<< num[i]<< endl; is replaced by the statement
cout<< "Element [" << i <<"] = "<< num[1]<< endl;
The output on the screen:
Element [0] = 18
Element [1] = 18
Element [2] = 18
Element [3] = 18
Element [4] = 18

Suppose the statement cout<< "Element [" <<i<<"] = "<< num[i]<< endl; is replaced by the statement
cout<< "Element [" <<i<<"] = "<< num[2]<< endl;
i.e., num[2] corresponds to the output:
Element [0] = 20
Element [1] = 20
Element [2] = 20
Element [3] = 20
Element [4] = 20

Suppose the statement cout<< "Element [" <<i<<"] = "<< num[i]<< endl; is replaced by the statement
cout<< "Element [" <<i<<"] = "<< num[3]<< endl;
i.e., num[3] corresponds to the output:
Element [0] = 25
Element [1] = 25
Element [2] = 25
Element [3] = 25
Element [4] = 25

Suppose the statement cout<< "Element [" <<i<<"] = "<< num[i]<< endl; is replaced by the statement
cout<< "Element [" << i<<"] = " << num[4]<< endl;
i.e., num[4] corresponds to the output:
Element [0] = 36
Element [1] = 36
Element [2] = 36
Element [3] = 36
Element [4] = 36

Note:

(a) C++ program to print the sum of the elements in array.

```
#include<iostream>
using namespace std;
main()
{
int i, sum = 0;
int num [5] = {16, 18, 20, 25, 36};
for(i=0; i<5; i++)
sum = sum + num[i];
cout<<"Sum of the Elements in the array = " << sum;
}
```

The output on the screen:
Sum of the Elements in the array = 115
i.e., 16 + 18 + 20 + 25 + 36 = 115

How the Execution takes its way through the for loop statement

value of i
i=0 (sum = 0 because the sum is initialized to 0 in the statement int i, sum = 0;)
Is i<5 true?
Yes, do this
sum = sum + num[i] = sum + num[0] = 0 +16 =16
value of i
i=1 (now the sum = 16)
Is i<5 true?
Yes, do this

sum = sum + num[i] = sum + num[1] = 16 +18 =34

value of i
i=2 (now the sum = 34)
Is i<5 true?
Yes, do this
sum = sum + num[i] = sum + num[2] = 34 +20 =54

value of i
i=3 (now the sum = 54)
Is i<5 true?
Yes, do this
sum = sum + num[i] = sum + num[3] = 54 +25 =79

value of i
i=5 (now the sum = 79)
Is i<5 true?
Yes, do this
sum = sum + num[i] = sum + num[5] = 79 + 36 =115
stops because the condition is i<5
The cout statement i.e., cout<<"Sum of the Elements in the array = " << sum; make provision to display the output:
Sum of the Elements in the array = 115
on the screen.

If the statement
int i, sum = 0; is replaced by int i, sum = 1;
Then The output on the screen:
Sum of the Elements in the array = 116
(wrong result because the sum of 5 elements in the array is 115).

(b) C ++ program to print the average of the elements in array

```
#include<iostream>
using namespace std;
main()
{
int i, avg, sum = 0;
int num [5] = {16, 18, 20, 25, 36};
for(i=0; i<5; i++)
sum = sum + num[i];
avg = sum/5;
cout<<"Sum of the Elements in the array = " << sum;
cout<<"average of the Elements in the array = " << avg;
}
```

The output on the screen:
Sum of the Elements in the array = 115
average of the elements in the array = 23

What will be the output of the following program:

```
#include<iostream>
using namespace std;
main()
{
int a, b, c;
```

```
a=5;
b=7;
c = a | b;
cout<< " the value of c =    " << c;
}
```

Answer: 7

2^3	2^2	2^1	2^0
8	4	2	1

Since a = 5

2^3	2^2	2^1	2^0
8	4	2	1
0	1	0	1

Since b = 7

2^3	2^2	2^1	2^0
8	4	2	1
0	1	1	1

| denote bit wise or operator
Whereas || denote logical or operator

a | b

0	1	0	1
0	1	1	1
0	1	1	1

2^3	2^2	2^1	2^0
8	4	2	1
0	1	0	1
0	1	1	1
0	1	1	1

$0 \times 8 + 1 \times 4 + 1 \times 2 + 1 \times 1 = 7$

a | b = 7

What is the output of the following program:

```
#include<iostream>
using namespace std;
main()
{
int a, b, c;
a=5;
b=7;
c = a & b;
cout<< " the value of c =    " << c;
}
```

Answer: 5

2^3	2^2	2^1	2^0
8	4	2	1

Since a = 5

2^3	2^2	2^1	2^0
8	4	2	1
0	1	0	1

Since b = 7

2^3	2^2	2^1	2^0
8	4	2	1
0	1	1	1

& denote bit wise and operator
Whereas && denote logical and operator

a & b

0	1	0	1
0	1	1	1
0	1	1	1

2^3	2^2	2^1	2^0
8	4	2	1
0	1	0	1
0	1	1	1
0	1	0	1

$0 \times 8 + 1 \times 4 + 0 \times 2 + 1 \times 1 = 5$

a & b = 5

Note:
Bitwise OR table
Hint: addition

a	b	a|b
0	1	1
1	0	1
1	1	1
0	0	0

Bitwise AND table
Hint: multiplication

a	b	a&b
0	1	0
1	0	0
1	1	1
0	0	0

XOR table

a	b	a^ b
0	1	1
1	0	1
1	1	0
0	0	0

What will be the output of the following C program:

```
#include <stdio.h>
main()
{
  int a, b, c;
  a=2;
  b=3;
c= ~ (a ^ b);
printf("value of c = %d", c);
}
```

Answer: ?

Note:
~ denote bit wise negation operator
Negation truth table

a	b	a^ b	~ (a ^ b)
0	1	1	0
0	0	1	0
1	1	0	1
0	1	0	1

Negation of a ^ b means inverse of a ^ b i.e., if the value of a ^ b is 1 then the negation of a ^ b (i.e., ~ (a^ b)) is 0 and if the value of a ^ b is 0 then the negation of a ^ b (i.e., ~ (a^ b)) is 1.

What is the mistake in the following programs:

```
#include<iostream>
using namespace std;
main()
{
int a, b, c;
a=8;
b=7;
c = a || b;
cout<< " the value of c =    " << c;
}
```

```
#include <iostream>
using namespace std;
int main()
{
    cout << "\n Hello World" << endl;
    cout << "\n Hello World" << endl;
    cout << "\n Hello World" << endl;
```

```
    return 0;
}
```

Note: there is no mistake in the second program-- the output of the second program is:

Hello World

Hello World

Hello World

Program 7.7

C++ program to print the output:
Name of the book = B
Price of the book = 135.00
Number of pages = 300
Edition = 8
using structures

```
#include<iostream>
using namespace std;
main()
{
struct book {
char name;
float price;
int pages;
int edition;
};
struct book b1= {'B', 135.00, 300, 8};
cout<<"Name of the book =    "<< b1.name<< endl;
cout<<"Price of the book = "<< b1.price<<endl;
cout<<"Number of pages = "<< b1.pages<<endl;
cout<<"Edition of the book = "<< b1.edition<< endl;
}
```

The output on the screen:
Name of the book = B
Price of the book = 135.00
Number of pages = 300
Edition of the book = 8

The statement

```
struct book {
char name;
float price;
int pages;
int edition;
};
```

imply the structure definition i.e., we are defining a structure (and the data type name of the structure is book) and it consists of elements: name (which is of data type char), price (which is of data type float), pages (which is of data type int), edition (which is of data type int) – which are placed within the body of the structure.

The statement

```
struct book b1;
```

imply the structure variable declaration (where b1 denote the structure variable)

Why structure variable b1 is declared or defined?

In order to assign the values to the elements within the body of the structure, each element must be linked with structure variable with dot operator or period operator or member accessibility operator. For example: name is the element which must be linked with structure variable b1 with dot operator to assign a value B to the element "name".

The cout statement
cout<<"Name of the book = "<< b1.name<< endl;
make provision to print the output on the screen:
Name of the book = B
on the screen.

Similarly,
The statement
cout<<"Price of the book = "<< b1.price<<endl;
make provision to print the output:
Price of the book = 135.00
on the screen.

The statement
cout<<"Number of pages = "<< b1.pages<<endl;
make provision to print the output:
Number of pages = 300
on the screen.

The statement
cout<<"Edition of the book = "<< b1.edition<< endl;
make provision to print the output:
Edition of the book = 8
on the screen.

What will be output of the following program?

```
#include<iostream>
using namespace std;
struct book {
char name;
float price;
int pages;
int edition;
};
main()
{
struct book b1;
b1.name = 'C';
b1.price = 135.00;
b1.pages = 300;
b1.edition = 8;
cout<<"Name of the book = bulgarian " <<  b1.name << endl;
cout<<"\n Price of the book = " << b1.price;
cout<<"\n Number of pages = " << b1.pages<<endl;
cout<< "\n Edition of the book = "<< b1.edition;
}
```

What will be the output of the following program:

```
#include <iostream>

using namespace std;
main()
{
      int F, m, a;
      m=2;
      a=3;
      F=m*a;
    cout << "force applied to the mass = " << F << "\t Newton" << endl;
 }
```

Answer:
force applied to the mass = 6 Newton

Examine the following program and write the output:

```
#include <iostream>
#include<cmath>
using namespace std;
main()
{
cout<<" \n E=mc squared Einstein's famous equation that gave birth to the atom bomb and heralded a new world of atomic physics"<<endl;
cout<<" \n                                                                                                                    "<<endl;
cout<<" \n E = energy m = mass c = speed of light in vacuum"<<endl;
cout<<"\n As we know c squared is huge so if you convert a small amount of mass you'll get a tremendous amount of energy" <<endl;
cout<<" \n For example if you convert 1kg of mass you'll get energy of "<<endl;
long int E, m, c;
m=1;
c=300000000;
E=m*c*c;
cout<<"\n "<< E << " joules "<<endl;
cout<<"\n address of Energy E in the computer memory =    " <<&E <<endl;
cout<<"\n address of Energy m in the computer memory =    " <<&m <<endl;
cout<<"\n address of Energy c in the computer memory =    " <<&c <<endl;
int b, z, a;
b = sizeof(E);
z = sizeof(m);
a = sizeof(z);
cout<<"\n space occupied by E in the computer memory = "<< b <<"bytes"<< endl;
cout<<"\n space occupied by m in the computer memory = "<< z <<"bytes"<< endl;
cout<<"\n space occupied by c in the computer memory = "<< a <<"bytes"<< endl;
cout<<" \n Suppose c would have been 3×10 to the power of -8 meter per second then For 1 kg of mass E = 9 ×10 to the power of -16 joules"<< endl;
cout<<" \n hence thousands and thousands of hydrogen atoms in the sun would have to burn up to release 4 × 10 to the power of 26 joules of energy per second in the form of radiation"<< endl;
int i;
for(i=0; i<5;i++)
cout<<" \n Therefore sun would have ceased "<< endl;
cout<<"to form black hole even before an ooze of organic molecules would react and built earliest cells and then advance to a wide variety of one celled organisms and evolve through a highly sophisticated form of life to primitive mammals"<<endl;
long int v;
```

```
v=300000000;
c=300000000;
if (v==c)
cout<<" \n rest mass of the photon is zero because light travels at the speed of light"<<endl;
else
cout<<" \n Albert Einstein's special theory of relativity has to be rewritten"<< endl;
int E1;
E1=2;
if (isalpha(E1))
cout<<"\n Einstein's equation does not hold good"<< endl;
else
cout<<"\n Einstein's equation holds good because light has zero rest mass"<<endl;
cout<<"\n masses of the individual substances are 16 \t 18 \t 19 \t 20\t 21\t kilograms" << endl;
int j, sum = 0;
int num [5] = {16, 18, 19, 20, 21};
for(j=0; j<5; j++)
sum = sum + num [j];
cout<<"\n sum of the    masses of the individual substances =    " << sum<< endl;
cout<<"\n sum of the    energies of the individual substances =    " << sum * 300000000 * 300000000 << endl;
cout<< "\n average energy = " << sum / 5<< endl;
cout<<"\n                                                                                                    "<< endl;
cout<<"\n                                                                                                    "<< endl;
int p, q, r;
p= 6;
q= 3;
cout<< "\n expected energy = " << p << "multiplied by ten to the power of minus 16 joules calculated using Einstein
equation"<< endl;
cout<< "\n experimental energy = " << q << "multiplied by ten to the power of minus 16 joules calculated using
Einstein equation"<< endl;
r= q-p;
cout<< "\n difference between experimental energy and expected energy ="<< r << " multiplied by ten to the power
of minus 16 joules"<< endl;
cout<<"\n  absolute  value  of  the  difference  between  experimental  energy  and  expected  energy
="<<abs(r)<<"multiplied by ten to the power of minus 16 joules" << endl;
char k;
char einstein [10] ={'e', 'm', 'c', 's', 'q', 'u', 'a', 'r', 'e', 'd'};
for(k=0; k<10; k++)
cout<<"\n Einstein["<<einstein [k]<<"] = " <<einstein [k] << endl;
int s;
cout<<"\n Hey! Einstein may not be wrong please repeat the experiment"<<endl;
for(s=1; s<3; s++)
cout<<"\n                                        *              ";
cout<<"\n                                     *****             ";
cout<<"\n                          **Albert Einstein**          ";
cout<<"\n                            **e= mc squared**          ";
cout<<"\n                                 *****";
double EP, h, f;
f=2;
h=pow((6.625), -34);
EP= h*f;
cout<<"\n energy    calculated using the Planck equation = " << EP << endl;
if (E== EP)
cout<<"\n hf cannot be equivalent to mc squared"<< endl;
if (E>EP)
cout<<"\n hf can be equivalent to mc squared"<< endl;
if (E<EP)
```

```
cout<<"\n Einstein and Planck equation cannot be equalized"<<endl;
cout<<"\n                                                                    "<<endl;
cout<<"\n                                                                "<<endl;
cout<<"\n for more details please refer the book          "<<endl;
struct book {
char name;
float price;
int pages;
int edition;
};
struct book b1;
b1.name = 'E';
b1.price = 135.00;
b1.pages = 300;
b1.edition = 8;
cout<<"\n Name of the book = "<<b1.name<<"insteinian physics" << endl;
cout<<"\n Price of the book = " << b1.price << endl;
cout<< "\n Number of pages =    " << b1.pages<< endl;
cout<<"\n Edition of the book = "<<b1.edition<<" th edition"<< endl;
cout<<"*********************************************\n" ;
cout<<"*********************************************\n";
cout<<" According to the Albert Einstein's law of variation of mass with velocity: \n";
cout<<" M = m0 / sqrt ((1- (u/c) squared) \n ";
cout<<" M = mass of the moving body \t    m0 = rest mass of the body \t u= velocity of the body \t c= speed of light
in vacuum\n ";
{
double M, m0;
long int u, c, b;
m0 = 0.999;
u = 200000000;
c = 300000000;
b = u/c * u /c;
M = m0 / sqrt (1 - b);
cout<<" \n Mass of the moving body =    "<< M<<endl;
if (M==m0 || M<m0)
cout<<" \n body is at rest or the body is moving with nonrelativistic speed"<<endl;
else
cout<<" \n body is moving with relativistic speed"<<endl;
}
}
```

IX

Java

Java is a high level programming language conceived by James Gosling, Patrick Naughton, Chris Warth, Ed Frank, and Mike Sheridan at Sun Microsystems, Inc. in 1991 to create programs to control consumer electronics (which is now a subsidiary of Oracle Corporation) and released in 1995, used in internet programming, mobile devices, games, e-business solutions etc., because of its reliability, high performance, simplicity and easy to use and quick to learn and rigid versus extensibility.

Process of Java program execution: A Java program:

```
public class HelloWorld {
public static void main(String [] args) {
System.out.println("Hello, World!");
}
```

} is written in notepad or text pad or java editor is called the source program (Unlike C & C++ language, java is a platform independent language because java program can be written in notepad or text pad or java editor and executed) and this source program is saved as HelloWorld .java (because the class name is public class HelloWorld {} the source file should be named as HelloWorld. java) and sent to the java compiler (i.e., javac compiler) where the source program is compiled (i.e., the program is entirely read and translated into Java byte codes (but not into machine language)). If the javac compiler finds any error during compilation, it provides information about the error to the user. The programmer has to review code and check for the solution. And if there are no errors the translated program (i.e., java byte codes -- a highly optimized set of instructions) is stored in computers main memory as HelloWorld.class and since the java byte codes cannot be trusted to be correct—therefore before execution they are verified and converted to machine level language i.e., machine code sequence of 0's and 1's by Java run-time system, which is called the Java Virtual Machine (JVM) and is executed by a Java interpreter and
Hello, World!
is displayed on the screen.
public class HelloWorld imply class and
{

} imply the body of the class (Here: the curly brace '{' imply the beginning of the class and the curly brace '}' imply the end of the class) within which the main method
public static void main(String [] args)
{
} is written.
public static void main(String [] args) imply main method (a collection of statements or methods like System.out.println( ) that are grouped together to perform an operation) and
{

} imply the body of the main method (Here: the curly brace '{' imply the beginning of the main method and the curly brace '}' imply the end of the main method) within which the program statements i.e.,
System.out.println("Hello, World!");
is written and executed (i.e., main method in java functions like main function main () in C and C++).
If the statement public class HelloWorld is replaced by the statement public class sample i.e., if the above program is rewritten as:

```
public class sample {
public static void main(String [] args) {
System.out.println("Hello, World!");
}
}
```

Then the error will be displayed on the screen because the program written in notepad or text pad or java editor is saved as HelloWorld.java not as sample.java. If we want to write the statement public class sample instead of the statement public class HelloWorld, then we have to save the program written in notepad or in java editor as sample.java or but not as HelloWorld.java. As said earlier, like C & C++, Java is not platform dependent because java program can be written in notepad/ text pad or java editor and can be executed (whereas C & C++ program can only be written in C & C++ editor and can be executed). And like C & C++, Java is also a case sensitive language i.e., capital letters (or upper case letters) must be avoided to prevent the display of error on the screen (For example: If the statement PUBLIC static void main(String [] args) is written instead of the statement public static void main(String [] args) or the statement Public class HelloWorld is written instead of public class HelloWorld, compilation Error will be displayed on the screen). And if we forget to end each program statement within the body of main method with a semicolon (;), Error will be displayed on the screen i.e., if we forget to end the statement System.out.println("Hello, World!") with a semicolon (;), Error will be displayed on the screen. After the compilation of the source program, the translated (or the complied) program is stored in the computer's memory as object file and the program is executed. The program begins its execution with the method public static void main (String [] args) (which is called the main method -- the entry point of the program execution). If a program begins its execution with main method, it takes the control of the computer from the operating system. And after the complete execution of the program, the execution is terminated and the main method returns back the control to the operating system.

Semicolon: program is a set of instructions and each instruction (or each statement) is ended by a semicolon. Like in an English paragraph each sentence is ended by a full stop which tells that one sentence ends and another begins, semicolon implies that one instruction (or statement) ends and another begins.
System.out.println (" ") → method which displays the statement enclosed by the double quotation marks with the parenthesis of System.out.println i.e., displays the output:
Hello, World!
on the screen.
Note: if “ ” is used instead of " ", Error will be displayed on the screen
The statement System.out.println("Hello, World!"); will not display any error on the screen.
The statement System.out.println(“Hello, World!”); will display error on the screen.
Even though the statement System.out.print("Hello, World!"); is written instead of the statement System.out.println("Hello, World!"); i.e., instead of println only print is used – no compilation error will be displayed on the screen i.e., Hello, World! will be outputted on the screen without display of any compilation error on the screen.
If the word args in the statement public static void main(String [] args) is replaced by another word say jamesgosling or java
i.e., the above program is rewritten as:

```
public class HelloWorld
{
public static void main(String [] jamesgosling)
{
System.out.println("Hello, World!");
}
}
```

or

```
public class HelloWorld {
public static void main (String [] java) {
System.out.println("Hello, World!");
}
}
```

No compilation error will be displayed on the screen i.e., Hello, World! will be outputted on the screen without display of any error on the screen.
If the statement
public static void main(String [] args)
or
public static void main(String [] jamesgosling)
or
public static void main(String [] java)
is replaced by the statement public static void main(String []) -- Then the error is displayed on the screen. Because no word is written after String [] – any word say args or argv or java or jamesgosling should be written after string[] to prevent the display of error on the screen.
Note: Most Java programmers prefer args and argv i.e., the statements
public static void main(String [] args) and public static void main(String [] argv) are preferred.
If the statement public static void main (String [] jamesgosling) is replaced by the statement public static void main (String [] james gosling) i.e., space is left between the words james and gosling. Then the compilation error will be displayed on the screen.
jamesgosling → no error.
james gosling → error.
Similarly, if the space is left between the words Hello and World i.e., if the statement public class Hello World is written instead of the statement public class HelloWorld. Then the compilation error will be displayed on the screen.

Note: All the programs written in java editor is saved as HelloWorld.java and executed – hence public class HelloWorld corresponds to all programs.

Program 7.8

Java program to print the word “hello Bill Gates” on screen

```
public class HelloWorld {
public static void main (String [] args) {
System.out.println("hello Bill Gates");
}
}
```

The output on the screen:
hello Bill Gates

Program 7.9

Java program to print the word “****hello silicon city****” on screen

```
public class HelloWorld {
public static void main(String [] args) {
System.out.println(" ****hello silicon city**** ");
}
}
```

The output on the screen:
****hello silicon city****
Program 8.0

Java program to print
*
*****
*****
*****
*****
on screen

```
public class HelloWorld {
public static void main(String [] args) {
System.out.println("\n          *          ");
System.out.println("\n    ***** ");
System.out.println("\n    ***** ");
System.out.println("\n    ***** ");
System.out.println("\n    ***** ");
}
}
```

The output on the screen:
*
*****
*****
*****
*****
If new line \n is not included in the above program then the output on the screen is:
****

Write a program to print the following outputs:

(c)

*
****
java

```
****
*
```

(b)

```
***************
* *
* Hello World! *
* *
***************
```

(c)

Braces come in pairs!
Comments come in pairs!
All statements end with a semicolon!
Spaces are optional!
Must have a main method!
java is done mostly in lowercase. Like C & C++ it's also a case-sensitive language

Answers:

```
public class HelloWorld {
public static void main (String [] args) {
System.out.println("\n               *          ");
System.out.println("\n           ****      ");
System.out.println("\n      ******* ");
System.out.println("\n           ****        ");
System.out.println("\n               *           ");
}
}
```

```
public class HelloWorld {
public static void main (String [] args) {
System.out.println("\n                 ****************              ");
System.out.println("\n                                  * *                                  ");
System.out.println("\n                 * Hello World! *                ");
System.out.println("\n                                  * *                                  ");
System.out.println("\n               ***************             ");
}
}
```

```
public class HelloWorld {
public static void main (String [] args) {
System.out.println("\n Braces come in pairs!");
System.out.println("\n Comments come in pairs!");
System.out.println("\n    All statements end with a semicolon!");
System.out.println("\n    Spaces are optional!");
System.out.println("\n Must have a main method!");
System.out.println("\n java is done mostly in lowercase. Like C & C++ it's also a case-sensitive language");
}
}
```

Program 8.1

Java program to find the area of the circle

```
public class HelloWorld {
public static void main (String [] args) {
int r, area;
r = 2;
area = 4 * 3.14 * r * r;
System.out.println("The area of the circle = " + area);
}
}
```

The output on the screen:
The area of the circle = 50
int means the integer data type.
Note: An integer is a whole number — no fractions, decimal parts, or funny stuff.
The statement
int r, area; imply that we are creating the integer variables r , area.
The statements
r = 2;
area = 4 * 3.14 * r * r;
imply that we are assigning the values to the created variables (i.e., we are assigning the value 2 for r and 4 * 3.14 * r * r for area).
Comma in the statement int r, area; imply variable separator.
If multiplication sign × is used instead of multiplication operator * i.e.,
area = 4 × 3.14 × r × r;
then the compilation error is displayed on the screen.
In C language, the statement
printf("The area of the circle = %d ", area);
make the provision to print the output on the screen.
In C++ language, the statement
cout<< "The area of the circle = " << area;
make the provision to print the output on the screen.
whereas in the Java language, the statement
System.out.println("The area of the circle = " + area);
make the provision to print the output on the screen.
In the statement System.out.println("The area of the circle = " + area);
There are two strings:
1. The area of the circle =
2. area
plus operator (+) functions as the concatenation operator (concatenation means connecting two statements to produce a single statement) – which (here) concatenates the string “The area of the circle = ” and the string “area (which is 4 * 3.14 * r * r (=50 since r = 2))” -- producing a String statement
The area of the circle = 50 which is displayed on the screen as the result.

If the statement System.out.println("The area of the circle = " area); is written instead of the statement System.out.println("The area of the circle = " + area); i.e., plus sign is omitted. Then the compilation error will be displayed on the screen.
(Like in C and C++)
If the multiplication sign × is used instead of multiplication operator * i.e.,
The statement area = 4 × 3.14 × r × r; is written instead of area = 4 * 3.14 * r * r
then the compilation error will be displayed on the screen.
The area of the circle is = 50. 24 (for r = 2) but The area of the circle = 50 is displayed on the screen because the data type int is used instead of data type float.
If the data type float is used instead of int i.e., the above program is rewritten as:

```
public class HelloWorld{
public static void main(String [] args) {
float r;
r = 2;
area = 4 * 3.14 * r * r;
```

```
System.out.println("The area of the circle = " + area);
}
}
```

Then the output on the screen is:
The area of the circle = 50.24
If you write 4 * 3.14 * r ^ 2; instead of 4 * 3.14 * r * r; (where r ^ 2 → r to the power of 2 or r square), then error is displayed on the screen because unlike other high level languages – there is no operator for performing exponentiation operation i.e., (like in C and C++) there is no operator for performing r ^ 2 operation so the statement 4 * 3.14 * r ^ 2; is invalid.
Even though if we write ARGS instead of args i.e., even though if we express args in capital letter, No error is displayed on the screen.
public static void main(String [] ARGS) → no error is displayed on the screen.

Note:
In
public class HelloWorld
HelloWorld is the name of the file within the source program is saved. public class HelloWorld because the source program is saved in the file named HelloWorld.java.

Program 8.1

Java program to find the circumference of the circle

```
public class HelloWorld {
public static void main (String [] args) {
float r, circumference;
r = 2;
circumference = 3.14 * r * r;
System.out.println("The circumference of the circle = " + circumference);
}
}
```

The output on the screen is:
The circumference of the circle = 12.56
What will be the output of the following programs:

```
public class HelloWorld {
public static void main (String [] args) {
int l, b, area;
l=2;
b=2.5;
area = l*b;
System.out.println("The area of the rectangle = " + area);
}
}
```

```
public class HelloWorld {
public static void main (String [] args) {
int a, b, c;
a= 3;
b=3;
c=3;
if ((a + b< c) || (b + c < a) || (a==b && b==c))
System.out.println(" the triangle is equilateral");
else
System.out.println(" the triangle is not possible");
}
```

```
}
```

What is the mistake in the following program:

```
public class HelloWorld {
public static void main (String [] argv) {
float l, b, area, volume;
l=2;
b=2.5;
h =2.9
area = l*b;
volume = l*b*h;
System.out.println("The volume of the rectangle = " + area);
}
```

“Shut up and code.”
ANONYMOUS - NOTICE ON THE OFFICE WALL OF AN INDUSTRIAL SOFTWARE MANAGER, 1970

Program 8.2

Java program to convert the temperature in Celsius to Fahrenheit

```
public class HelloWorld{
public static void main(String [] args){
float C, F;
C=38.5;
F = 9*C/5 +32;
System.out.println("temperature in Fahrenheit= " +F);
}
}
```

The output on the screen:
temperature in Fahrenheit= 101.3
Note:

Program I:

```
public class HelloWorld
{
public static void main(String [] args)
{
int a, b, sum;
a=1;
b=2;
sum = a + b;
System.out.println("the sum of a and b = " + sum);
}
}
```

The output on the screen:
the sum of a and b = 3

If you want to supply the values for a and b through the key board, then we have to include the statements:
import java.util.Scanner;
Scanner scan = new Scanner(System.in);
and replace the statements
a=1;
b=2;

by the statements
System.out.print("Enter any two Numbers: ");
a = scan.nextInt();
b = scan.nextInt();
i.e., the program should be rewritten as:

```
import java.util.Scanner;
public class HelloWorld
{
public static void main(String [] args) {
int a, b, sum;
Scanner scan = new Scanner(System.in);
System.out.print("Enter any two Numbers: ");
a = scan.nextInt();
b = scan.nextInt();
sum = a + b;
System.out.println("the sum of a and b = " + sum);
}
}
```

The output on the screen:
Enter any two Numbers:
If you enter two numbers 2 and 3
the sum of a and b = 5 will be outputted on the screen

Program II:

```
public class HelloWorld
{
public static void main(String [] args) {
float x;
x = 233;
System.out.println(" square root of a number = " + Math.sqrt(x));
  }
}
```

The output on the screen:
square root of a number = 15.264

If you want to supply the value for x through the key board, then we have to include the statements:
import java.util.Scanner;
Scanner scan = new Scanner(System.in);
and replace the statement
x = 233;
by the statements
System.out.print("Enter any Number: ");
x = scan.nextFloat();
i.e., the program should be rewritten as:

```
import java.util.Scanner;
public class HelloWorld {
public static void main(String [] args) {
int x;
Scanner scan = new Scanner(System.in);
System.out.print("Enter any Number: ");
x = scan.nextFloat();
System.out.println(" square root of a number = " + Math.sqrt(x));
}
```

```
}
```
The output on the screen:
Enter any Number:
If you enter the number 233
square root of a number = 15.264337522 will be outputted on the screen.

Program III:

```
public class HelloWorld
{
public static void main(String [] args) {
double x;
x = 233;
System.out.println(" square root of a number = " + Math.sqrt(x));
  }
}
```
The output on the screen:
square root of a number = 15.264337522473747

If you want to supply the value for x through the key board, then we have to include the statements:
import java.util.Scanner;
Scanner scan = new Scanner(System.in);
and replace the statement
x = 233;
by the statements
System.out.print("Enter any Number: ");
x = scan.nextDouble();
i.e., the program should be rewritten as:

```
import java.util.Scanner;
public class HelloWorld {
public static void main(String [] args) {
double x;
Scanner scan = new Scanner(System.in);
System.out.print("Enter any Number: ");
x = scan.nextDouble();
System.out.println(" square root of a number = " + Math.sqrt(x));
}
}
```
The output on the screen:
Enter any Number:
If you enter the number 233
square root of a number = 15.264337522473747 will be outputted on the screen.

Program IV:

```
public class HelloWorld{
public static void main(String[] args) {
char c;
c = 'A';
System.out.println("ch= " + c);
}
}
```
The output on the screen:
ch= A

If you want to supply the value for c through the key board, then we have to include the statements:
import java.util.Scanner;
Scanner scan = new Scanner(System.in);
and replace the statement
c = 'A';
by the statements
System.out.print("Enter a character:");
c = (char)System.in.read();
i.e., the program should be rewritten as:

```
public class HelloWorld {
public static void main(String[] args) throws Exception {
  char c;
  System.out.print("Enter a character:");
  c = (char)System.in.read();
  System.out.println("ch= " + c);
  }
}
```

The output on the screen:
Enter a character:
If you enter the character K
ch= K will be outputted on the screen.

Note: Exception is a problem that arises during the execution of a program. When an exception occurs, program abnormally terminates and disrupts—throws Exception should be written after the statement public static void main(String[] args) so that the exceptions are thrown to the operating system to handle and the program will be successfully executed and the output will be displayed on the screen.

Program V:

```
import java.util.Scanner;
public class HelloWorld {
public static void main(String [] args) {
String m;
Scanner in = new Scanner(System.in);
System.out.print("Enter the word: ");
m = in.nextLine();
System.out.println(" the word you entered = " + m);
}
}
```

The output on the screen:
Enter the word:
If you enter the word dog
the word you entered = dog will be outputted on the screen.

Note:
If the statement
m = scan.nextLine(); is written instead of
m = in.nextLine();
Then we have to replace the statement
Scanner in = new Scanner(System.in);
by the statement
Scanner scan = new Scanner(System.in);
Otherwise compilation error will be displayed on the screen.

What is the mistake in the following program:

```
public class HelloWorld
{
static public void main(String args []) {
float x;
x = 233;
System.out.println(" cube root of a number = " + Math.cbrt(x));
  }
}
```

Answer:

There is no mistake in the above program. The statement public static void main(String[] args) can also be written as static public void main(String args [])
The output on the screen is:
cube root of a number = 6.1534494936636825

Program 8.3

Java program to find the product of two numbers.

```
public class HelloWorld{
public static void main(String [] args) {
int a, b, product;
a=1;
b=2;
product = a * b;
System.out.println("the product of a and b = " + product);
}
}
```

The output on the screen:
the sum of a and b = 2

If you want to supply the values for a and b through the key board, then we have to include the statements:
import java.util.Scanner;
Scanner scan = new Scanner(System.in);
and replace the statements
a=1;
b=2;
by the statements
System.out.print("Enter any two Numbers: ");
a = scan.nextInt();
b = scan.nextInt();
i.e., the program should be rewritten as:

```
import java.util.Scanner;
public class HelloWorld{
public static void main(String [] args) {
int a, b, product;
Scanner scan = new Scanner(System.in);
System.out.print("Enter any two Numbers: ");
a = scan.nextInt();
b = scan.nextInt();
product = a * b;
System.out.println("the product of a and b = " + product);
}
```

```
}
```
The output on the screen:
Enter any two Numbers:
If you enter two numbers 6 and 3
the product of a and b = 18 will be outputted on the screen

If you want to assign the floating point values for a & b, then the statement int a, b, sum; should be replaced by the statement float a, b, sum;
and the statements
```
a = scan.nextInt();
b = scan.nextInt();
```
should be replaced by the statements
```
a = scan.nextFloat();
b = scan.nextFloat();
```

i.e., the above program should be rewritten as:

```
import java.util.Scanner;
public class HelloWorld {
public static void main(String [] args) {
float a, b, product;
Scanner scan = new Scanner(System.in);
System.out.print("Enter any two Numbers: ");
a = scan.nextFloat();
b = scan.nextFloat();
product = a * b;
System.out.println("the product of a and b = " + product);
}
}
```
The output on the screen:
Enter any two Numbers:
If you enter two floating point values 2.9 and 3.6
the product of a and b = 10.44 will be outputted on the screen.

float is used instead of int because a and b are assigned fractional values (i.e., 2.9 and 3.6) if int is used instead of float then the result will not be clearly outputted i.e., instead of 10.44 the computer displays only 10 (as said earlier).

If the statement System.out.println("the product of a and b = " + product); is replaced by the statement
```
System.out.println(a + "* " + b + " = " + product);
```
Then the output on the screen is:
2.9 * 3.6 = 10.44

Note: The word public in public class HelloWorld implies that the program or the data within the program (such as methods, variables etc.) can be accessed directly by an external java program.
If replace the word public by private i.e., private class HelloWorld is written instead of public class HelloWorld -- then the program or the data within the program (such as methods, variables etc.) cannot be accessed directly by an external program.

If you insert a value 2^3 for a and 3^2 for b, then as said earlier wrong result or compilation error will be flagged on the screen.
```
a=2^3
b=3^2; → ERROR
a=2* 2*2
b=3*3; → Result will be outputted on the screen i.e.,
```
the product of a and b = 72
If you want to insert a 10 digit number for a and b i.e.,

a=1000000000
b=3000000000, then the statement
int a, b, product; should be replaced by the statement long int a, b, product;
i.e.,

```
public class HelloWorld{
public static void main(String [] args){
long int a, b, product;
a=1000000000;
b=2000000000;
product = a * b;
System.out.println("the product of a and b = " + product);
}
}
```

The output on the screen:
the product of a and b = 3000000000000000000

“Code doesn't exist until it's checked into source control.”

:JEFF ATWOOD

Did you know that

Harvard MBA candidate Dan Bricklin and programmer Bob Frankston developed VisiCalc, the program that turned the personal computer into a business machine.

What will be the output of the following program:

```
public class HelloWorld{
static public void main(String args []) {
float x;
x = 2;
System.out.println(" square of a number = " + Math.pow((x), 2));
}
}
```

Answer:
square of a number = 4

Program 8.4

Java program to find the square of a number

```
public class HelloWorld{
public static void main(String [] args){
int a, b;
a=2;
b = a * a;
System.out.println("the square of a = " + b);
}
}
```

The output on the screen:
the square of a = 4

If you want to supply the value for a through the key board, then we have to include the statements:

```
import java.util.Scanner;
Scanner scan = new Scanner(System.in);
```

and replace the statement
a=2;
by the statements
System.out.print("Enter any Number: ");
a = scan.nextInt();

i.e., the program should be rewritten as:

```
import java.util.Scanner;
public class HelloWorld{
public static void main(String [] args) {
int a, b;
Scanner scan = new Scanner(System.in);
System.out.println("Enter any Number: ");
a = scan.nextInt();
b = a * a;
System.out.println("the square of a = " + b);
}
}
```

The output on the screen:
Enter any number:
If you enter a number 3
the square of a = 9 will be outputted on the screen.

Note:

If scan.nextint() is written instead of scan.nextInt()
public static void main(string [] args); is written instead of public static void main(String [] args)
System.out.println(the square of a = + b); is written instead of System.out.println("the square of a = " + b);
Then the compilation error will be displayed on the screen.

Program 8.5

Java program to find the greatest of two numbers using
(a) if - if statement
(b) if - else statement

The syntax of if - if statement is:

```
if (this condition is true)
{
print this statement using the method System.out.println( );
}
if (this condition is true)
{
print this statement using the method System.out.println( );
}
```

(a)

```
public class HelloWorld {
public static void main(String [] args){
int a, b;
a=2;
b =3;
if(a>b)
```

```
{
System.out.println("a is greater than b");
}
if(b>a)
{
System.out.println("b is greater than a");
}
}
}
```

The output on the screen:
b is greater than a

(a>b) and (b>a) are the conditions and if the condition (a> b) is true, then the statement

```
{
System.out.println("a is greater than b");
}
```

make provision to print the output:
a is greater than b
and if the condition (a> b) is not obeyed and the condition (b>a) is true, then the statement

```
{
System.out.println("b is greater than a");
}
```

make provision to print the output:
b is greater than a

(b)
The syntax of if – else statement is:

```
if (this condition is true)
{
print this statement using the method System.out.println( );
}
else
{
print this statement using the method System.out.println( );
}
```

```
public class HelloWorld{
public static void main(String [] args){
int a, b;
a=2;
b =3;
if(a>b)
{
System.out.println("a is greater than b");
}
else
{
System.out.println("b is greater than a");
}
}
}
```

The output on the screen:
b is greater than a
In the above program:
if the condition (a> b) is true, then the statement

```
{
System.out.println("a is greater than b");
}
```
make provision to print the output:
a is greater than b
else
the statement
```
{
System.out.println("b is greater than a");
}
```
make provision to print the output:
b is greater than a

If you want to supply the values for a and b through the key board, then the above program should be rewritten as:

```
import java.util.Scanner;
public class HelloWorld{
public static void main(String [] args){
int a, b;
Scanner scan = new Scanner(System.in);
System.out.println("Enter any two Numbers: ");
a = scan.nextInt();
b = scan.nextInt();
if(a>b)
{
System.out.println("a is greater than b");
}
if(b>a)
{
System.out.println("b is greater than a");
}
}
}
```
The output on the screen:
Enter any two Numbers:
If you enter two numbers 2 and 3
b is greater than a will be outputted on the screen.

Note:
Even if the statements
System.out.println("a is greater than b");
System.out.println ("b is greater than a");
are not written within the braces {}
i.e.,
```
import java.util.Scanner;
public class HelloWorld{
public static void main(String [] args){
int a, b;
Scanner scan = new Scanner(System.in);
System.out.println("Enter any two Numbers: ");
a = scan.nextInt();
b = scan.nextInt();
if(a>b)
System.out.println("a is greater than b");
if(b>a)
System.out.println("b is greater than a");
```

```
}
}
```

There will no display of compilation error on the screen or there will be no change in the output displayed on the screen (i.e., b is greater than a will be outputted on the screen).

Program 8.6

Java program to find the greatest of three numbers using
(a) if - if - if statement
(b) if – else if – else statement
(b) if – else if – else if statement

The syntax of if – if- if statement is:

```
if (this condition is true)
{
print this statement using the method System.out.println( );
}
if (this condition is true)
{
print this statement using the method System.out.println( );
}
if (this condition is true)
{
print this statement using the method System.out.println( );
}
```

(a)

```
public class HelloWorld{
public static void main(String [] args){
int a, b, c;
a=2;
b =3;
c=4;
if(a>b&&a>c)
{
System.out.println("a is greater than b and c");
}
if(b>a&&b>c)
{
System.out.println("b is greater than a and c");
}
if(c>b&&c>a)
{
System.out.println("c is greater than b and a");
}
}
}
```

The output on the screen:
c is greater than b and a
symbol && imply and i.e., the condition
(a>b&&a>c) imply
a is greater than b and a is greater than c
(a>b&&a>c), (b>a&&b>c) and (c>b&&c>a) are the conditions and if the condition (a>b&&a>c) is true, then the statement

```
{
System.out.println("a is greater than b and c");
}
```
make provision to print the output:
a is greater than b and c
and if the condition(a>b&&a>c) is not true and the condition (b>a&&b>c) is true, then the statement
```
{
System.out.println("b is greater than a and c");
}
```
make provision to print the output:
b is greater than a and c

and if the condition (b>a&&b>c) is not true and the condition (c>b&&c>a) is true, then the statement
```
{
System.out.println("c is greater than b and a");
}
```
make provision to print the output:
c is greater than b and a

If you want to supply the values for a, b and c through the key board, then the above program should be rewritten as:

```
import java.util.Scanner;
public class HelloWorld{
public static void main(String [] args) {
int a, b, c;
Scanner scan = new Scanner(System.in);
System.out.println("Enter any three Numbers: ");
a = scan.nextInt();
b = scan.nextInt();
c = scan.nextInt();
if(a>b&&a>c)
{
System.out.println("a is greater than b and c");
}
if(b>a&&b>c)
{
System.out.println("b is greater than a and c");
}
if(c>b&&c>a)
{
System.out.println("c is greater than b and a");
}
}
}
```
The output on the screen:
Enter any three Numbers:
If you enter three numbers 2, 3 and 4
c is greater than b and a will be outputted on the screen.

The syntax of if – else if – else statement is:

```
if (this condition is true)
{
print this statement using the method System.out.println( );
}
```

```
else if (this condition is true)
{
print this statement using the method System.out.println( );
}
else
{
print this statement using the method System.out.println( );
}
```

(b)

```
public class HelloWorld{
public static void main(String [] args){
int a, b, c;
a=2;
b =3;
c=4;
if(a>b&&a>c)
{
System.out.println("a is greater than b and c");
}
else if(b>a&&b>c)
{
System.out.println("b is greater than a and c");
}
else
{
System.out.println("c is greater than b and a");
}
}
}
```

The output on the screen:
c is greater than b and a

The syntax of if – else if – else if statement is:

```
if (this condition is true)
{
print this statement using the method System.out.println( );
}
else if (this condition is true)
{
print this statement using the method System.out.println( );
}
else if (this condition is true)
{
print this statement using the method System.out.println( );
}
```

(c)

```
public class HelloWorld {
public static void main(String [] args){
int a, b, c;
a=2;
b =3;
```

```
c=4;
if(a>b&&a>c)
{
System.out.println("a is greater than b and c");
}
else if(b>a&&b>c)
{
System.out.println("b is greater than a and c");
}
else if(c>b&&c>a)
{
System.out.println("c is greater than b and a");
}
}
}
```

The output on the screen:
c is greater than b and a
Note:

If the statements

```
if(a>b&&a>c)
{
System.out.println("a is greater than b and c");
}
else if(b>a&&b>c)
{
System.out.println("b is greater than a and c");
}
else if(c>b&&c>a)
{
System.out.println("c is greater than b and a");
}
```

are replaced by the statements

```
if(a>b&&a>c)
{
System.out.println(a + "is greater than" + b    + "and" + c);
}
else if(b>a&&b>c)
{
System.out.println(b + "is greater than" + a    + "and" + c);
}
else if(c>b&&c>a)
{
System.out.println(c + "is greater than" + b    + "and" + a);
}
```

Then the output on the screen is:
4 is greater than 3 and 2

Program 8.7

Java program to find the average of 10 numbers

```
import java.util.Scanner;
public class HelloWorld{
```

```
public static void main(String [] args) {
int N1, N2, N3, N4, N5, N6, N7, N8, N9, N10, X;
Scanner scan = new Scanner(System.in);
System.out.println("Enter any ten Numbers: ");
N1 = scan.nextInt();
N2 = scan.nextInt();
N3 = scan.nextInt();
N4 = scan.nextInt();
N5 = scan.nextInt();
N6 = scan.nextInt();
N7 = scan.nextInt();
N8 = scan.nextInt();
N9 = scan.nextInt();
N10 = scan.nextInt();
X = (N1 + N2 + N3 + N4 + N5 + N6 + N7 + N8 + N9 + N10) /10;
System.out.println("the average of 10 numbers = " + X);
}
}
```

The output on the screen:
Enter any ten Numbers:
If you enter ten numbers 1, 2, 3, 4, 5, 6, 7, 8, 9 and 10
the average of 10 numbers = 5 will be outputted on the screen.

Note: The average of 10 numbers is 5.5, the output on the screen is 5 because int is used instead of float.

“Time is so short, you can't make a debug...”

: SCOTT ADAMS

Program 8.8

Java program to find the simple interest

```
public class HelloWorld{
public static void main(String [] args) {
int P,T, R, SI;
P = 1000;
T = 2;
R = 3;
SI = P*T*R/100;
System.out.println("the simple interest = " + SI);
}
}
```

The output on the screen:
the simple interest = 60

If you want to supply the values for P, T and R through the key board, then the above program should take the form:

```
import java.util.Scanner;
public class HelloWorld {
public static void main(String [] args) {
int P,T, R, SI;
Scanner scan = new Scanner(System.in);
System.out.println("Enter principal amount:");
P = scan.nextInt();
System.out.println("Enter time:");
```

```
T = scan.nextInt();
System.out.println("Enter rate of interest:");
R = scan.nextInt();
SI = P*T*R/100;
System.out.println("the simple interest = " + SI);
}
}
```

The output on the screen:
Enter principal amount:
If you enter the principal amount 1000
Enter time:
If you enter the time 2
Enter rate of interest:
If you enter the rate of interest 3
the simple interest = 60 will be outputted on the screen.

Program 8.9

Java program to find the senior citizen

```
public class HelloWorld{
public static void main(String [] args){
int age;
age=20;
if(age> = 60)
{
System.out.println("senior citizen");
}
if(age<60)
{
System.out.println("not a senior citizen");
}
}
}
```

The output on the screen:
not a senior citizen
(age> = 60) implies age greater than or equal to 60

If you want to supply the value for age through the key board, then the above program should be rewritten as:

```
import java.util.Scanner;
public class HelloWorld{
public static void main(String [] args){
int age;
Scanner scan = new Scanner(System.in);
System.out.println("Enter the age: ");
age = scan.nextInt();
if(age> = 60)
{
System.out.println("senior citizen");
}
if(age<60)
{
System.out.println("not a senior citizen");
}
}
```

```
}
```
The output on the screen:
Enter the age:
If you enter the age 60
senior citizen will be outputted on the screen.
Suppose if you enter the age 28
not a senior citizen will be outputted on the screen.

Note:
If the symbol >> is used instead of > and << is used instead of <
Then the compilation error will be displayed on the screen.

Program 9.0

Java program to get marks for 3 subjects and declare the result
If the marks >= 35 in all the subjects the student passes else fails.

```
public class HelloWorld{
public static void main(String [] args){
int M1, M2,M3;
M1 = 38;
M2= 45;
M3 = 67;
if(M1>= 35 && M2>= 35 && M3>= 35)
{
System.out.println("candidate is passed");
}
else
{
System.out.println("candidate is failed");
}
}
}
```
The output on the screen:
candidate is passed
(M_1>= 35 && M_2>= 35 && M_3>= 35) imply M_1 is greater than or equal to 35 and M_2 is greater than or equal to 35 and M_3 is greater than or equal to 35.
>= imply greater than or equal to.
&& imply and whereas & imply address.
(M_1>= 35 && M_2>= 35 && M_3>= 35) is the condition and if the condition (M_1>= 35 && M_2>= 35 && M_3>= 35) is true, then the statement
```
{
System.out.println("candidate is passed");
}
```

make provision to print the output:
candidate is passed
else the statement
```
{
System.out.println("candidate is failed");
}
```
make provision to print the output:
candidate is failed

If you want to supply the values for marks M_1, M_2 and M_3 through the key board, then the above program should be rewritten as:

```
import java.util.Scanner;
public class HelloWorld{
public static void main(String [] args) {
int age;
Scanner scan = new Scanner(System.in);
System.out.println("Enter any three Numbers: ");
M1= scan.nextInt();
M2 = scan.nextInt();
M3 = scan.nextInt();
if(M1>= 35 && M2>= 35 && M3>= 35)
{
System.out.println("candidate is passed");
}
else
{
System.out.println("candidate is failed");
}
}
}
```

The output on the screen:
Enter any three Numbers:
If you enter three numbers 26, 28, 39
candidate is failed will be outputted on the screen.

“One of my most productive days was throwing away 1000 lines of code.”

: KEN THOMPSON

Did you know that

In 1833, Charles Babbage developed the analytical engine. This machine consisted of five functional units such as input unit, memory unit, arithmetic unit, control unit and output unit. The architecture of the modern digital computer resembles the analytical engine and hence Charles Babbage is called the father of computers.

Program 9.1

Java program to find profit or loss

```
import java.util.Scanner;
public class HelloWorld{
public static void main(String [] args) {
int CP, SP, loss, profit;
Scanner scan = new Scanner(System.in);
System.out.println("Enter cost price: ");
CP = scan.nextInt();
System.out.println("Enter selling price: ");
SP = scan.nextInt();
if(SP>CP)
{
System.out.println("profit= " + (SP-CP));
}
if(CP>SP)
{
System.out.println("loss ="    +(CP-SP));
}
```

```
}
}
```
The output on the screen:
Enter cost price:
If you enter the cost price 25
Enter selling price:
If you enter the selling price 26
profit = 1 will be outputted on the screen.

If the condition (SP>CP) is true, then the statement
```
{
System.out.println("profit= " + (SP-CP));
}
```
make provision to print the output:
profit = (SP-CP) (in this case profit = 26-25 =1)

If the condition (CP>SP) is true, then the statement
```
{
System.out.println("loss ="    + (CP-SP));
}
```
make provision to print the output:
loss = (CP-SP)

Note: if the minus sign (–) is used instead of (-) i.e., (CP– SP) is written instead of (CP- SP), the error will be displayed on the screen (because like C & C ++, java is also a case sensitive language).

If the statement
System.out.println("profit= " + SP-CP);
is written instead of the statement
System.out.println("profit= " + (SP-CP));
i.e., SP-CP is written instead of (SP-CP)
Then the Compilation Error will be displayed on the screen.

What is the mistake in the following program:

```
import java.util.Scanner;
public class HelloWorld{
public static void main(String [] args){
float I, C;
Scanner scan = new Scanner(System.in);
System.out.println("Enter the length in inches: ");
I = scan.nextInt();
C = 2.54*I;
System.out.print("length in centimeters = " + C);
}
}
```

Program 9.2

Java program to find the incremented and decremented values of two numbers

```
public class HelloWorld{
public static void main(String [] args){
int a, b, c, d, e, f;
a = 10;
b=12;
```

```
c=a+1;
d=b+1;
e=a-1;
f=b-1;
System.out.print("the incremented value of a = "+ c);
System.out.print("the incremented value of b = "+ d);
System.out.print("the decremented value of a = "+ e);
System.out.print("the decremented value of b = "+ f);
}
}
```

The output on the screen:
the incremented value of a = 11 the incremented value of b = 13 the decremented value of a = 9 the decremented value of b = 11

If the statements

```
System.out.print("the incremented value of a = "+ c);
System.out.print("the incremented value of b = " + d);
System.out.print("the decremented value of a = " + e);
System.out.print("the decremented value of b = " + f);
```

are replaced by the statements

```
System.out.print("\n the incremented value of a = "    + c);
System.out.print("\n the incremented value of b = " + d);
System.out.print("\n the decremented value of a = " + e);
System.out.print("\n the decremented value of b = " + f);
```

i.e., if the above program is rewritten as:

```
public class HelloWorld{
public static void main(String [] args) {
int a, b, c, d, e, f;
a = 10;
b=12;
c=a+1;
d=b+1;
e=a-1;
f=b-1;
System.out.print("\n the incremented value of a = "    + c);
System.out.print("\n the incremented value of b = " + d);
System.out.print("\n the decremented value of a = " + e);
System.out.print("\n the decremented value of b = " + f);
}
}
```

Then the output on the screen is:
the incremented value of a = 11
the incremented value of b = 13
the decremented value of a = 9
the decremented value of b = 11

i.e., \n make provision for the another result to print in the new line. If you want to supply the values for a and b through the key board, then the above program should take the form:

```
import java.util.Scanner;
public class HelloWorld{
public static void main(String [] args){
int a, b, c, d, e, f;
Scanner scan = new Scanner(System.in);
System.out.println("Enter any Number: ");
```

```
a = scan.nextInt();
System.out.println("Enter any Number: ");
b = scan.nextInt();
c=a+1;
d=b+1;
e=a-1;
f=b-1;
System.out.print("\n the incremented value of a = "    + c);
System.out.print("\n the incremented value of b = " + d);
System.out.print("\n the decremented value of a = " + e);
System.out.print("\n the decremented value of b = " + f);
}
}
```

The output on the screen:
Enter any Number:
If you enter the value 2
Enter any Number:
If you enter the value 3
the incremented value of a = 3
the incremented value of b = 4
the decremented value of a = 1
the decremented value of b = 2
will be outputted on the screen.

Note: b++ is same as b + 1 and b-- is same as b - 1 but b++ or b-- should be used only in case of for loop or loop statements. Usage of b++ or b-- instead of b +1 or b-1 in the certain online compilers like coding ground (tutorials point) yields error or displays the wrong result.

What will be the output of the following programs:

```
import java.util.Scanner;
public class temperature{
public static void main(String [] args) {
float T1, T2, A;
Scanner scan = new Scanner(System.in);
System.out.println("Enter any Number: ");
T1 = scan.nextFloat();
System.out.println("Enter any Number: ");
T2 = scan.nextFloat();
A = (T1 + T2) / 2;
System.out.println("the average temperature of the day = " + A);
}
}
```

```
import java.util.Scanner;
public class HelloWorld{
public static void main(String [] args) {
int P;
Scanner scan = new Scanner(System.in);
System.out.println("Enter the percentage: ");
P = scan.nextInt();
if(P >= 60)
{
System.out.println("first class");
}
if(P>=50&&P <60)
```

```
{
System.out.println("second class");
}
if(P>=40&&P<=50 )
{
System.out.println("pass class");
}
if(P<40)
{
System.out.println("fail");
}
}
}
```

Program 9.3

Java program to calculate the discounted price and the total price after discount
Given:
If purchase value is greater than 1000, 10% discount
If purchase value is greater than 5000, 20% discount
If purchase value is greater than 10000, 30% discount

(a) discounted price

```
import java.util.Scanner;
public class HelloWorld{
public static void main(String [] args) {
int PV, dis;
Scanner scan = new Scanner(System.in);
System.out.println("Enter purchased value: ");
PV = scan.nextInt();
if(PV<1000)
{
System.out.println("dis = " + PV* 0.1);
}
if(PV>5000)
{
System.out.println("dis = " + PV* 0.2);
}
if(PV<10000)
{
System.out.println("dis= " + PV* 0.3);
}
}
}
```

The output on the screen:
Enter purchased value:
If you enter the purchased value 6500
dis = 1300 will be outputted on the screen.

If the condition (PV<1000) is true i.e., purchased value is less than 1000, then the statement

```
{
System.out.println("dis = " + PV* 0.1);
}
```

make provision to print the output:
dis= PV* 10% = PV* 10 /100 = PV* 0.1
If the condition (PV< 5000) is true i.e., purchased value is less than 5000, then the statement

```
{
System.out.println("dis = " + PV* 0.2);
}
```
make provision to print the output:
dis= PV* 20% = PV* 20 /100 = PV* 0.2
If the condition (PV< 10000) is true i.e., purchased value is less than 10000, then the statement
```
{
System.out.println("dis = " + PV* 0.3);
}
```
make provision to print the output:
dis= PV* 30% = PV* 30 /100 = PV* 0.3

(b) total price

```
import java.util.Scanner;
public class HelloWorld{
public static void main(String [] args) {
int PV, total;
Scanner scan = new Scanner(System.in);
System.out.println("Enter purchased value: ");
PV = scan.nextInt();
if(PV<1000)
{
System.out.println("total= " +    PV - PV* 0.1);
}
if(PV>5000)
{
System.out.println("total = " + PV- PV* 0.2);
}
if(PV<10000)
{
System.out.println("total= " + PV- PV* 0.3);
}
}
}
```
The output on the screen:
Enter purchased value:
If you enter the purchased value 650
total = 585 will be outputted on the screen.

If the condition (PV<1000) is true i.e., purchased value is less than 1000, then the statement
```
{
System.out.println("total = " +    PV - PV* 0.1);
}
```
make provision to print the output:
total =PV- dis = PV- PV*10% = PV- PV* 10 /100 = PV - PV * 0.1

If the condition (PV< 5000) is true i.e., purchased value is less than 5000, then the statement
```
{
System.out.println("total = " +    PV - PV* 0.2);
}
```
make provision to print the output:
total =PV- dis = PV- PV*20% = PV- PV* 20 /100 = PV - PV * 0.2

If the condition (PV< 10000) is true i.e., purchased value is less than 10000, then the statement
```
{
```

```
System.out.println("total = " +    PV - PV* 0.3);
}
```

make provision to print the output:
total =PV- dis = PV- PV*30% = PV- PV* 30 /100 = PV - PV * 0.3

Note: Combing both the programs (above), we can write:

```
import java.util.Scanner;
public class HelloWorld{
public static void main(String [] args){
int PV, dis, total;
Scanner scan = new Scanner(System.in);
System.out.println("Enter purchased value: ");
PV = scan.nextInt();
if(PV<1000)
{
System.out.println("dis = " + PV* 0.1);
System.out.println("total= " + total - dis);
}
if(PV>5000)
{
System.out.println("dis = " + PV* 0.2);
System.out.println("total= " + total - dis);
}
if(PV<10000)
{
System.out.println("dis = " + PV* 0.3);
System.out.println("total= " + total - dis);
}
}
}
```

The output on the screen:
Enter purchased value:
If you enter the purchased value 850
dis = 85
total = 765
will be outputted on the screen.

Program 9.4

Java program to print the first ten natural numbers using for loop statement

```
public class HelloWorld{
public static void main(String [] args){
int i;
for (i=1; i<=10; i++)
System.out.println("value of i = " + i);
}
}
```

The output on the screen is:
value of i = 1 value of i = 2 value of i= 3 value of i= 4 value of i= 5 value of i= 6 value of i = 7 value of i= 8 value of i = 9 value of i = 10

for (i=1; i<=10; i++) denote the for loop statement and the syntax of the for loop statement is:
for (initialization; condition; increment)
Here:

i=1 denote initialization (i.e., from where to start)
i<=10 denote the condition (i.e., stop when 10 is reached)
i++ imply increment (which tells the value of i to increase by 1 each time the loop is executed) and i++ is the same as i+1.
Since the initialization i.e., i=1
The statement System.out.println("value of i = " + i); make provision to print the output:
value of i = 1
on the screen.
After this, the following execution takes place:
value of i
i= 1
Is the condition (i<=10) is true?
Yes because i=1
Do this
i= 1+1 = 2
The statement System.out.println("value of i = " + i); make provision to print the output:
value of i = 2
Now, the value of i is:
i= 2
Is the condition (i<=10) is true?
Yes because i=2
Do this
i= 2+1 = 3
The statement System.out.println("value of i = " + i); make provision to print the output:
value of i = 3
Now, the value of i is:
i= 3
Is the condition (i<=10) is true?
Yes because i=3
Do this
i= 3+1 = 4
The statement System.out.println("value of i = " + i); make provision to print the output:
value of i = 4
Now, the value of i is:
i= 4
Is the condition (i<=10) is true?
Yes because i=4
Do this
i= 4+1 = 5
The statement System.out.println("value of i = " + i); make provision to print the output:
value of i = 5
Now, the value of i is:
i= 5
Is the condition (i<=10) is true?
Yes because i=5
Do this
i= 5+1 = 6
The statement System.out.println("value of i = " + i); make provision to print the output:
value of i = 6
Now, the value of i is:
i= 6
Is the condition (i<=10) is true?
Yes because i=6
Do this
i= 6+1 = 7
The statement System.out.println("value of i = " + i); make provision to print the output:

value of i = 7
Now, the value of i is:
i= 7
Is the condition (i<=10) is true?
Yes because i=7
Do this
i= 7+1 = 8
The statement System.out.println("value of i = " + i); make provision to print the output:
value of i = 8
Now, the value of i is:
i= 8
Is the condition (i<=10) is true?
Yes because i=8
Do this
i= 8+1 = 9
The statement System.out.println("value of i = " + i); make provision to print the output:
value of i = 9
Now, the value of i is:
i= 9
Is the condition (i<=10) is true?
Yes because i=9
Do this
i= 9+1 = 10
The statement System.out.println("value of i = " + i); make provision to print the output:
value of i = 10
stop because the condition i<=10 is achieved.

If new line \n is introduced i.e., the statement System.out.println("value of i = " + i); is replaced by the statement System.out.println("\n value of i = " + i); i.e.,

```
public class HelloWorld{
public static void main(String [] args){
int i;
for (i=1; i<=10; i++)
System.out.println("\n value of i = " + i);
}
}
```

Then the output on the screen is:
value of i = 1
value of i = 2
value of i = 3
value of i = 4
value of i = 5
value of i = 6
value of i = 7
value of i = 8
value of i = 9
value of i = 10
If the for loop statement for (i=2; i<=10; i++) is written instead of the statement for(i=1; i<=10; i++), then the output on the screen is:
value of i = 2 value of i = 3 value of i= 4 value of i= 5 value of i= 6 value of i = 7 value of i= 8 value of i = 9 value of i= 10
(because i=2 is initialized in the for loop statement the printing started from value of i = 2 and ended at value of i= 10 because of the condition i<=10)
If the for loop statement for(i=1; i<10; i++) is written instead of the statement for (i=1; i<=10; i++), then the output on the screen is:

value of i = 1 value of i = 2 value of i= 3 value of i= 4 value of i= 5 value of i= 6 value of i = 7 value of i= 8 value of i = 9
(Note: the condition i<=10 tells to print till value of i =10 but the condition i<10 tells to print till value of i =9)
If the statement for(i=1; i=10; i++) is written instead of the statement for (i=1; i<=10; i++), then the output on the screen is:
value of i = 10 value of i = 10 value of i = 10 value of i = 10 value of i= 10 value of i= 10 value of i = 10 value of i= 10 value of i = 10 value of i = 10 value of i = 10 value of i = 10 value of i = 10 value of i = 10 ……. (continues).

Note:
If the statement System.out.println("value of i = " + i); is replaced by the statement
System.out.println(" \n " + i);
i.e.,

```
public class HelloWorld{
public static void main(String [] args){
int i;
for (i=1; i<=10; i++)
System.out.println("\n " + i);
}
}
```

The output on the screen is:
1
2
3
4
5
6
7
8
9
10
What is the mistake in the following program:

```
public class HelloWorld{
public static void main(String []args) throws Exception{
System.out.println("Hello World");
}
}
```

Answer:
There is no mistake in the above program. Addition of the statement throws Exception does not make any change in the output displayed on the screen or give rise to any compilation error on the screen.

Program 9.5

Java program to print the first ten natural numbers using for while loop statement

The syntax of while loop statement is:

```
while (this is the condition)
{
execute this statement;
}
```

```
public class HelloWorld{
public static void main(String [] args) {
```

```
int i = 1;
while (i<=10)
{
System.out.println("\n            " +    i++);
}
}
}
```

The output on the screen is:

```
1
2
3
4
5
6
7
8
9
10
```

(a<=10) is the condition and the statements

```
while (a<=10)
{
System.out.println("\n"+    a++);
}
```

imply that while the condition (a<=10) is to print till 10, print till 10 using the statement

```
{
System.out.println("\n"+    a++);
}
```

i.e.,

```
1
2
3
4
5
6
7
8
9
10
```

Note: The statement int i = 1; imply that we are creating an integer variable i and we are initializing i =1.

If the statement int i=1; is replaced by the statement int i;
i.e.,

```
public class HelloWorld{
public static void main(String [] args) {
int i;
while (i<=10)
{
System.out.println("\n          " +    i++);
}
}
}
```

Then the compilation error will be displayed on the screen because initialization is not defined i.e., from where to start is not declared.

If the statement int i = 1; is replaced by the int i = 0;
i.e.,

```
public class HelloWorld{
public static void main(String [] args) {
int i = 0;
while (i<=10)
{
System.out.println("\n              " +    i++);
}
}
}
```

Then the output on the screen is:

```
0
1
2
3
4
5
  6
7
8
9
10
```

Similarly if the statement int i = 0; is replaced by the int i = 7;
Then the output on the screen is:

```
7
8
9
10
```

Program 9.6

Java program to print first 10 numbers using do while loop statement
The syntax of do while loop statement is:

```
do
{
execute this statement;
}
while(this is the condition);
```

```
public class HelloWorld{
public static void main(String [] args){
int i = 1;
do
{
System.out.println(" \n i = " +    i++);
} while (i<=10);
}
}
```

The output on the screen is:

```
i= 1
i = 2
i= 3
```

i= 4
i= 5
i= 6
i = 7
i= 8
i= 9
i= 10

The statements
do
{
System.out.println(" \n i = " + i++);
}while (i<=10);
imply print till i=10 using the statement
do
{
System.out.println(" \n i= " + i++);
}
while the condition (i<=10) is to print till i = 10 (starting from i = 1 because of the statement int i=1;)

Write a program to print
The Internet is not for sissies
10 times using for loop statement.

Answer:

```
public class HelloWorld{
public static void main(String [] args){
int i;
for (i=1; i<=10; i++)
System.out.println("\n The Internet is not for sissies");
}
}
```

Program 9.7

Java program to print the characters from A to Z using for loop, do while loop and while loop statement.

(a) Java program to print the characters from A to Z using for loop statement:

```
public class HelloWorld{
public static void main(String [] args) {
char a;
for( a='A'; a<='Z'; a++)
System.out.println("\n " + a);
}
}
```

The output on the screen:
A
B
C
D
E
F
G
H

I
J
K
L
M
N
O
P
Q
R
S
T
W
X
Y
Z

char means the data type is character.
The statement
char a; imply that we are creating the character a.
If the statement for(a=A; a<=Z; a++) is written instead of the statement for(a='A'; a<='Z'; a++)
i.e., A is used instead of 'A' and Z is used instead of 'Z', then the compilation error will be displayed on the console screen.

(b) Java program to print the characters from A to Z using while loop statement:

```
public class HelloWorld{
public static void main(String [] args) {
char a = 'A';
while (a<='Z')
{
System.out.println("\n    "    + a++);
}
}
}
```

Note: If the statement char a; is written instead of char a = 'A';
i.e.,

```
public class alphabets{
public static void main(String [] args) {
char a;
while (a<='Z')
{
System.out.println("\n    "    + a++);
}
}
}
```

Then the compilation error will be displayed on the screen.

(c) Java program to print the characters from A to Z using do while loop statement:

```
public class HelloWorld{
public static void main(String [] args) {
char a = 'A';
do
{
```

```
System.out.println("\n "    +    a++);
} while (a<='Z');
}
}
```

Program 9.8

Java program to print the given number is even or odd.

```
import java.util.Scanner;
public class HelloWorld{
public static void main(String [] args) {
int a;
Scanner scan = new Scanner(System.in);
System.out.println("Enter a number: ");
a = scan.nextInt();
if(a%2 = = 0)
{
System.out.println("the number is even");
}
else
{
System.out.println("the number is odd");
}
}
}
```

The output on the screen:
Enter a number:
If you enter the number 4
the number is even will be outputted on the screen.

(a%2 = = 0) is the condition and this condition imply: a divided by 2 yields reminder = 0.
For example: if you enter the number 4
Then a = 4
Then 4 divided by 2 yields the remainder = 0
Then the statement

```
{
System.out.println("the number is even");
}
```

make provision to print the output:
the number is even
(Note: in Java language also = = implies equal to)
if you enter the number 3
Then a = 3
Then 3 divided by 2 yields the remainder = 1
Then the statement

```
{
System.out.println("the number is odd");
}
```

make provision to print the output:
the number is odd

Program 9.9

Java program to print the remainder of two numbers

```
import java.util.Scanner;
```

```
public class HelloWorld{
public static void main(String [] args) {
int a, b, c;
Scanner scan = new Scanner(System.in);
System.out.println("Enter a number: ");
a = scan.nextInt();
System.out.println("Enter a number: ");
b = scan.nextInt();
c = a%b;
System.out.println("the remainder of a and b =    " + c);
}
}
```

The output on the screen:
Enter a number:
If you enter the number 3
Enter a number:
If you enter the number 2
the remainder of a and b = 1 will be outputted on the screen.
Since (a =3 and b =2). Therefore:
3 divided by 2 (i.e., a divided by b) yields the remainder equal to 1.

Program 10.0

Java program to check equivalence of two numbers.

```
import java.util.Scanner;
public class HelloWorld{
public static void main(String [] args) {
int x, y;
Scanner scan = new Scanner(System.in);
System.out.println("Enter a number: ");
x = scan.nextInt();
System.out.println("Enter a number: ");
y = scan.nextInt();
if(x-y==0)
{
System.out.println("the two numbers are equivalent");
}
else
{
System.out.println("the number are not equivalent");
}
}
```

The output on the screen:
Enter a number:
If you enter the number 2
Enter a number:
If you enter the number 2
the two numbers are equivalent will be outputted on the screen.

Since 2-2 is equal to 0 (i.e., x-y = = 0). Therefore: the statement

```
{
System.out.println("the two numbers are equivalent");
}
```

makes provision to print the output:
two numbers are equivalent

If you enter the integers 3 and 2
The output on the screen is:
the two numbers are not equivalent
Since 3-2 is not equal to 0 (i.e., x-y != 0). Therefore: the statement
{
System.out.println("the two numbers are not equivalent");
}
makes provision to print the output:
two numbers are not equivalent
(Note: Like in C & C++, in Java language also != implies not equal to)

Program 10.1

Java program to print the leap year or not

```
public class HelloWorld{
public static void main(String [] args) {
int year;
year =1996;
if(year%4==0)
{
System.out.println("leap year");
}
else
{
System.out.println("not a leap year");
}
}
}
```

The output on the screen:
leap year
Since year =1996. Therefore:
1996 divided by 4 (i.e., year divided by 4) yields the remainder equal to 0.
The statement
{
System.out.println("leap year");
}
makes provision to print the output:
leap year
If the year is = 1995. Then
1995 divided by 4 (i.e., year divided by 4) yields the remainder not equal to 0.
The statement
{
System.out.println("not a leap year");
}
makes provision to print the output:
not a leap year

“Writing code has a place in the human hierarchy worth somewhere above grave robbing and beneath managing.”

: GERALD WEINBERG

What will be the output on the screen:

```
public class HelloWorld{
int a =5;
public static void main(String[] args){
int a =2 ;
System.out.println(" value of a = " + a);
}
}
```

Answer:
value of a = 2

If the statement System.out.println(" value of a = " + a); is replaced by the statement
System.out.println(" value of a = " + ::a); (where :: denote scope resolution operator)
i.e.,

```
public class HelloWorld{
int a =5;
public static void main(String[] args){
int a =2 ;
System.out.println(" value of a = " + ::a);
}
}
```

Then the compilation error will be displayed on the screen because like C++ -- java does not hold / support the resolution operator.

Program 10.1

Java program to print whether the given number is positive or negative

```
public class HelloWorld{
public static void main(String [] args){
int a;
a = -35;
if(a>0)
{
System.out.println("number is positive");
}
else
{
System.out.println(" number entered is negative");
}
}
}
```

The output on the screen:
number entered is negative
Since a = -35. Therefore:
a is less than 0 i.e., a<0
The statement

```
{
System.out.println("number is negative");
}
```

makes provision to print the output:
number entered is negative

Program 10.2

Java program to print the sum of the first 10 digits using for loop statement:

```
public class HelloWorld{
public static void main(String [] args){
int i,    sum = 0;
for( i=1; i<=10; i++)
sum = sum + i;
System.out.println("sum of    the first 10 digits = " + sum);
}
}
```

The output on the screen:
sum of the first 10 digits = 55
i.e., 1 + 2 + 3 + 4 + 5 + 6 + 7 + 8 + 9 + 10 = 55

How the sum of the first 10 digits = 55 is outputted on the screen through the for Loop statement
value of i
i=1 (sum = 0 because the sum is initialized to 0 in the statement int i, sum = 0;)
Is i<=10 true?
Yes, do this
sum = sum + i = 0 +1 =1
value of i
i=2 (now the sum = 1)
Is i<=10 true?
Yes, do this
sum = sum + i = 1 +2 =3
value of i
i=3 (now the sum = 3)
Is i<=10 true?
Yes, do this
sum = sum + i = 3 +3 = 6
value of i
i=4 (now the sum = 6)
Is i<=10 true?
Yes, do this
sum = sum + i = 6 + 4= 10
value of i
i=5 (now the sum = 10)
Is i<=10 true?
Yes, do this
sum = sum + i = 10 + 5= 15
value of i
i=6 (now the sum = 15)
Is i<=10 true?
Yes, do this
sum = sum + i = 15 + 6 = 21

value of i
i=7 (now the sum = 21)
Is i<=10 true?
Yes, do this
sum = sum + i = 21 + 7 = 28
value of i
i=8 (now the sum = 28)
Is i<=10 true?
Yes, do this
sum = sum + i = 28 + 8 = 36
value of i

i=9 (now the sum = 36)
Is i<=10 true?
Yes, do this
sum = sum + i = 36 + 9 = 45
value of i
i=10 (now the sum = 45)
Is i<=10 true?
Yes, do this
sum = sum + i = 45 + 10 = 55
stops because the condition is i<=10

The statement System.out.println("sum of the first 10 digits = " + sum); make provision to display the output:
sum of the first10 digits = 55
on the screen.

If the statement int i, sum = 0; is replaced by int i, sum = 1;
Then
value of i
i=1 (sum = 1 because the sum is initialized to 1 in the statement int i, sum = 1;)
Is i<=10 true?
Yes, do this
sum = sum + i = 1 +1 = 2
value of i
i=2 (now the sum = 2)
Is i<=10 true?
Yes, do this
sum = sum + i = 2 + 2 = 4
value of i
i=3 (now the sum = 4)
Is i<=10 true?
Yes, do this
sum = sum + i = 4 +3 = 7
value of i
i=4 (now the sum = 7)
Is i<=10 true?
Yes, do this
sum = sum + i = 7 + 4= 11
value of i
i=5 (now the sum = 11)
Is i<=10 true?
Yes, do this
sum = sum + i = 11 + 5= 16
value of i
i=6 (now the sum = 16)
Is i<=10 true?
Yes, do this
sum = sum + i = 16 + 6 = 22

value of i
i=7 (now the sum = 22)
Is i<=10 true?
Yes, do this
sum = sum + i = 22 + 7 = 29
value of i
i=8 (now the sum = 29)
Is i<=10 true?

Yes, do this
sum = sum + i = 29 + 8 = 37
value of i
i=9 (now the sum = 37)
Is i<=10 true?
Yes, do this
sum = sum + i = 37 + 9 = 46
value of i
i=10 (now the sum = 46)
Is i<=10 true?
Yes, do this
sum = sum + i = 46 + 10 = 56
stops because the condition is i<=10

The statement System.out.println("sum of the first 10 digits = " + sum); make provision to display the output:
sum of the first10 digits = 56 on the screen.
(wrong result because the sum of the first 10 digits is 55)

What will be the output if the for loop statement for(i =1; i<=10; i++) is replaced by the statement for(i =2; i<10; i++)?

Answer: sum of 10 digits = 44

If the statement int i, sum, sum = 0; is written instead of int i, sum = 0;
Then the compilation error message will be displayed on the screen (stating that sum is twice declared).

If the for loop is ended with a semicolon i.e.,
for(i=1; i<=10; i++);
then the compilation error will be displayed on the screen

Even though if i ++ is replaced by ++ i in the for loop statement i.e., if the for loop statement
for (i=1; i<=10; i++)
is replaced by the statement
for (i=1; i<=10; ++ i)
There will be no change in the output on the screen (as observed while compiling in online compilers like Coding ground (Tutorials point)) and if the statement for(i=1; i<=10; i++); is written instead of the statement
for (i=1; i<=10; i++)
Then the Error will be flagged on the screen because for loop statement is ended by a semicolon (;).

Program 10.3

Java program to print the average of the first10 numbers using for loop statement

```
public class HelloWorld{
public static void main(String [] args){
int i, avg, sum = 0;
for( i=1; i<=10; i++)
sum = sum + i;
avg = sum/10;
System.out.println("sum of the first 10 numbers = " + sum);
System.out.println("average of the first 10 numbers =  " + avg);
}
}
```

The output on the screen:
sum of the first 10 numbers = 55

average of the first 10 numbers = 5
The average of the first 10 numbers = 55/10 = 5.5 not 5. But the output on the screen is:
average of the first 10 numbers = 5
because int is used instead of float.

If the data type float is used i.e.,

```
public class HelloWorld{
public static void main(String [] args) {
float i, avg, sum = 0;
for( i=1; i<=10; i++)
sum = sum + i;
avg = sum/10;
System.out.println("sum of the first 10 numbers = " + sum);
System.out.println("average of the first 10 numbers = " + avg);
}
}
```

The output on the screen:
sum of the first 10 numbers = 55
average of the first 10 numbers = 5.5

Program 10.4

Java program to print the product of the first10 digits using for loop statement

```
public class HelloWorld{
public static void main(String [] args) {
int i, product = 1;
for( i=1; i<=10; i++)
product = product * i;
System.out.println("the product of the first10 digits = " + product);
}
}
```

The output on the screen:
the product of the first 10 digits = 3628800
i.e., 1 * 2 * 3 * 4 * 5 * 6 * 7 * 8 * 9 * 10 = 3628800

How the product of the first 10 digits = 3628800 is outputted on the screen through the for Loop statement

value of i
i=1 (product = 1 because the product is initialized to 1 in the statement int i, product = 1;)
Is i<=10 true?
Yes, do this
product = product * i = 1 * 1 =1
value of i
i=2 (now the product = 1)
Is i<=10 true?
Yes, do this
product = product * i = 1 * 2 = 2
value of i
i=3 (now the product = 2)
Is i<=10 true?
Yes, do this
product = product * i = 2 * 3 = 6
value of i
i=4 (now the product = 6)

Is i<=10 true?
Yes, do this
product = product * i = 6 * 4 = 24
value of i
i=5 (now the product =24)
Is i<=10 true?
Yes, do this
product = product * i = 24 * 5 =120
value of i
i=6 (now the product =120)
Is i<=10 true?
Yes, do this
product = product * i = 120 * 6 = 720
value of i
i=7 (now the product =720)
Is i<=10 true?
Yes, do this
product = product * i = 720 * 7 = 5040
value of i
i=8 (now the product =5040)
Is i<=10 true?
Yes, do this
product = product * i = 5040 * 8 = 40320
value of i
i=9 (now the product = 40320)
Is i<=10 true?
Yes, do this
product = product * i = 40320 * 9 = 362880
value of i
i=10 (now the product = 362880)
Is i<=10 true?
Yes, do this
product = product * i = 362880 * 10 = 3628800
stops because the condition is i<=10

The statement System.out.println("the product of the first10 digits = " + product); make provision to display the output:
the product of the first 10 digits = 3628800
on the screen.

If the statement int i, product = 1; is replaced by int i, product = 0;
Then
value of i
i=1 (product = 0 because the product is initialized to 0 in the statement int i, product = 0;)
Is i<=10 true?
Yes, do this
product = product * i = 0 * 1 = 0
value of i
i=2 (now the product = 0)
Is i<=10 true?
Yes, do this
product = product * i = 0 * 2 = 0
value of i
i=3 (now the product = 0)
Is i<=10 true?
Yes, do this

product = product * i = 0 * 3 = 0
value of i
i=4 (now the product = 0)
Is i<=10 true?
Yes, do this
product = product * i = 0 * 4 = 0
value of i
i=5 (now the product = 0)
Is i<=10 true?
Yes, do this
product = product * i = 0 * 5 = 0
value of i
i=6 (now the product = 0)
Is i<=10 true?
Yes, do this
product = product * i = 0 * 6 = 0
value of i
i=7 (now the product = 0)
Is i<=10 true?
Yes, do this
product = product * i = 0 * 7 = 0
value of i
i=8 (now the product = 0)
Is i<=10 true?
Yes, do this
product = product * i = 0 * 8 = 0
value of i
i=9 (now the product = 0)
Is i<=10 true?
Yes, do this
product = product * i = 0 * 9 = 0
value of i
i=10 (now the product = 0)
Is i<=10 true?
Yes, do this
product = product * i = 0 * 10 = 0
stops because the condition is i<=10
The statement System.out.println("the product of the first10 digits = " + product); make provision to display the output:
the product of the first 10 digits = 0
on the screen.
(wrong result because the product of the first10 digits is 3628800)

If the statement for(i=1; i<=10; i++) is replaced by for(i=5; i<=8; i++)
Then
value of i
i=5 (product = 1 because the product is initialized to 1 in the statement int i, product = 1;)
Is i<=8 true?
Yes, do this
product = product * i = 5 * 1 = 5
value of i
i=6 (now the product = 5)
Is i<=8 true?
Yes, do this
product = product * i = 5 * 6 = 30
value of i

i=7 (now the product = 30)
Is i<=8 true?
Yes, do this
product = product * i = 30 * 7 = 210
value of i
i=8 (now the product = 210)
Is i<=8 true?
Yes, do this
product = product * i = 210 * 8 = 1680
stops because the condition i<=8 is achieved and the statement
System.out.println("the product of the first10 digits = " + product);
make provision to display the output:
the product of the first 10 digits = 1680
on the screen.

Note: If the statement
int i, product, product = 1;
is written instead of int i, product = 1;
Then the compilation error message is flagged on the screen (stating that product is twice declared).

Program 10.5

Java Program to print the table of a number using the for loop statement

```
import java.util.Scanner;
public class HelloWorld{
public static void main(String [] args){
int n, i;
Scanner scan = new Scanner(System.in);
System.out.println("Enter a number: ");
n = scan.nextInt();
for( i=1; i<=5; i++)
System.out.println ( \n n + " * " + i + " = " + n * i);
}
}
```

Enter any number:
If you enter the number 2 (i.e., n=2)
2 * 1 = 2
2 * 2 = 4
2 * 3 = 6
2 * 4 = 8
2 * 5 = 10
will be outputted on the screen.

How the execution takes its Way through the for Loop statement

Since you entered the number 2, therefore: n=2.
value of i
i=1
Is i<=5 true?
Yes, print this
2 * 1 = 2
using the statement System.out.println (\n n + " * " + i + " = " + n * i);

value of i
i=2
Is i<=5 true?
Yes, print this

2 * 2 = 4
using the statement System.out.println (\n n + " * " + i + " = " + n * i);

value of i
i=3
Is i<=5 true?
Yes, print this
2 * 3 = 6
using the statement System.out.println (\n n + " * " + i + " = " + n * i);

value of i
i=4
Is i<=5 true?
Yes, print this
2 * 4 = 8
using the statement System.out.println (\n n + " * " + i + " = " + n * i);

value of i
i=5
Is i<=5 true?
Yes, print this
2 * 5 = 10
using the statement System.out.println (\n n + " * " + i + " = " + n * i);

stop Now because the condition i <=5 is achieved.

If the symbol * is replaced by +
i.e.,

```
import java.util.Scanner;
public class HelloWorld{
public static void main(String [] args){
int n, i;
Scanner scan = new Scanner(System.in);
System.out.println("Enter a number: ");
n = scan.nextInt();
for( i=1; i<=5; i++)
System.out.println ( \n n + " + " + i + " = " + n + i);
}
}
```

The output on the screen:
Enter a number:
If you enter the number 2 (i.e., n=2)
2 + 1 = 3
2 + 2 = 4
2 + 3 = 5
2 + 4 = 6
2 + 5 = 7
will be outputted on the screen.

Program 10.6

Java program to print the first 10 numbers starting from one together with their squares

```
public class HelloWorld{
public static void main(String[] args){
int i;
```

```
for( i=1; i<=10; i++)
System.out.println(" number = " +    i +    " its square =    " + i*i);
}
}
```

The output on the screen:
number = 1 its square=1number = 2 its square=4number = 3 its square=9number = 4 its square=16number = 5 its square=25number = 6 its square=36number = 7 its square=49number = 8 its square=64number = 9 its square=81number= 10 its square=100

If the statement System.out.println(" number = " + i + " its square = " + i*i); is replaced by the statement System.out.println(" \n number = " + i + " its square = " + i*i);

i.e., if the above program is rewritten as:

```
public class HelloWorld{
public static void main(String[] args){
int i;
for( i=1; i<=10; i++)
System.out.println(" \n number = " +    i +    " its square =    " + i*i);
}
}
```

Then the output on the screen is:
number = 1 its square=1
number = 2 its square=4
number = 3 its square=9
number = 4 its square=16
number = 5 its square=25
number = 6 its square=36
number = 7 its square=49
number = 8 its square=64
number = 9 its square=81
number= 10 its square=100

If the statement System.out.println(" \n number = " + i + " its square = " + i*i); is replaced by the statement System.out.println(" \n number = " + i + " \t its square = " + i*i);
i.e., if the above program is rewritten as:

```
public class HelloWorld{
public static void main(String[] args){
int i;
for( i=1; i<=10; i++)
System.out.println(" \n number = " +    i +    " \t its square =    " + i*i);
}
}
```

Then the output on the screen is:
number=1 its square=1
number=2 its square=4
number=3 its square=9
number=4 its square=16
number=5 its square=25
number=6 its square=36
number=7 its square=49
number=8 its square=64
number=9 its square=81

number=10 its square=100

tab /t is included because to leave space between
number =1 and its square=1

If the statement System.out.println(" \n number = " + i + " \t its square = " + i*i); is replaced by the statement
System.out.println(" \n number = " + i + " \n its square = " + i*i);
i.e., if the above program is rewritten as:

```
public class HelloWorld{
public static void main(String[] args){
int i;
for( i=1; i<=10; i++)
System.out.println(" \n number = " + i + " \n its square = " + i*i);
}
}
```

Then the output on the screen is:

```
number = 1
its square=1
number = 2
its square=4
number = 3
its square=9
number = 4
its square=16
number = 5
its square=25
number = 6
its square=36
number = 7
its square=49
number = 8
its square=64
number = 9
its square=81
number= 10
its square=100
```

Write a program to print the first 10 numbers starting from one together with their squares and cubes:

Answer:

```
public class HelloWorld{
public static void main(String[] args) throws Exception{
int i;
for( i=1; i<=10; i++)
System.out.println(" \n number = " + i + " its square = " + i*i + " its cube = " + i*i*i);
}
}
```

“When I am working on a problem I never think about beauty. I think only how to solve the problem.
But when I have finished, if the solution is not beautiful, I know it is wrong.”
--R. BUCKMINSTER FULLER

Program 10.7

Java program to print the sum of two numbers using method

```
public class HelloWorld{
public static void main(String[] args){
int a, b, c;
a = 11;
b = 6;
c = add (a, b);
System.out.println(" sum of two numbers = " + c);
}
public static int add (int a, int b) {
 return (a+b) ;
}
}
```

The output on the screen:
sum of two numbers = 17

public static void main(String[] args) imply main method and
{
} imply the body of the main method with in which the program statements:
int a, b, c;
a = 11;
b = 6;
c = add (a, b);
System.out.println(" sum of two numbers = " + c); are written.

Like in C ++ (the function declaration is not made) and unlike in C ((the function declaration is made) -- there is no need for method declaration in Java (i.e., without the method declaration the program will be successfully executed and the result will be outputted on the screen)

public static int add (int a, int b) imply method to add two integers x and y and
{
 return (a+b) ;
}
} imply the body of the method public static int add (int a, int b)

main method public static void main(String[] args) and the method public static int add (int a, int b) should be written inside the body of the public class HelloWorld.

The statement int a, b, c; imply that we creating the integer variables a, b and c. The statements
a = 11;
b = 6;
c = add (a, b);
imply that we are assigning the values to the created variables.

The statement c = add (x, y); imply method call (i.e., we are calling the method public static int add (int a, int b) to add the values (i.e., 11 and 6) and return the result (i.e., 17) to the statement System.out.println(" sum of two numbers = " + c); to make provision to display the output of the sum of two entered numbers as 17 on the screen.

If you want to supply the values for a and b through the key board, then we have to include the statements:
import java.util.Scanner;
Scanner scan = new Scanner(System.in);
and replace the statements
a = 11;
b = 6;
by the statements

```
System.out.println("Enter any two numbers: ");
a = scan.nextInt();
b = scan.nextInt();
```

i.e., the above program should be rewritten as:

```
import java.util.Scanner;
public class HelloWorld{
public static void main(String[] args) {
int a, b, c;
Scanner scan = new Scanner(System.in);
System.out.println("Enter any two numbers: ");
a = scan.nextInt();
b = scan.nextInt();
c = add (a, b);
System.out.println(" sum of two numbers   = " + c);
}
public static int add (int a, int b) {
return (a+b) ;
}
}
```

The output on the screen:
Enter any two numbers:
If you enter the values 2 and 3
sum of two numbers = 5 will be outputted on the screen.

Java program to print the product of two numbers using method

```
import java.util.Scanner;
public class HelloWorld{
public static void main(String[] args) {
int a, b, c;
Scanner scan = new Scanner(System.in);
System.out.println("Enter any two numbers: ");
a = scan.nextInt();
b = scan.nextInt();
c = mult (a, b);
System.out.println(" product of two numbers   = " + c);
}
public static int mult (int a, int b){
return (a*b) ;
}
}
```

The output on the screen:
Enter any two numbers:
If you enter the values 2 and 3
product of two numbers = 6 will be outputted on the screen.

Java program to print the greatest of two numbers using method

```
import java.util.Scanner;
public class HelloWorld{
public static void main(String[] args) {
int a, b;
Scanner scan = new Scanner(System.in);
System.out.println("Enter any two numbers: ");
a = scan.nextInt();
```

```
b = scan.nextInt();
result =   max (a, b);
System.out.println(" largest of two numbers   = " + result);
}
public static int max (int a, int b) {
if(a>b)
return a;
if(b>a)
return b;
}
}
```

The output on the screen:
Enter any two numbers:
If you enter two numbers 3 and 5
largest of two numbers= 5 will be outputted on the screen.

Java program to print the greatest of three numbers using method

```
import java.util.Scanner;
public class HelloWorld{
public static void main(String[] args) {
int a, b, c;
Scanner scan = new Scanner(System.in);
System.out.println("Enter any three numbers: ");
a = scan.nextInt();
b = scan.nextInt();
c = scan.nextInt();
result =   max (a, b, c);
System.out.println(" largest of three numbers   = " + result);
}
public static int max (int a, int b, int c) {
if(a>b)
return a;
if(b>a)
return b;
if(c>a&&c>b)
return c;
}
}
```

The output on the screen:
Enter any three numbers:
If you enter three numbers 3, 5 and 10
largest of three numbers = 10 will be outputted on the screen.

Java program to print the square of the number using method

```
import java.util.Scanner;
public class HelloWorld{
public static void main(String[] args) {
int x;
Scanner scan = new Scanner(System.in);
System.out.println("Enter any number: ");
x = scan.nextInt();
System.out.println("square of the number = " + square (x));
}
public static int square (int x){
```

```
return x*x;
}
}
```

The output on the screen is:
Enter any number:
If you enter the number 5
square of the number = 25 will be outputted on the screen.

Program 10.8

Switch (case) allows to make decision from the number of choices i.e., from the number of cases
For example:

```
public class HelloWorld{
public static void main(String[] args)throws Exception{
char ch;
System.out.print("Enter a character:");
ch = (char)System.in.read();
switch(ch)
{
case 'R':
System.out.print("Red");
break;
case 'W':
System.out.print("White");
break;
case 'Y':
System.out.print("Yellow");
break;
case 'G':
System.out.print("Green");
break;
default:
System.out.print("Error");
break;
}
}
}
```

The output on the screen is:
Enter a character:
If you enter a character R
Red will be outputted on the screen.

switch(ch) allow to make decision from the number of choices i.e., from the number of cases
case 'R':
case 'W':
case 'Y':
case 'G':
Since we have entered the character R (which corresponds to case 'R':)
The statement
System.out.print("Red");
make provision to display the output
Red
on the screen.
Suppose you enter a character K
The output on the screen is:

Error
(Entered character K does not correspond to any of the cases
case 'R':
case 'W':
case 'Y':
case 'G':
Therefore the statements
default:
System.out.print("Error");
make provision to display the output
Error
on the screen).
If the statements
{
case 'R':
System.out.print("Red");
break;
case 'W':
System.out.print("White");
break;
case 'Y':
System.out.print("Yellow");
break;
case 'G':
System.out.print("Green");
break;
default:
System.out.print("Error");
break;
} are replaced by the statements
{
case 'R':
System.out.print("Red");
case 'W':
System.out.print("White");
case 'Y':
System.out.print("Yellow");
break;
case 'G':
System.out.print("Green");
break;
default:
System.out.print("Error");
break;
}
i.e., if the statement break; is not written after the statements
case 'R':
System.out.print("Red");

case 'W':
System.out.print("White");
Then the output on the screen is:
Red
White
Yellow
i.e., the output is printed till yellow even though you have entered the character R.

Note: C and C++ supports pointers and structures whereas Java does not i.e., Java do not support structures and pointers because JVM (Java virtual machine—a core component of java) do not support structures and pointers.

Program 10.9

Java program to print the output

```
Element [0] = 16
Element [1] = 18
Element [2] = 20
Element [3] = 25
Element [4] = 36
using arrays:

public class HelloWorld{
public static void main(String[] args){
int i;
int [] num = {16, 18, 20, 25, 36};
for(i=0; i<5; i++)
System.out.println("Element [" +   i + " ] = " + num[i]);
}
}
```

The output on the screen:

```
Element [0] = 16
Element [1] = 18
Element [2] = 20
Element [3] = 25
Element [4] = 36
```

Ends because of the condition i<5.

Note:
Array declaration in C:

```
int num [5] = {16, 18, 20, 25, 36};
or
int num [] = {16, 18, 20, 25, 36};
```

Array declaration in C++:

```
int num [5] = {16, 18, 20, 25, 36};
or
int num [] = {16, 18, 20, 25, 36};
```

But array declaration in java:

```
int [] num = {16, 18, 20, 25, 36};
```

*If you write int [5] num = {16, 18, 20, 25, 36}; instead of int [] num = {16, 18, 20, 25, 36};
Then the compilation error will be displayed on the screen.

The statement int [] num = {16, 18, 20, 25, 36}; imply that we are creating an integer array (and the name of array is num) consisting of 5 values (i.e., 16, 18, 20, 25, 36) of the same data type int.

With the declaration int [] num = {16, 18, 20, 25, 36}; -- computer creates 5 memory cells (because there are 5 elements within the braces {}) with name num[0], num[1], num[2], num[3], num[4]. And since
int [ ] num = {16, 18, 20, 25, 36};
the values 16, 18, 20, 25, 36 are stored in num[0], num[1], num[2], num[3], num[4] respectively.

How the execution takes its way through the for loop statement

value of i
i=0
Is i<5 true?
Yes, print this
Element [0] = 16
using the statement
System.out.println("Element [" + i + "] = " + num[i]);
value of i
i=1
Is i<5 true?
Yes, print this
Element [1] = 18
using the statement
System.out.println("Element [" + i + "] = " + num[i]);

value of i
i=2
Is i<5 true?
Yes, print this
Element [2] = 20
using the statement
System.out.println("Element [" + i + "] = " + num[i]);

value of i
i=3
Is i<5 true?
Yes, print this
Element [3] = 25
using the statement
System.out.println("Element [" + i + "] = " + num[i]);

value of i
i=4
Is i<5 true?
Yes, print this
Element [4] = 36
using the statement
System.out.println("Element [" + i + "] = " + num[i]);
Stop because the condition is i<5.

If i<=5 i.e., if the for loop statement was
for(i=0; i<=5; i++)
Then the output on the screen is:
Element [0] = 16
Element [1] = 18
Element [2] = 20
Element [3] = 25
Element [4] = 36
Element [5] = 365

365 is the number stored in the memory i.e., any number stored in the memory will be displayed.

If the statement int [] a = {16, 18, 20, 25, 36}; is replaced by the statement
int [5] a = {16, 18, 20, 25, 36};

or by the statement
int num [i] = {16, 18, 20, 25, 36};
Then the compilation error will be displayed on the screen.

Suppose the statement System.out.println("Element [" + i + "] = " + a[i]); is replaced by the statement System.out.println("Element [" + i + "] = " + a[0]);

Then the output on the screen is:
Element [0] = 16
Element [1] = 16
Element [2] = 16
Element [3] = 16
Element [4] = 16

Suppose the statement System.out.println("Element [" + i + "] = " + a[i]); is replaced by the statement System.out.println("Element [" + i + "] = " + a[1]);

Then the output on the screen is:
Element [0] = 18
Element [1] = 18
Element [2] = 18
Element [3] = 18
Element [4] = 18

Suppose the statement System.out.println("Element [" + i + "] = " + a[i]); is replaced by the statement System.out.println("Element [" + i + "] = " + a[2]);
i.e., a[2] corresponds to the output:
Element [0] = 20
Element [1] = 20
Element [2] = 20
Element [3] = 20
Element [4] = 20

Suppose the statement System.out.println("Element [" + i + "] = " + a[i]); is replaced by the statement System.out.println("Element [" + i + "] = " + a[3]);
i.e., a[3] corresponds to the output:
Element [0] = 25
Element [1] = 25
Element [2] = 25
Element [3] = 25
Element [4] = 25

Suppose the statement System.out.println("Element [" + i + "] = " + a[i]); is replaced by the statement System.out.println("Element [" + i + "] = " + a[4]);
i.e., a[4] corresponds to the output:
Element [0] = 36
Element [1] = 36
Element [2] = 36
Element [3] = 36
Element [4] = 36

(a) Java program to print the sum of the elements in array.

```
public class HelloWorld{
public static void main(String[] args){
int i, sum = 0;
```

```
int [] num = {16, 18, 20, 25, 36};
for(i=0; i<5; i++)
sum = sum + num[i];
System.out.println("Sum of the Elements in the array  = " +  sum);
}
}
```

The output on the screen:
Sum of the Elements in the array = 115
i.e., 16 + 18 + 20 + 25 + 36 = 115

How the Execution takes its way through the for loop statement

value of i
i=0 (sum = 0 because the sum is initialized to 0 in the statement int i, sum = 0;)
Is i<5 true?
Yes, do this
sum = sum + num[i] = sum + num[0] = 0 +16 =16
value of i
i=1 (now the sum = 16)
Is i<5 true?
Yes, do this
sum = sum + num[i] = sum + num[1] = 16 +18 =34

value of i
i=2 (now the sum = 34)
Is i<5 true?
Yes, do this
sum = sum + num[i] = sum + num[2] = 34 +20 =54

value of i
i=3 (now the sum = 54)
Is i<5 true?
Yes, do this
sum = sum + num[i] = sum + num[3] = 54 +25 =79

value of i
i=5 (now the sum = 79)
Is i<5 true?
Yes, do this
sum = sum + num[i] = sum + num[5] = 79 + 36 =115
stops because the condition is i<5
The statement System.out.println("Sum of the Elements in the array = " + sum); make provision to display the output:
Sum of the Elements in the array = 115
on the screen.

If the statement
int i, sum = 0; is replaced by int i, sum = 1;
Then The output on the screen:
Sum of the Elements in the array = 116
(wrong result because the sum of the elements in the array is 115).

(b) Java program to print the average of the elements in the array

```
public class HelloWorld{
```

```
public static void main(String[] args){
int i, avg, sum = 0;
int [] num = {16, 18, 20, 25, 36};
for(i=0; i<5; i++)
sum = sum + num[i];
avg = sum/5;
System.out.println("Sum of the Elements in the array = " +    sum);
System.out.println("average of the Elements in the array = " +    avg);
}
}
```

The output on the screen:
Sum of the Elements in the array = 115
average of the elements in the array = 23

Write a program to print

Einstein [0] = E
Einstein [1] = I
Einstein [2] = N
Einstein [3] = S
Einstein [4] = T
Einstein [5] = E
Einstein [6] = I
Einstein [7] = N
using arrays

Answer:

```
public class HelloWorld{
public static void main(String[] args) throws Exception{
int i;
char [] name = {' E' , ' I', ' N', ' S', ' T ', ' E', ' I', ' N'};
for(i=0; i<8; i++)
System.out.println("Einstein [" + i + " ] = " + name[i]);
}
}
```

What will be the output of the following program?

```
public class HelloWorld{
public static void main(String[] args) throws Exception{
int i;
int [] name = {' E' , ' I', ' N', ' S', ' T ', ' E', ' I', ' N'};
for(i=0; i<8; i++)
System.out.println("Einstein [" +   i + " ] = " + name[i]);
}
}
```

Answer:

Einstein [0] = 69
Einstein [1] = 73
Einstein [2] = 78
Einstein [3] = 83
Einstein [4] = 84
Einstein [5] = 69

```
Einstein [6] = 73
Einstein [7] = 78

public class HelloWorld{
public static void main(String[] args) throws Exception{
int i;
char [] name = {' E' , ' I', ' N', ' S', ' T ', ' E', ' I', ' N'};
for(i=0; i<8; i++)
System.out.println("Einstein [" +   i + " ] = " + name[i]);
}
}
```

Answer:

```
Einstein [0] = E
Einstein [1] = I
Einstein [2] = N
Einstein [3] = S
Einstein [4] = T
Einstein [5] = E
Einstein [6] = I
Einstein [7] = N

public class HelloWorld{
public static void main(String[] args) throws Exception{
int i;
char [] body   = {'b', 'o', 'd', 'y'};
for(i=0; i<4; i++)
System.out.println("body [" +   body [i] + " ] = " + body [i]);
}
}
```

Answer:

```
body [b] = b
body [o] = o
body [d] = d
body [y] = y
```

Examine the following program and write the output:

```
public class HelloWorld {
public static void main(String [] args) throws Exception{
System.out.println(" \n E=mc squared Einstein's famous equation that gave birth to the atom bomb and heralded a new world of atomic physics");
System.out.println(" \n                                                                                                    ");
System.out.println("\n E = energy m = mass c = speed of light in vacuum");
System.out.println("\n As we know c squared is huge so if you convert a small amount of mass you'll get a tremendous amount of energy");
System.out.println(" \n For example if you convert 1kg of mass you'll get energy of ");
int E, m, c;
m=1;
c=300000000;
E=m*c*c;
System.out.println("\n " +   E +   " joules ");
```

```
System.out.println(" \n Suppose c would have been 3*10 to the power of -8 meter per second then For 1 kg of mass E = 9 *10 to the power of -16 joules");
System.out.println(" \n hence thousands and thousands of hydrogen atoms in the sun would have to burn up to release 4 * 10 to the power of 26 joules of energy per second in the form of radiation");
int i;
for(i=0; i<5;i++)
System.out.println(" \n Therefore sun would have ceased ");
System.out.println("\n to form black hole even before an ooze of organic molecules would react and built earliest cells and then advance to a wide variety of one celled organisms and evolve through a highly sophisticated form of life to primitive mammals");
int v;
v=300000000;
c=300000000;
if (v==c)
System.out.println("\n rest mass of the photon is zero because light travels at the speed of light");
else
System.out.println(" \n Albert Einstein's special theory of relativity has to be rewritten");
System.out.println("\n masses of the individual substances are 16 \t 18 \t 19 \t 20\t 21\t kilograms");
int j, sum = 0;
int[] num = {16, 18, 19, 20, 21};
for(j=0; j<5;j++)
sum = sum + num [j];
System.out.println("\n sum of the    masses of the individual substances =    " + sum);
System.out.println("\n sum of the    energies of the individual substances =    "    + sum * 300000000 * 300000000);
System.out.println("\n average energy = " + sum / 5);
System.out.println("\n                                                                                                    ");
System.out.println("\n                                                                                                    ");
int p, q, r;
p= 6;
q= 3;
System.out.println("\n expected energy = "    +    p + "multiplied by ten to the power of minus 16 joules calculated using Einstein equation");
System.out.println("\n experimental energy = " + q + "multiplied by ten to the power of minus 16 joules calculated using Einstein equation");
r= q-p;
System.out.println("\n difference between experimental energy and expected energy =" + r + " multiplied by ten to the power of minus 16 joules");
System.out.println("\n absolute value of the difference between experimental energy and expected energy =" + Math.abs(r) + "multiplied by ten to the power of minus 16 joules");
char k;
char [ ] einstein={'e', 'm', 'c', 's', 'q', 'u', 'a', 'r', 'e', 'd'};
for(k=0; k<10; k++)
System.out.println("\n Einstein[" + einstein [k] + "] = " + einstein [k]);
int u;
for(u=1; u<3; u++)
System.out.println("\n    Hey! Einstein may not be wrong please repeat the experiment");
System.out.println("\n                                                *                ");
System.out.println("\n                                              *****              ");
System.out.println("\n                                    **Albert Einstein**          ");
System.out.println("\n                                      **e= mc squared**          ");
System.out.println("\n                                                 *****");
   double f, h, E1;
f=2.5;
h= Math.pow(6.625, -34);
E1 = h*f;
System.out.println("\n energy    calculated using the Planck equation = " + E1);
```

```
if (E==E1)
System.out.println("\n hf cannot be equivalent to mc squared");
if(E>E1)
System.out.println("\n hf can be equivalent to mc squared");
if (E<E1)
System.out.println("\n Einstein and Planck equation cannot be equalized");
System.out.println("\n                                                    ");
System.out.println("\n                                                                                    ");
System.out.println("\n for more details please refer the book          ");
System.out.println("\n Einsteinian Physics");
System.out.println("*********************************************\n");
System.out.println("*********************************************\n");
System.out.println("According to the Albert Einstein's law of variation of mass with velocity: \n");
System.out.println("M = m0 / sqrt ((1- (u/c) squared) \n ");
System.out.println("M = mass of the moving body \t    m0 = rest mass of the body \t u= velocity of the body \t c=
speed of light in vacuum\n ");
{
double M, m0;
int g, b;
m0 = 0.999;
g = 200000000;
b = g/c * g /c;
M = m0 / Math.sqrt (1 - b);
System.out.println(" \n Mass of the moving body =    " +    M);
if (M==m0 || M<m0)
System.out.println("\n body is at rest or the body is moving with nonrelativistic speed");
else
System.out.println("\n body is moving with relativistic speed");
}
}
}
```

Note:

```
import java.util.Scanner;
public class HelloWorld {
public static void main(String [] args) {
int x, y;
Scanner scan = new Scanner(System.in);
System.out.print("Enter any Number: ");
x = scan.nextFloat();
System.out.print("Enter any Number: ");
y = scan.nextInt();
System.out.println(" square root of x = " + Math.sqrt(x));
System.out.println(" square root of y = " + Math.sqrt(y));
}
}
```

The output on the screen:
Enter any Number:
If you enter the number 9
square root of x = 3 will be outputted on the screen.
Enter any Number:
If you enter the number 4
square root of y = 2 will be outputted on the screen.

If

```
                                        /*

*/
```

is introduced i.e., if the above program is rewritten as:

```
import java.util.Scanner;
public class HelloWorld {
public static void main(String [] args) {
int x, y;
Scanner scan = new Scanner(System.in);
System.out.print("Enter any Number: ");
x = scan.nextInt();
/*
System.out.print("Enter any Number: ");
y = scan.nextInt();
                                                                        */
System.out.println(" square root of x = " + Math.sqrt(x));
/*
System.out.println(" square root of y = " + Math.sqrt(y));
                                                                        */
}
}
```

Then the output on the screen is:
Enter any Number:
If you enter the number 9
square root of x = 3 will be outputted on the screen.

What is the mistake in the following program:

```
public class HelloWorld {
public static void main(String [] args) {
long float x;
Scanner scan = new Scanner(System.in);
System.out.print("Enter any Number: ");
x = scan.nextFloat();
System.out.println(" square root of x = " + Math.cbrt(x));
}
```

Answer:
long float x; should not be used -- only float x should be used because Java do not support the data type such as long int, long float etc.

Comparison of C, C++ and Java

- C & C++ support pointers and structures while Java do not.
- The code of C and C++ are directly converted into machine level language and it is executed while the code of Java is converted into Java byte codes and then it is converted into machine level language and it is executed.
- C uses scanf as input function to read the character or integer entered through the keyboard and printf as output function to print the output on the screen. C++ uses cin as input function to read the character or integer entered through the keyboard and cout as output function to print the output on the screen. But Java uses scan.nextInt() or scan.nextFloat() as input method to read the variable entered through the keyboard and System.out.println as output method to print the output on the screen.
- Functions are in C & C++ whereas methods are in Java.
- C & C++ are platform dependent whereas Java is platform independent.
- In C & C++, program instruction codes are written and executed within the body of main function main() where as in Java -- program instruction codes are written and executed within

the body of main method public static void main(String[] args).

- Data types like int float, char are same in C, C++ & Java.
- C is structured language whereas C++ & Java is object oriented language (i.e., C++ & Java has the extensive power and immense extensibility to write large scale complex programs).
- Operators such as %d, %f & %c are used in C whereas no operators are used in C++ & Java.
- A program written in Java usually requires more memory space than the same program written in C &C++.

Bibliography

1. Let Us C by Yashavant Kanetkar.
2. C for Dummies by Dan Gookin.
3. C ++: The complete reference by Herbert Schildt.
4. Programming with C by Byron S. Gottfried.
5. INTRODUCTION to JAVA by Jane Meyerowitz.
6. Java 2: The Complete Reference by Herbert Schildt.
7. Java For Dummies by Barry Burd.
8. Computer Concepts and C Programming by P.B. Kotur.
9. C PROGRAMMING TUTORIAL: Simply Easy Learning by tutorialspoint.com.
10. C PROGRAMMING NOTE by T K Rajan.
11. An Introduction to the C Programming Language and Software Design by Tim Bailey.
12. JAVA Elements: Principles of Programming in JAVA by Bailey.
13. C ++: A Beginners Guide, Teach Yourself C++ by Herbert Schildt.
14. C, C ++ & Java (www. w3 schools. com).
15. A programming with class: A C++ introduction to computer science by Kamin.
16. JAVA hand book by Naughton.
17. Teach yourself JAVA by O'Neil.
18. AT & T Bell laboratories: The C programmer's hand book.
19. C++ Program Design by Cohoon.
20. An introduction to object oriented programming with Java by WU THOMAS.
21. [Stroustrup,1994] Bjarne Stroustrup: The Design and Evolution of C++. Addison Wesley. 1994.
22. [Stroustrup,1991] Bjarne Stroustrup: The C++Programming Language.AddisonWesley.1991.
23. J. Gosling, B. Joy, and G. Steele. The Java Language Specification. Java Series. Sun Microsystems, 1996.
24. Ashok N Kamthane, Programming and Data structures, Pearson Education.
25. A programming language independent companion to Roberge/Bauer/Smith, "Engaged Learning for Programming in C++: A Laboratory Course", Jones and Bartlett Publishers, 2nd Edition, ©2001, ISBN 0763714232.
26. A TUTORIAL ON POINTERS AND ARRAYS IN C by Ted Jensen.
27. Pure basic A beginners guide to Computer Programming by Gary Willoughby.
28. C++ for dummies by Stephen Randy Davis.
29. C to Java: Converting Pointers into References by Erik D. Demaine.
30. Why C++ is not just an Object Oriented Programming Language by Bjarne Stroustrup.
31. [Stroustrup, 1994] Bjarne Stroustrup: The Design and Evolution of C++. AddisonWesley.1994.
32. Java Programming: A Practical Approach by Xavier.
33. Herb Schildt's Java Programming Cookbook by Herbert Schildt.
34. The Java Programming Language by Arnold.
35. Computer Concepts and Programming in C by A.P. Godse, D.A. Godse .
36. Programming in C by Stephen G. Kochan.
37. C Programming: The Definitive Beginner's Reference by Harry. H. Chaudhary.
38. The C Programming Language: 2nd Edition by Brian W. Kernighan, Dennis M. Ritchie.
39. Practical C++ Programming by Steve Oualline.
40. COMPUTER BASICS AND C PROGRAMMING by V. RAJARAMAN.
41. Computer Fundamentals and Programming in C by J. B. Dixit.
42. Java: how to program by Deitel & Deitel, Prentice Hall 1997.
43. Java: A framework for programming and problem solving by Kenneth Lambert and Martin Osborne, PWS Publishing Company 1999.
44. Developing Java Software by Russel Winder and Graham Roberts. John Wiley & Sons 1998.

4/25/2016