Structured Programming Approach (MU)

Harish G. Narula

Semester II – Common to All Branches

(Book Code – etME175)

Chapter	Chapter Name
Chapter 1	Introduction to Computer, Algorithm and Flo
Chapter 2	Fundamentals of C-Programming
Chapter 3	Control Structures : Looping

Chapter 4	**Control Structures : Branching and Nested Co Structure**
Chapter 5	**Functions and Parameter**
Chapter 6	**Arrays, String, Structure, Union, Pointers and**

CHAPTER 1

Introduction to Computer, Algorithm and Flowchart

Syllabus Topic : Turing Model

Q. Write a short note on Turing model.

Ans. :

Turing model

- A Turing machine is an abstract machine that obeys a set of rules laid down for it. The more practical form of the same is called as a mathematical model.
- The Turing model forms the basis of software development. It shows the behavior expected from the system to be designed. Using a Turing model, one can design and develop the step expected in the procedural oriented programming. The concept of procedure oriented programming will be seen in the next chapter

Syllabus Topic : Von Neumann Model

Q. Write a short note on von Neumann model.

Ans. :

von Neumann model

- The computer as seen by you can be as illustrated in the Fig. 1.1.1.
- Besides other units, it is made of a monitor (standard display unit), CPU (Central Processing Unit) and keyboard (standard input device).
- A computer can be visualized as a system with Input/Output (I/O) devices, Central Processing Unit (CPU) and the memory interconnected as shown in the Fig. 1.1.2. This is the hardware view of a computer.

Fig. 1.1.1 : Computer

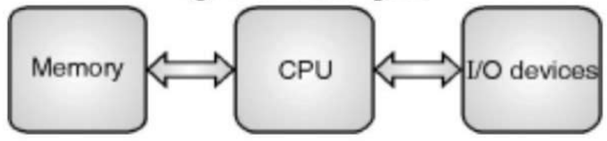


Fig. 1.1.2 : Hardware structure of a computer

- As a programmer, the computer can be seen as a programmable system. It is a system that can be programmed by the programmer.

Syllabus Topic : Basics of Positional Number System

Q. Explain how to convert a decimal number to binary with example

Ans. :

- Decimal to binary : The steps to be followed for converting a decimal number to a binary number are :
  - Divide the decimal number by 2.
  - Note down the quotient and remainder as shown in the example below.
  - Repeat the above procedure till the quotient is zero.
  - The last remainder is the MSB (Most Significant Bit i.e. the bit with highest weight) and the first remainder is the LSB (Least Significant Bit).

E.g. $(5)_{10} = (\ ?\ )_2$

$(5)_{10}$means the number 5 in decimal (base 10) number system. Base 10 means 10 different representations 0 to 9.

$(?)_2$means the number in binary (base 2) number system. Base 2 means 2 different representations 0 and 1.

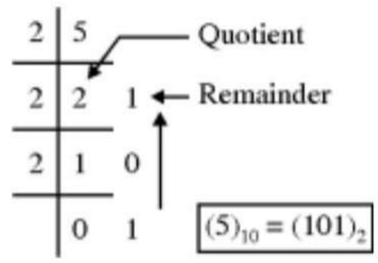


Fig. 1.3.1

Q. Explain how to convert a binary number to decimal with example

Ans. :

- Binary to decimal : The steps to be followed for converting a binary number to a decimal number are :
- Multiply the binary digits (bits) with powers of 2 according to their positional weight. The position of the first bit (going from right to left) is 0 and it keeps on incrementing as you go towards left for every bit.

E.g. $(11100)_2 = (\ ?\ )_{10}$

$= 1 \times 2^4 + 1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 0 \times 2$

$= 16 + 8 + 4 + 0 + 0$

$= (28)_{10}$

Q. Explain the AND, OR, EXOR and NOT operations on binary data with examples.

Ans. :

- There are various operations that are performed on binary data like AND, OR, EXOR and NOT. These operations are explained below.

1.AND : The output is 1 if and only if A AND B (A & B being the inputs) are 1.

A(INPUT)	B(INPUT)	Y(OUTPUT)
0	0	0
0	1	0
1	0	0
1	1	1

2.OR : The output is 1 if and only if A OR B (A & B being the inputs) is 1.

A(INPUT)	B(INPUT)	Y(OUTPUT)
0	0	0
0	1	1
1	0	1
1	1	1

3.NOT : The output is NOT A (A being the input) i.e. complement of A.

A(INPUT)	Y(OUTPUT)
0	1
1	0

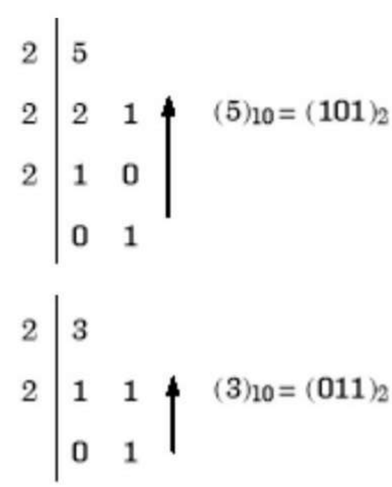
4.EXOR : The output is 1 if and only if EXCLUSIVELY A OR B (A & B being the inputs) are 1.

A(INPUT)	B(INPUT)	Y(OUTPUT)
0	0	0
0	1	1
1	0	1
1	1	0

Some examples of performing these operations on the binary data are given below :

1.5 AND 3

In 'C' programming this is written as 5 & 3.



Note : We can prefix 0's to a number in binary as in decimal. Here, $(011)_2$ is written to make it 3 digit data, as $(5)_{10}$ = $(101)_2$ is 3 digit.

Now, 5 & 3

$$\begin{array}{r} (1\,0\,1)_2 \\ \&\ (0\,1\,1)_2 \\ \hline (0\,0\,1)_2 \\ \hline \end{array}$$

Note :& is bitwise AND operator i.e. it works on each bit as shown in the above example.

$(001)_2 = 0 \times 2^2 + 0 \times 2^1 + 1 \times 2$

$= (1)_{10}$

$(5)_{10}$ & $(3)_{10} = (1)_{10}$

2.13 AND 10

$$\begin{array}{c|cc} 2 & 13 & \\ 2 & 6 & 1 \\ 2 & 3 & 0 \\ 2 & 1 & 1 \\ & 0 & 1 \end{array} \quad (13)_{10} = (1101)_2$$

$$\begin{array}{c|cc} 2 & 10 & \\ 2 & 5 & 0 \\ 2 & 2 & 1 \\ 2 & 1 & 0 \\ & 0 & 1 \end{array} \quad (10)_{10} = (1010)_2$$

Now; 13 and 10

$$\begin{array}{r} (1\,1\,0\,1)_2 \\ \&\ (1\,0\,1\,0)_2 \\ \hline (1\,0\,0\,0)_2 \\ \hline \end{array}$$

$(1000)_2 = 1 \times 2^3 + 0 \times 2^2 + 0 \times 2^1 + 0 \times 2^0 = (8)_{10}$

$(13)_{10}$ & $(10)_{10} = (8)_{10}$

3.7 OR 6

In 'C' programming, this is written as 7|6.

$$\begin{array}{c|cc} 2 & 7 & \\ 2 & 3 & 1 \\ 2 & 1 & 1 \\ & 0 & 1 \end{array} \quad (7)_{10} = (111)_2$$

$$\begin{array}{c|cc} 2 & 6 & \\ 2 & 3 & 0 \\ 2 & 1 & 1 \\ & 0 & 1 \end{array} \quad (6)_{10} = (110)_2$$

Now; 7 OR 6

$$
\begin{array}{r}
(1\,1\;\;1)_2 \\
|\;\;(1\,1\,0)_2 \\
\hline
(1\,1\,1)_2 \\
\hline
\end{array}
$$

$(111)_2 = 1 \times 2^2 + 1 \times 2^1 + 1 \times 2$

$= (7)_{10}$

$\therefore \quad (7)_{10} \mid (6)_{10} = (7)_{10}$

4.5 OR 2

$$
\begin{array}{c|cc}
2 & 5 & \\
2 & 2 & 1 \\
2 & 1 & 0 \\
 & 0 & 1
\end{array}
\qquad (5)_{10} = (101)_2
$$

$$
\begin{array}{c|cc}
2 & 2 & \\
2 & 1 & 0 \\
 & 0 & 1
\end{array}
\qquad (2)_{10} = (010)_2
$$

Now, 5 OR 2

$$
\begin{array}{r}
(1\,0\;\;1)_2 \\
|\;\;(0\,1\,0)_2 \\
\hline
(1\,1\,1)_2 \\
\hline
\end{array}
$$

$(111)_2 = 1 \times 2^2 + 1 \times 2^1 + 1 \times 2$

$= (7)_{10}$

$\therefore \quad (5)_{10} \mid (2)_{10} = (7)_{10}$

5.6 EXOR 3

In C, this is written as 6 ^ 3

$$
\begin{array}{c|cc}
2 & 6 & \\
2 & 3 & 0 \\
2 & 1 & 1 \\
 & 0 & 1
\end{array}
\qquad (6)_{10} = (110)_2
$$

$$
\begin{array}{c|cc}
2 & 3 & \\
2 & 1 & 1 \\
 & 0 & 1
\end{array}
\qquad (3)_{10} = (011)_2
$$

Now; 6 EXOR 3

$$
\begin{array}{r}
(1\,1\;\;0)_2 \\
\wedge\;\;(0\,1\,1)_2 \\
\hline
(1\,0\,1)_2 \\
\hline
\end{array}
$$

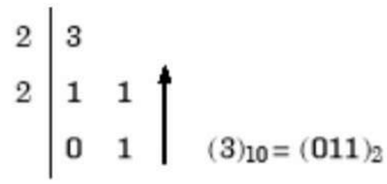
$(101)_2 = 1 \times 2^2 + 0 \times 2^1 + 1 \times 2$

$= (5)_{10}$

$\therefore \quad (6)_{10} \wedge (3)_{10} = (5)_{10}$

6.NOT 3

In C, this is written as ~ 3



Now; ~ 3 is

0 0 0 1 1

1 1 1 0 0

Note :1.The data may be of more than 2 bits and hence you will get the output depending on the size of data.

2.In computer binary language you will get this value as $(-4)_{10}$.

3.Hence if you say $\sim(10)_{10}$, you will get $-(11)_{10}$ and so on.

Syllabus Topic : Introduction to Operating System and Component of an Operating System

Q. Write a short note on OS and its components.

Ans. :

OS and its components

- Operating system is the software that you use to operate your PC (like Windows or Linux) .The code of open source operating system is made available for the user.
- Linux is an example of an open source operating system. Windows and Macintosh both are closed (proprietary) systems.
- Linux is a Unix-like computer operating system assembled under the model of free and open source softwaredevelopment and distribution.
- The defining component of Linux is the Linux kernel, an operating system kernel first released 5 October 1991 by Linus Torvalds. Bharat Operating System (BOSS) is also one of the good eg. Of open source O.S.
- Let us take a look at differences between Linux and windows.

Features	Linux	Windows
Distribution	**Open Source**	**Closed Source**
Price	Free or at a much lower price than Microsoft Windows.	Microsoft Windows can run between $50.00 - $150.00 US dollars per each license copy
Ease	It is harder to use for new computer users.	Windows is still much easier to use for new computer users.
Reliability	The majority of Linux variants and versions are notoriously reliable and can often run for months and years without the need to reboot.	It still cannot match the reliability of Linux.
Software	Linux has a large variety of available software programs, utilities, and games.	Windows has a much larger selection of available software.
Security	Linux has always been a very secure operating system.	Most vulnerable to viruses and other attacks.

Q. Explain how problem is defined with a help of suitable example. ***(May 2013)***

Ans. :

- When solving a problem we always first need to define the problem properly, so as to know what problem is exactly to be solved. For example if a shopkeeper has low sale, then the problem for this shopkeeper has to be properly defined before the owner thinks of solving this problem. If the sale is low because of the location, then his problem is location and hence the solution should be to change the place so as to increase the sale. If the problem of low sale is because of the lack of the required goods for the customer, then his attempt in solving the problem should be to procure the relevant goods etc.
- The problem statement has to be defined considering the various points like :

1. Inputs required
2. Process and
3. Outputs expected

- For example if we have to define a problem statement for the programmer to generate prime numbers. The statement "Write a program to generate prime numbers" sounds incomplete. We need to either specify the number of prime number's required or the range within which the prime numbers are required. Thus a perfect statement could be "Write a program to generate the first 'n' prime numbers, where the value of 'n' is taken from user".
- Thus the problem statement can be given as :

1. Inputs required : Value of 'n', where 'n' is the count of prime numbers required
2. Process : Check for each number to be prime or not, beginning from 2. Display the number if it is prime and keep a track of the count.
3. Output : A list of 'n' prime numbers.

Q.Determine the type of triangle, given its sides (i.e. isosceles, scalene, equilateral). Is the above problem definition complete ? If not, make this problem definition complete. ***(May 2014)***

Ans. :

- When solving a problem we always first need to define the problem properly, so as to know what problem is exactly to be solved. For example if a shopkeeper has low sale, then the problem for this shopkeeper has to be properly defined before the owner thinks of solving this problem.
- If the sale is low because of the location, then his problem is location and hence the solution should be to change the place so as to increase the sale.
- If the problem of low sale is because of the lack of the required goods for the customer, then his attempt in solving the problem should be to procure the relevant goods etc.
- The problem statement has to be defined considering the various points like :

1. Inputs required
2. Process and
3. Outputs expected

- The problem definition given in the question is not accurate and it can be completely defined as :

1.Inputs required : The lengths of three sides say, a, b and c

2.Process : Check for the values of a, b and c to decide the triangle as equilateral, isosceles or scalene.

3.Output : If all the sides are equal, display "Equilateral triangle"; if two sides are equal display "Isosceles triangle", else display "Scalene triangle".

Q.Find the roots of a quadratic equation. Is the above problem definition complete? If not, make this problem definition complete. ***(Dec. 2013)***

Ans. :

- When solving a problem we always first need to define the problem properly, so as to know what problem is exactly to be solved. For example if a shopkeeper has low sale, then the problem for this shopkeeper has to be properly defined before the owner thinks of solving this problem.
- If the sale is low because of the location, then his problem is location and hence the solution should be to change the place so as to increase the sale.
- If the problem of low sale is because of the lack of the required goods for the customer, then his attempt in solving the problem should be to procure the relevant goods etc.
- The problem statement has to be defined considering the various points like :

1. Inputs required
2. Process and
3. Outputs expected

- The problem definition given in the question is not accurate and it can be completely defined as :

1.Inputs required : The coefficients of the quadratic equation, say, a, b and c

2.Process : Check for the values of a, b and c to decide the roots are real or imaginary. Accordingly calculate the roots using the formula.

3.Output : Display the roots of the quadratic equation.

Syllabus Topic : Three Constructs of Algorithm: Sequence, Decision (Selection) and Repetition

Q. What do you mean by algorithm? Which points you should consider while developing the algorithm ? ***(Dec. 2013, May 2014, May 2016)***

Ans. :

Algorithm :

- An algorithm is a finite set of statements, each of which has a clear meaning and can be executed in a finite amount of time and with a finite amount of effort. Thus whatever is the size of input data, the algorithm can solve the given problem in finite amount of time.
- The following points should be considered while developing an algorithm :

(i)Non-ambiguity : This property of an algorithm indicates that each of the statement in the algorithm must be clear and precise. There must be no ambiguity in any of the statement.

(ii)Range of Input : The range of the input for which the algorithm works is also to be compulsorily mentioned in the algorithm. There should be clear indication for the range of inputs for which the algorithm may fail.

(iii)Multiplicity : The algorithm can be represented in multiple ways. An algorithm can be written in English language or by the graphical representation called as flowchart or the Pseudo code Algorithm.

(iv)Speed : The one of the important property of an algorithm is that it should produce the result at a fast speed or efficiently.

(v)Finiteness : The algorithm should be finite i.e. there should be no infinite condition leading to a never ending procedure and hence never completing the task.

➢Algorithm 1.6.1 : Write an algorithm to display a statement "Hello Friend".

- **Step-form Algorithm**

Step I:Display the sentence "Hello Friend".

Step II:Stop

- Pseudo code Algorithm

Step I:START

Step II:PRINT "Hello Friend".

Step III:STOP

➢Algorithm 1.6.2 :Write an algorithm to accept a number and display its square.

- **Step-form Algorithm**

Step I :Indicate the user to enter the input number by displaying suitable sentence.

Step II :Wait for the user to enter the number.

Step III:Calculate the square of the user entered number.

Step IV:Display the calculated result.

Step V:Stop

- Pseudo code Algorithm

Step I :START

Step II:PRINT "Enter a number".

Step III :INPUT n.

Step IV:a = n * n.

Step V:PRINT a.

Step VI:STOP

➢Algorithm 1.6.3 :Write an algorithm to accept two numbers and display its product.

- **Step-form Algorithm**

Step I:Indicate the user to enter the input numbers by displaying suitable sentence.

Step II:Wait for the user to enter the two numbers.

Step III:Calculate the product of the user entered numbers.

Step IV:Display the calculated result.

Step V:Stop

- Pseudo-code Algorithm

Step I:START

Step II:PRINT "Enter two numbers".

Step III:INPUT a,b.

Step IV:x = a * b.

Step V:PRINT x.

Step VI:STOP

➢Algorithm 1.6.4 :Write an algorithm to calculate and display the area and perimeter of a rectangle.

- Step-form Algorithm

Step I:Indicate the user to enter the length and breadth of the rectangle as input from user by displaying suitable sentence.

Step II:Wait for the user to enter the input.

Step III:Calculate the area and perimeter using the corresponding formulae i.e. $2(l + b)$ is perimeter and $(l \times b)$ is area.

Step IV:Display the calculated area and perimeter.

Step V:Stop

- Pseudo code Algorithm

Step I:START

Step II:PRINT "Enter the length and breadth of the rectangle".

Step III:INPUT l, b.

Step IV:a = l * b

Step V:p = 2 (l + b)

Step VI:PRINT a, p.

Step VII:STOP

➢Algorithm 1.6.5 :Write an algorithm to separate the digits of a three digit number and display the three digits separately.

- **Step-form Algorithm**

Step I :Indicate the user to enter a three digit number by displaying suitable sentence.

Step II :Wait for the user to enter the input.

Step III :Get the last digit of the user entered number into a variable using the modulus operator (or remainder after division by 10). And the quotient operator to get the remaining part of the number into the same variable as that of the original number.

Step IV :Repeat the step III to get the remaining two digits.

Step V :Display the separated digits.

Step VI :Stop

- Pseudo code Algorithm

Step I:START

Step II:PRINT "Enter a three digit number".

Step III:INPUT n.

Step IV:d1 = n mod 10

STEP V:d2 = (n / 10) mod 10

STEP VI:d3 = n / 100

Step VII:PRINT d1, d2, d3

Step VIII:STOP

➢Algorithm 1.6.6 : Write an algorithm to swap two numbers.

- **Step-form Algorithm**

Step I:Indicate the user to enter two numbers by displaying suitable sentence.

Step II:Wait for the user to enter the input.

Step III:Using a temporary variable swap the two numbers. The logic to swap the numbers is shown below :

i.Put the first number in the temporary variable.

ii.Put the second number in the first variable.

iii.Put the temporary number in the second variable

Step IV:Display the two numbers indicating the two numbers are swapped.

Step V:Stop

- Pseudo code Algorithm

Step I:START

Step II:PRINT "Enter two numbers".

Step III:INPUT a, b.

Step IV:temp = a.

Step V:a = b

Step VI:b = temp

Step VII:PRINT a , b.

Step VIII:STOP

➢Algorithm 1.6.7 :Write an algorithm to display the word “Computer” five times.

- **Step-form Algorithm**

Step I:Initialize a variable as a counter with the initial value as 1.

Step II:Check if the counter variable is greater than 5, if yes goto step VI

Step III:Display the word "Computer".

Step IV:Increment the counter.

Step V:Goto step II

Step VI:Stop.

- Pseudo code Algorithm

Step I:START

Step II:i = 1.

Step III:IF i > 5 THEN GOTO step VII

Step IV:PRINT "Computer".

Step V:i = i + 1.

Step VI:GOTO step III

Step VII:STOP.

➢Algorithm 1.6.8 :Write an algorithm to display the first ten natural numbers.

- **Step-form Algorithm**

Step I:Initialize a counter with the value as 1

II:Check if the counter is greater than 10, if yes, then goto step VI.

III:Display the value of the counter variable.

IV:Increment the counter variable

V:Goto step II

Step VI:Stop.

- Pseudo code Algorithm

Step I:START

Step II:i = 1

III:IF i > 10 THEN GOTO step VII.

IV:PRINT i.

V:i = i + 1

VI:GOTO step III

Step VII:STOP.

➢Algorithm 1.6.9 : Write an algorithm to display the first n natural numbers, where the value of n is taken from user

- **Step-form Algorithm**

Step I:Indicate the user to enter a number by displaying suitable sentence.

Step II:Wait for the user to enter the input.

Step III:Initialize a counter with the value as 1.

Step IV:Check if the counter is greater than the user entered variable, if yes,

then goto step VIII.

Step V:Display the value of the counter variable.

Step VI:Increment the counter variable

Step VII:Goto step IV

Step VIII:Stop.

- Pseudo code Algorithm

Step I:START

Step II:PRINT "Enter a number".

Step III:INPUT n.

Step IV:i = 1.

Step V:IF i > n THEN GOTO step IX.

Step VI:PRINT i.

Step VII:i = i + 1.

Step VIII:GOTO step V

Step IX:STOP.

➢Algorithm 1.6.10 :Write an algorithm find the factorial of a number.

- **Step-form Algorithm**

Step I:Indicate the user to enter a number by displaying suitable sentence.

Step II:Wait for the user to enter the input.

Step III:Initialize a counter with the value as 1 and a "fact" variable as 1

Step IV:Check if the counter is greater than the user entered variable, if yes,

then goto step VIII.

Step V:Multiply the "fact" variable with the counter value.

Step VI:Increment the counter variable

Step VII:Goto step IV

Step VIII:Display the value of the factorial i.e. the variable "fact"

Step IX:Stop.

- Pseudo code Algorithm

STEP I:START

Step II:PRINT "Enter a number".

Step III:INPUT n.

Step IV:i = 1, fact = 1

Step V:IF i > n THEN GOTO step IX.

Step VI:fact = fact * i.

Step VII:i = i + 1

Step VIII:GOTO step V

Step IX:PRINT fact

Step X:STOP

➢Algorithm 1.6.11 :Write an algorithm to check if the entered number is prime number or not.

- **Step-form Algorithm**

Step I:Indicate the user to enter the input number by displaying suitable sentence.

Step II:Wait for the user to enter the number to be checked.

Step III:Initialize a divisor variable with the value as 2.

Step IV:Check if the user entered variable is divisible by the divisor variable, if yes, then goto step VII.

Step V:Increment the divisor variable.

Step VI:Goto step IV

Step VII:If the user entered number is equal to the current divisor variable value then display that the number entered by the user is a prime number; else display that the user entered number is not a prime number

Step VIII:Stop.

- Pseudo code Algorithm

Step I:START

Step II:PRINT "Enter a number".

Step III:INPUT n.

Step IV:x = 2.

Step V:IF (n mod x) = 0 THEN GOTO step VIII

Step VI:x = x + 1.

Step VII:GOTO step V

Step VIII:IF x = n THEN PRINT "Prime number"

ELSE PRINT "Not a prime number"

Step IX:STOP.

➢Algorithm 1.6.12 :Write an algorithm to check if the year entered is leap year or not.

Note : A normal year is said to consist of 365 days. But the actual time required for Earth to revolve around the sun is 365.242199 days. Hence to average it out a day is added in every fourth year which gives average of 365.25 days per year. To reach more closer to the above mentioned time, every 100th year isnota leap year and every 400 yearsis leap year. This brings theaverage time for a yearto be 365.2425 days almost closest.

- Step-form Algorithm

Step I :Indicate the user to enter a number by displaying suitable sentence.

Step II:Wait for the user to enter the input.

Step III:Check if the user entered number is a leap year by the necessary condition. (The condition to be checked is that the year should not be divisible by 100 and divisible by 4 or divisible by 400). If yes, then display that the year is a leap year; else display that the year is not a leap year.

Step IV:Stop.

- Pseudo code Algorithm

STEP I:START

Step II :PRINT "Enter a year number".

Step III:INPUT y.

Step IV:IF (y mod 4 = 0 AND y mod 100 ≠ 0) OR (y mod 400 = 0) THEN

PRINT "It is a leap year"

ELSE PRINT "Not a leap year"

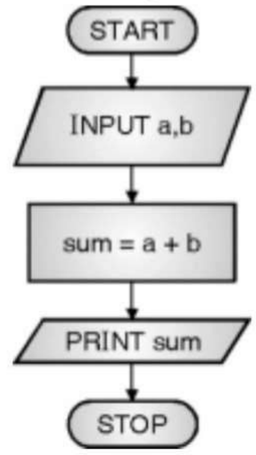
Step V:STOP.

Syllabus Topic : Three Constructs of Flowchart : Sequence, Decision (Selection) and Repetition

Flowchart 1 : Draw a flowchart to add two numbers.

Solution :

- As shown in the Flowchart 1, initially two numbers are accepted from user, this is shown using a parallelogram. The two numbers are then added, shown using a rectangle. The result is displayed, again shown using parallelogram. The start and stop are shown using oval.

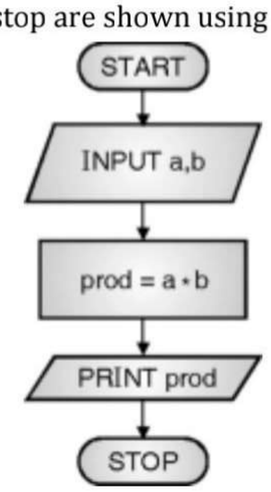


Flowchart 1

Flowchart 2 : Draw a flowchart to find the product of two numbers.

Solution :

- As shown in the Flowchart 2, initially two numbers are accepted from user, this is shown using a parallelogram. The two numbers are then multiplied, shown using a rectangle. The result is displayed, again shown using parallelogram. The start and stop are shown using oval.

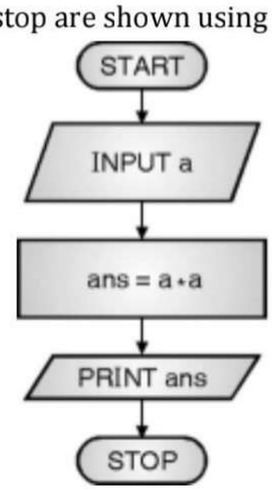


Flowchart 2

Flowchart 3 : Draw a flowchart to find the square of a numbers.

Solution :

- As shown in the Flowchart 3, initially one number is accepted from user, this is shown using a parallelogram. The number is then multiplied with itself, shown using a rectangle. The result is displayed, again shown using parallelogram. The start and stop are shown using oval.

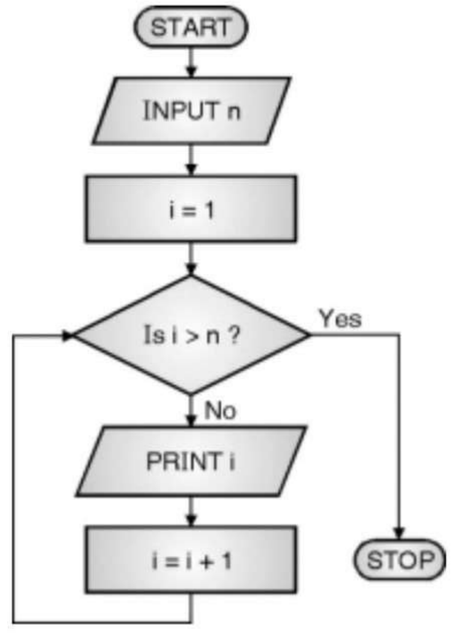


Flowchart 3

Flowchart 4 : Draw a flowchart to display the first 'n' natural numbers.

Solution :

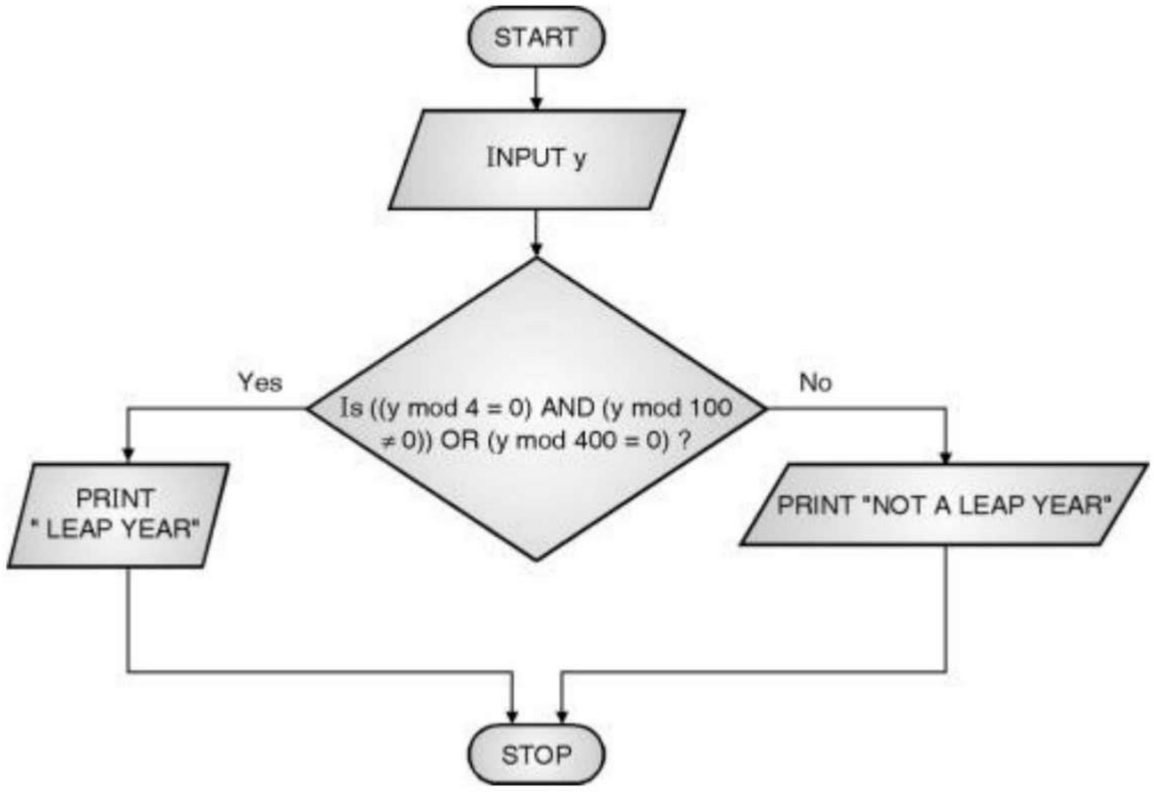
- In Flowchart 4 again the value of a variable say 'n' is taken from user. Another counter variable is initialized to the value 1.
- The value of the counter variable is checked. If the counter variable value is less than or equal to the value of variable 'n'; then the counter variable value is displayed and is incremented. Then the counter variable value is again checked. This process keeps on repeating until the counter value crosses the value of 'n', after which the algorithm stops.



Flowchart 4

Flowchart 5 : Draw a flowchart to check if the year entered by user is leap year or not.

Solution :



Flowchart 5

- In Flowchart 5, again the year is taken as input from user with a parallelogram. The decision box checks the year is a leap year or not by the proper condition. If the condition is true, it prints that the year is a leap year else it is not a leap year.

Q. For a problem of sorting numbers in ascending order, in which cases time complexity is calculated. ***(May 2013)***

Ans. :

Time Complexity (Efficiency)

- The time complexity of an algorithm is the time required by the computer to execute the program implemented according to the corresponding algorithm.
- The calculation of time required to execute the program is slightly complicated and difficult as the time required to execute the program depends on various other parameters besides the algorithm implementation.
- The different parameters on which the execution time depends are instruction set used, other programs running in the computer, hardware or processor speed, etc.
- The time complexity is hence given in terms of frequency where the frequency is the number of times the instructions are to be executed. For example if the instructions are to be executed for 'n' times (where 'n' is the input size or number of input values) then the time complexity is said to be 'n'.

1.Worst Case Condition (Big O notation)

- Based on the input, the time required to execute the program for the given algorithm keeps on changing. The input that requires maximum time for execution is called as the worst case condition and the notation used for this is called as the Big O notation.
- This is the most important parameter considered for evaluating the efficiency of the algorithm. For example if there are some numbers to be arranged in ascending order and they are given as input in exactly descending order, then the time required for implementing the algorithm for this case of input is called as the worst case condition.
- Thus an algorithm that requires the execution of the instructions for 'n' times (where 'n' is the given input data set size), then it is represented as O(n).
- If an algorithm requires the execution of the instructions for twice as the value of 'n', then it is given as : O(2n).
- If an algorithm requires the execution of the instructions for 'n' times 'n', then it is given as : $O(n^2)$.
- If an algorithm requires the execution of the instructions for the cube of 'n' times, then it is given as : $O(n^3)$.
- Thus the conditions mentioned above using big O notation indicate that they are the worst case conditions.

2.Average Case Condition (Ω notation)

- The average case condition is when the time required to execute the implementation of the algorithm is average i.e. mid-way.
- Let us consider the same example i.e. for sorting the numbers in ascending order. In this case some of the input data is already sorted, while some requires to be sorting.
- One can say that in this case the time required will be such that it will be average of the time required for different sets of data, or the data arranged in different possible ways. This representation is given by Omega (Ω)
- Thus an algorithm that requires the execution of the instructions for 'n' times (where 'n' is the given input data set size), then it is represented as Ω(n).

- If an algorithm requires the execution of the instructions for twice as the value of 'n', then it is given as : $\Omega(2n)$.
- If an algorithm requires the execution of the instructions for 'n' times 'n', then it is given as : $\Omega(n^2)$.
- If an algorithm requires the execution of the instructions for the cube of 'n' times, then it is given as : $\Omega(n^3)$.
- Thus the conditions mentioned above using Ω notationindicate that they are the average case conditions.

3.Best Case Condition (θ notation)

- The best case condition is when the time required to execute the implementation of the algorithm is best.
- Let us consider the same example for sorting the numbers in ascending order. In this case if all the input data is already sorted then the time required to execute the implementation of the algorithm will be minimum and hence is considered as the best case condition. This is represented by θ·
- As seen for average and worst case conditions, we can similarly have the following statements
- Thus an algorithm that requires the execution of the instructions for 'n' times (where 'n' is the given input data set size), then it is represented as $\theta(n)$.
- If an algorithm requires the execution of the instructions for twice as the value of 'n', then it is given as : $\theta(2n)$.
- If an algorithm requires the execution of the instructions for 'n' times 'n', then it is given as : $\theta(n^2)$.
- If an algorithm requires the execution of the instructions for the cube of 'n' times, then it is given as : $\theta(n^3)$.
- Thus the conditions mentioned above using θ notation indicate that they are the best case conditions.

❑❑❑

CHAPTER 2

Fundamentals of C-Programming

Syllabus Topic : Character Set

Q.What is character set ? List the character set.

Ans. :

Character set

- The character set of any programming language indicates the different characters the program can contain. Character set includes all the alphabets, digits and special symbols supported by the processor.
- When we will be writing C programs, all these will be found in our program. Table 2.6.1 gives a list of C character set.

Table 2.6.1 : Character Set of C

Sr. No.	Characters	List included
1.	Alphabets (Upper case and lower case)	A, B, C .....Z a, b, c, ...z
2.	Digits (numbers)	0,1,2,....9
3.	Special symbols (all those seen on a keyboard, nothing besides that)	< > { } ( ) [ ] , . ; : ! ? ' " / + * = % & # @ | \ ~ ` $ ^ _ - (The names used for these symbols are given in the table 2.8.2)
4.	Other special characters	Blank Space, Tab, Carriage Return (Enter Key)

Syllabus Topic : Keywords

Q.What are keywords ? List the keywords.

Ans. :

Keywords

- Keywords are some special words that have a predefined meaning for the C compiler. Hence, these words cannot be used as identifiers.
- These are a set of words which are reserved for the certain operations and hence are also sometimes referred as reserved words.
- All keywords are in lower case.
- The keywords used in C are as given below :

auto	else	long	switch
break	enum	register	typedef
case	extern	return	union
char	float	short	unsigned
const	for	signed	void
continue	goto	sizeof	volatile
default	if	static	while
do	int	struct	_Packed
double			

- These keywords will be used in different places in programming. These keywords include all those keywords also as declared by ANSI (American National Standards Institute) C. This institute has published the standards to be used in C programming language.

Syllabus Topic : Identifiers

Q.What is an identifier ? What are the rules to make identifier ?

Ans. :

Identifier

- Identifiers are names given to different user defined things like variables, constants, functions, classes, objects, structures, unions, etc. While making these identifiers we need to follow some rules.
- These rules are stated below :

1.The identifier can consist of alphabets, digits and a special symbol i.e. '_' (underscore).

2.An identifier cannot start with a digit. It can start either with an alphabet or underscore.

3.It cannot contain any special symbol except underscore. Blank spaces are also not allowed.

4.It cannot be a keyword.

5.It is case sensitive i.e. an alphabet capital in one identifier with same name in another identifier with that alphabet small case will be considered different (for more details see examples in this section).

6.Earlier; there was a limit of the length of the identifier to be 32 characters, but now this limit is removed. Hence, an identifier can be as long as required and minimum of one character.

Syllabus Topic : Data Types

Q. List the various types of data types.

Ans. :

Data types

- The data type decides the type of data and the memory locations required for storing that type of data in the memory.
- The data types of C can be divided into three types : Primitive, Derived and User defined data types.
- The different primitive types of data that can be used in C are integer, character, fraction type numbers, etc.
- Table 2.6.3 shows the different primitive data types and the memory space required for storing them.

Table 2.6.3 : Data types

Sr. No.	Data type	Type of data to be stored	Range	Space required in memory (bytes)
1.	char	1 character in ASCII form	– 127 to 128	1
2.	signed char	1 character in ASCII form	– 127 to 128	1
3.	unsigned char	1 character in ASCII form	0 to 255	1
4.	int	Integer nos.	– 32768 to 32767	2
5.	signed int	Integer nos.	– 32768 to 32767	2
6.	unsigned int	Integer nos.	0 to 65535	2
7.	short int	Integer nos.	– 32768 to 32767	2
8.	long int	Integer nos.	– 2147483648 to 2147483647	4
9.	signed short int	Integer nos.	– 32768 to 32767	2
10.	signed long int	Integer nos.	– 2147483648 to 2147483647	4
11.	unsigned short int	Integer nos.	0 to 65535	2
12.	unsigned long int	Integer nos.	0 to 4294967296	4
13.	float	Fraction nos.	$3.4e^{-38}$ to $3.4e^{38}$	4
14.	double	Fraction nos.	$1.7e^{-308}$ to $1.7e^{308}$	8
15.	long double	Fraction nos.	$3.4e^{-4932}$ to $1.1e^{4932}$	10

Syllabus Topic : Constants and Variables

Q. Define constants and variables.

Ans. :

Constants

- Constants are values given to the identifiers that do not change their values throughout the execution of the program.
- Constants can be defined either by writing the keyword const before the data type or by using #define.

Variables

- Variables are values given to identifiers that can change their values during the execution of the program.

Syllabus Topic : Structure of a C program

➢Program 2.7.1 : Write is a simple program to display a statement “Hello Friend”.

- Step-form Algorithm

Step I :Display the sentence "Hello Friend" using printf() function

Step II :Wait for user to press a key using getch() function.

Step III :Stop

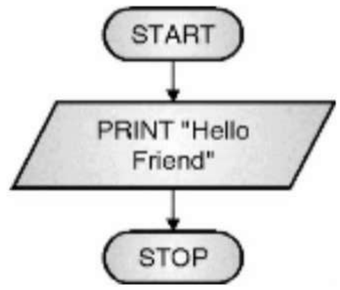
- Pseudo code Algorithm

STEP I:START

Step II :PRINT "Hello Friend"

Step III :STOP

- Flowchart 1



Flowchart 1

➢Program

```
//This is a simple program to display a statement
// "Hello Friend"
#include<conio.h>
#include<stdio.h>
void main ()
{
printf("Hello Friend");
getch();
}
```

Output

```
Hello Friend
```

Syllabus Topic : Preprocessor

Q.Explain Integrated Development Environment.

Ans. :

Integrated Development Environment

- IDE is a toolkit to write and execute a program. The most widely used IDE with the C programs is the Turbo C IDE.
- This IDE contains of five tools namely : Editor, Preprocessor, Compiler, Linker and Loader.
- Editor : It is used to type and edit the program.
- Preprocessor : It is responsible for performing certain operations before the compiling, like including header files, declaring constants, etc. It always begins with the '#' sign.

For e.g.#include<stdio.h>

#define pi 3.14;

- Compiler : It is used to convert the C program to machine language. As the processor cannot understand the language of C, we need to convert it into a machine understandable language i.e. machine language.
- Linker : It is responsible for linking (connecting) the different functions in a program. We may write multiple functions in the program or conditional/loop operations in our program. These kinds of operations are to be handled by the linker.
- Loader : It loads the binary file from the memory and gets is executed by the processor. The binary file that the loader loads is the machine language program understood by the processor.
- The above program 2.7.1 in section 2.7 can be executed using the following steps :

1.Double click TC.exe to get the editor window.

2.Type the program in the editor window.

3.Save the program as .cpp for e.g. prog101.cpp.

4.Compile the program by pressing Alt + F9 i.e. press the Alt and F9 key together.

5.Execute the program by pressing Ctrl + F9

6.See the output on the screen and press enter to come back to editor window.

Syllabus Topic : Operators

Q.Write short note on operators.

Ans. :

Operators

- The operators are one of the tokens of C. There are different types of operators classified based on their functions.
- Many of these operators although listed here, but will be understood in the subsequent chapters as and when we use them. So you may just go through the operators and understand them later while programming.
- The data on which the operation is performed are called as operands. If an operator requires one operand, it is called as Unary operator. If an operator requires two operands, then it is called as Binary operator and if it requires three operands, it is called as Ternary operator.

Syllabus Topic : Unary Operators

Q.Explain unary operators.

Ans. :

Unary operators

1.Unary minus (–)

- The symbol shown in the bracket is used as unary minus operator.
- It returns the negative value of the variable to which it is preceded.
- For e.g. if x = 3 and y = 6 then

y = – x;

will make the value of y as – 3.

2.Casting Operator (( )) or Type Conversion

- It is many a times required to convert one data type to another.

- The casting operator is used for type conversion i.e. it can be used to convert a data of one type to another data type.

For e.g. if we have int x=3;

float y=5.6;

then the statement, x=(int) y;

will result in the value of x as 5.

We will see the use of this operator in the programming.

3.Logical Not operator (!)

- The symbol shown in the brackets i.e. the exclamation mark is used as a logical not operator. It is used to check certain conditions in a condition statement. It performs the logical not of the result obtained from the expression on its right.
- For e.g. if x=1, then y=!x;

will result with y having the value 0.

- We will see some more logical not operator based expressions and program followed by this section.

4.Address of operator (&)

- This operator returns the address of the variable associated with it. It will be studied more deeply in the chapter on Pointers.
- For e.g. if we write y=&x;

then the memory address allocated to the variable x will be copied into the variable y. For this the variable y must be a pointer variable.

5. Indirection operator or value of operator (*)

- This operator returns the value of the data at the given address. This operator will also be studied in more details in the chapter on pointers.
- For e.g. z=*y;

will give the value of the data stored at the address given by y.

6.Scope Resolution operator (::)

- It resolves the scope of a variable to be inside the function or outside the function. This operator will also be studied in a later chapter.
- For e.g. if an external variable x is to be accessed then the same can be accessed using the scope resolution operator by the statement, y=::x;

7. Size of operator

- This operator is used to know the size of a variable as required to store its value in memory. It can also be used to find the size of a data type.
- The space required to store different data type is different ranging from 1 byte to 10 bytes.
- For e.g. the statement sizeof(int);

will return the value 2, as int requires 2 bytes.

- Another e.g.: int x,y;

char a;

x=sizeof(a);

y=sizeof(float);

then x will have 1, and y will have 4

8.Bitwise not operator (~)

- This operator is used to perform bitwise NOT operation. We have also seen some examples of NOT operations.
- The bitwise NOT operator can be used to perform binary NOT operation. This operation can be performed by using the operator given in brackets i.e. ~.
- For e.g. If x=3,

then y=~x;

will result in y having a value of – 4

9.Increment Operator (++)

- It returns the value of the variable added with one and stores the result in the variable itself.
- For e.g. if x=5,

then x++;

will make the value of x equal to 6.

- This "x++;" (also called as post increment operator) can also be written as "++x;" (also called as pre increment operator).
- It can also be used to store the result in another variable. But in this case the post increment and pre increment statements will have different behavior as explained below with examples.
- In post increment case, for e.g. if x=5,

then y=x++;

will make the value of y equal to 5 and x equal to 6.

As the name says post increment, it first gives the previous value and then increments.

- In pre increment case, for e.g. if x=5,

then y=++x;

will make the value of y equal to 6 and x equal to 6.

As the name says pre increment, it first increments and then gives the incremented value.

- More such examples will be seen with programs.

10.Decrement Operator(–)

- It returns the value of the variable subtracted with one and stores the result in the variable itself.

  For e.g. if x=5,

then x–;

will make the value of x equal to 4.

- This "x–;" (also called as post decrement operator) can also be written as "–x;" (also called as pre decrement operator).

- It can also be used to store the result in another variable. But in this case the post decrement and pre decrement statements will have different behavior as explained below with examples.
- In post decrement case, for e.g. if x=5,

then y=x–;

will make the value of y equal to 5 and x equal to 4.

As the name says post decrement, it first gives the previous value and then decrements.

- In pre decrement case, for e.g. if x=5,

then y=–x;

will make the value of y equal to 4 and x equal to 4.

As the name says pre decrement, it first decrements and then gives the decremented value.

Syllabus Topic : Arithmetic Operators, Bitwise Operators, Logical Operators, Relational Operators

Q.Briefly explain various binary operators.

Ans. :

The various binary operators are as follows :

- Operators that require two operands are called as binary operators.
- These operators are further classified into various types namely the Arithmetic operators, Logical operators, Bitwise operators and Relational operators.

I)Arithmetic operators

- This set includes the basic arithmetic operators to perform basic arithmetic operations like addition, subtraction, multiplication and division. There are five operators in this set. They are :

  1.* to find the product

  2./ to find the quotient after division

  3.%to find the remainder after division

  4.+ to find the sum

  5.–to find the difference

- One important thing to be noted here is that the '/ ' operator returns only the quotient, while the '%' operator (also called as MOD operator) returns the remainder after division.
- The sign of the remainder is always the same as that of the dividend.
- MOD operator is possible only for int or char type of data. It doesn't work on float and double type of data.
- For example of each of these operators

  1.2 * 2 = 4;

  2.For int type of data, 5 / 3 =1;

  For float type of data, 5 / 3 = 1.67

  3.5 % 3 =2;

  4.2 + 2 = 4;

  5.3 – 2 = 1;

II)Logical operators

- Logical operators follow the same truth table as for the bitwise operators; but they are used to check conditions instead of performing operations on a data.
- The logical operators are AND and OR. The symbols used for these operators in C are && and || respectively.
- For example a statement

y > 5 && y < 10;

will result in "true" i.e. 1 if the value of y is greater than 5 AND less than 10, else it will result in false i.e. 0.

- Another example a statement

  y > 5 || y==2;

will result in "true" i.e. 1 if the value of y is greater than 5 OR equal to 2, else it will result in false i.e. 0.

- Logical operators will be understood in more details with the expressions and program examples followed by this section.

III)Relational operators

- The relational operators are used to test the relation between two variables or a variable and constant.
- The operators like '<' (less than) and '>' (greater than) used in the above examples are relational operators. We have seen the example also of the "==" (equality operator) in the above logical operators.
- A list of all the relational operators is given below :

1.==used to check if the two things are equal.

2.!=used to check if the two things are not equal.

3.<used to check if the first data is less than the second one.

4.> used to check if the first data is greater than the second one.

5.<=used to check if the first data is less than or equal to the second one.

6.>= used to check if the first data is greater than or equal to the second one.

Q.Explain with example bitwise operators in C language. ***(Dec. 2014, May 2015, Dec. 2015)***

Ans. :

Bitwise operators

- These operators work bitwise on a data.
- They perform different operations on bits of a data like AND, OR, EXOR and NOT.
- The operators are listed below :

1.~to perform bitwise NOT operation.

2.&to perform bitwise AND operation.

3.|to perform bitwise OR operation.

4.to perform bitwise EXOR operation.

5.<<to perform bitwise left shift operation.

6.>>to perform bitwise right shift operation.

- These operators are use to perform bitwise binary operations on the data.

- As we have addition, subtraction operations in decimal data for performing arithmetic operations; similarly AND, OR, NOT are basic bitwise operations on the binary data.
- The result of these operations can be understood from the following examples :

1. 5 & 3 = 1

$$\begin{array}{rcl} (5)_{10} & = & (0\ 1\ 0\ 1)_2 \\ (3)_{10} & = & (0\ 0\ 1\ 1)_2 \\ \hline & & (0\ 0\ 0\ 1)_2 = (1)_{10} \end{array}$$

2. 12 | 9 = 13

$$\begin{array}{rcl} (12)_{10} & = & (1\ 1\ 0\ 0)_2 \\ (9)_{10} & = & (1\ 0\ 0\ 1)_2 \\ \hline & & (1\ 1\ 0\ 1)_2 = (13)_{10} \end{array}$$

3. 8 ^ 10 = 2

$$\begin{array}{rcl} (8)_{10} & = & (1\ 0\ 0\ 0)_2 \\ (10)_{10} & = & (1\ 0\ 1\ 0)_2 \\ \hline & & (0\ 0\ 1\ 0)_2 = (2)_{10} \end{array}$$

4. ~7 = – 8

$$\begin{array}{rcl} (7)_{10} & = & (0\ 1\ 1\ 1)_2 \\ \hline & & (1\ 0\ 0\ 0)_2 = (-8)_{10} \end{array}$$

(According to C, wherein it will take more no. of bits and hence -8 will be the result). Hence; in general the ~x is always equal to – (x+1).

5.10 << 2 = 40

Assuming the data to be char i.e. 8 bit data

$(10)_{10} = (0\ 0\ 0\ 0\ 1\ 0\ 1\ 0)_2$

After shifting left once$(0\ 0\ 0\ 1\ 0\ 1\ 0\ 0)_2$

After shifting left for the second time$(0\ 0\ 1\ 0\ 1\ 0\ 0\ 0)_2 = (40)_{10}$

Note : When shifting to left, each of the bit is shifted left. The first bit is lost and the last bit is inserted as 0.

6.13 >> 3 = 1

Assuming the data to be char i.e. 8 bit data

$(13)_{10} = (0\ 0\ 0\ 0\ 1\ 0\ 1\ 1)_2$

After shifting left once$(0\ 0\ 0\ 0\ 0\ 1\ 0\ 1)_2$

After shifting left for the second time$(0\ 0\ 0\ 0\ 0\ 0\ 1\ 0)_2$

After shifting for the third time$(0\ 0\ 0\ 0\ 0\ 0\ 0\ 1)_2 = (1)_{10}$

Note : When shifting to right, each of the bit is shifted right. The first bit is inserted as 0 and the last bit is lost.

Syllabus Topic : Conditional Operators

Q.Explain ? : operator in C. ***(May 2016)***

Ans. :

Ternary/Conditional Operator

- An operator that requires three operands is called as a ternary operator.
- There is only one ternary operator in C. This operator is used to check a condition and accordingly do one of the two things based on the condition being true or false.
- The syntax (way of writing) of this operator is as given below:

(condition) ? <value if condition is true> : <value if condition is false>;

- Hence; as shown in the syntax, first the condition is to be written in brackets followed by a question mark (?). Then the operation that is to be performed if condition is true and then a colon (:) followed by the operation to be performed if the condition is false.
- For example :

z = (x>y)?x:y;

This statement will put the value of x into z if the given condition i.e. x > y is true. Else; the value of y will be put into z. Hence, the value of the greater variable will be put into z.

- A slightly complicated use of this operator is to find the greatest of three numbers as shown in the example.

g=(x>y)?((x>z)?x:z)):((y>z)?y:z);

This statement will give the largest of x, y and z into the variable g.

Syllabus Topic : Assignment Operators

Q.List various assignment operators in C.

Ans. :

Assignment Operators

- These operators are used to assign the value of the expression or variable on the right of the assignment operator to the variable on its left.
- The simple assignment operator is '='. But there are some more assignment operators called as composite assignment operators.
- The different assignment operators are as listed below :

1.=:This operator assigns the value of the expression or variable on its right to the variable on its left. For e.g. y=x+2;

2.+=:This operator adds the variable on its left and right and the result is put into the variable on its left. For e.g. y+=x; is same as y=y+x;

3.=:This operator subtracts the variable on its right from the variable on its left and the result is put into the variable on its left. For e.g. y – = x; is same as y = y – x;

4.*=: This operator multiplies the variable on its left and right and the result is put into the variable on its left. For e.g. y*=x; is same as y=y*x;

5./=:This operator divides the variable on its right from the variable on its left and the result is put into the variable on its left. For e.g. y/=x; is same as y=y/x;

6.%=:This operator finds the remainder by dividing the variable on its right from the variable on its left and the result is put into the variable on its left. For e.g. y%=x; is same as y=y%x;

7.&=:This operator ANDs the variables on its left and right and the result is put into the variable on its left. For e.g. y&=x; is same as y=y&x;

8.|=:This operator ORs the variables on its left and right and the result is put into the variable on its left. For e.g. y|=x; is same as y=y|x;

9.^=:This operator EXORs the variables on its left and right and the result is put into the variable on its left. For e.g. y^=x; is same as y=y^x;

10.<<=:This operator shifts in left direction the variable on its left for the number of times indicated by the variable or value on right and the result is put into the variable on its left. For e.g. y<<=x; is same as y=y<<x;

11.>>=:This operator shifts in right direction the variable on its left for the number of times indicated by the variable or value on right and the result is put into the variable on its left. For e.g. y>>=x; is same as y=y>>x;

Syllabus Topic : Comma and other Operators

Q.Write short note on precedence and associativity of operatorsin C.

Ans. :

Precedence and Associativity of Operators

- The precedence of the operators means the sequence in which the operators will be operated on, in case of multiple operators in a statement i.e. which operator will be executed first and which operator will be executed later.
- The associativity of operators refers to the direction in which the operation will be performed in case of equal precedence operators i.e. if multiple additions are there in a statement then it will be performed from left to right. We will see more about this in the examples followed by this section.
- The precedence and associativity table is as given in Table 2.10.1.

Table 2.10.1 : Precedence and Associativity of operators

Precedence	Operators	Operation	Associativity
1.	() [] -> .	Function call Array element select Structure element select Structure element select	Left to right
2.	+ – ++ – – ! ~ * & sizeof ()	Unary plus Unary minus Increment operator Decrement operator Logical NOT Bitwise NOT Pointer value of operator Pointer address of operator Size of a variable or data type Type cast operator	Right to left
3.	* / %	Multiply Divide and get quotient Divide and get remainder	Left to right
4.	+ –	Add Subtract	Left to right
5.	<< >>	Left shift operator Right shift operator	Left to right
6.	< <= > >=	Less than Less than or equal to Greater than Greater than or equal to	Left to right
7.	== !=	Equality Inequality	Left to right
8.	&	Bitwise AND	Left to right
9.	^	Bitwise EXOR	Left to right
10.	|	Bitwise OR	Left to right

Syllabus Topic : Expression and Statements

Q.Write the C assignment expressions for the following :

1.A is greater than B and greater than C

2.A is either less than B or greater than C

3.Side is equal to square root of ($a^2 + b^2 + 2ab$)

4. $a = \dfrac{xy + z\left(\frac{x}{y}\right) + zy}{x + y + z}$

5. $x = \dfrac{-b \pm \sqrt{b^2 - 4ac}}{2a}$

6.z = a +

Ans. :

1.(A > B) && (A > C)

2.(A < B) || (A > C)

3.Side = pow ((a * a + b * b + 2 * a * b), 0.5)

4.a = x(* y + z * (x / y) + z * y) / (x + y + z);

5.x_1 = (– b + pow ((b * b – 4 * a * c) , 0.5))/(2 * a);

x_2 = (– b – pow ((b * b – 4 * a * c) , 0.5))/(2 * a);

6.z = a + 1 (1 + (k + a) / 2);

Syllabus Topic : Data Input and Output - printf(), scanf(),getchar(), putchar(), gets(), puts() and Library Functions

Q.What are various I/O functions in 'C'?

Ans. :

I/O Functions

- Inbuilt IO functions are available in C for displaying data on the standard output device i.e. monitor and accepting the input from the standard input device i.e. keyboard.
- These IO functions are classified as formatted and unformatted functions based on the formatting permitted or not.

I. Formatted IO Functions

There are two formatted IO functions in C namely printf() to display a data on monitor and scanf() to accept data from keyboard.

1.printf()

Syntax : printf("Format string", list of variables or expressions)

The format string can contain the following :

(a)Character set (A-Z, a-z, 0-9, all special symbols on the keyboard)

(b)Blank spaces

(c)Escape sequences

(d)Field width

The width of a value can be fixed by specifying the field width. Examples of such case are given below.

(e)Format specifiers

Format specifiers specify the format of the variable data whose value is to be displayed. The format specifiers for different types of data is given in the Table 2.12.1.

Table 2.12.1 : Format specifiers

Sr. No.	Format specifier	Data type
1.	%d or % i	For integer type of data
2.	%f	For float type of data
3.	%c	For char type of data
4.	%lf	For double type of data
5.	%Lf	For long double type of data

Examples :

1. printf("The number of buildings is %d",b);

This statement will print the output with the format specifier replaced with the value of b.

2. printf("The simple interest is %f",si);

This statement will print the output where the format specifier %f will be replaced with the float type value of the variable si.

3. printf("The simple interest is %5.2f",si);

The format specifier in this statement is accompanied with a field width i.e. 5.2 (%5.2f), which indicates five digits before the fraction point and 2 digits after the fraction point. This statement will display the output wherein the value of si will be displayed with five digits before the fraction point and only two digits after the fraction point.

2.scanf()

Syntax : scanf("format String", address of variables);

Here, format string can contain format specifier, field width and assignment suppression character. The assignment suppression character (*) is used to discard one of the user entered value.

The address of the variable is obtained with the help of the address of operator (&).

Examples :

1.scanf("%d",&x);

This statement is used to accept a int type value from user in the variable x

2.scanf("%d %d %f",&x,&y,&z);

This statement is used to accept two int type data into variables x and y. It also accepts a float type data into the variable z.

II. Unformatted IO Functions

The unformatted IO functions do not have any format specifier. They are mostly used to accept and display only one character or a string (string is a set of characters to make a word or sentence).

The different unformatted IO functions are listed below;

1.getch():This function is used accept one character from the user.

2.getche():This function is used accept one character from the user and echo it (display it on the screen).

3.getchar():This function is used to accept one character from the user and echo it (display it on the screen) and also wait after that for the user to press the enter key.

4.gets():This function is used to accept a string from the user. We will see in details of this when studying the chapter on strings.

5.putch() or:These functions are used to display a character on the monitor. putchar()

6.puts():This function is used to display a string on the monitor.

➢Program 2.13.1 :Write a C program to accept a number and display its square.

- Step-form Algorithm

Step I :Declare the required variables.

Step II :Indicate the user to enter the input number by displaying suitable sentence using printf() function.

Step III:Wait using the scanf() function for the user to enter the number.

Step IV:Calculate the square of the user entered number.

Step V :Display the calculated result using printf() statement.

Step VI:Wait for user to press a key using getch() function.

Step VII :Stop

- Pseudo code Algorithm

Step I :START

Step II :PRINT "Enter a number".

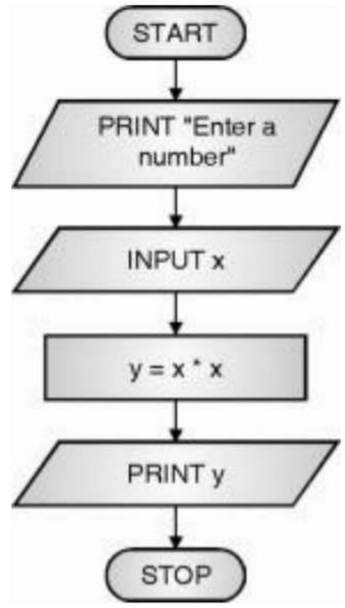
Step III:INPUT x.

Step IV:y = x * x.

Step V :PRINT y.

Step VI:STOP

- Flowchart : (Refer Flowchart 2)



Flowchart 2

➢Program

```
//Program 2.2 : This is a simple program to calculate the square of a number
#include<stdio.h>
#include<conio.h>
void main ()
{
clrscr();
int x, y;
printf("Enter a number:");
scanf("%d",&x);
y=x*x;
printf("The square of %d is %d",x,y);
getch();
}
```

Output

```
Enter a number:6
```

```
The square of 6 is 36
```

➢Program 2.13.2 :Write a C program to accept two numbers and display its product.

- Step-form Algorithm

Step I :Declare the required variables.

Step II :Indicate the user to enter the input numbers by displaying suitable sentence using printf() function.

Step III:Wait using the scanf() function for the user to enter the two numbers.

Step IV:Calculate the product of the user entered numbers.

Step V:Display the calculated result using printf() statement.

Step VI:Wait for user to press a key using getch() function.

Step VII:Stop

- Pseudo code Algorithm

Step I :START.

Step II :PRINT "Enter two numbers".

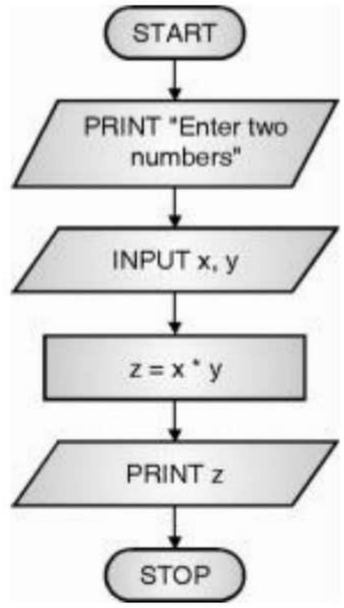
Step III:INPUT x, y.

Step IV:z = x * y.

Step V:PRINT z

Step VI:Stop

- Flowchart : (Refer Flowchart 3)



Flowchart 3

➢Program

```
#include<stdio.h>
#include<conio.h>
void main ()
{
int x, y, z;
clrscr();
printf("Enter two number:");
```

```
scanf("%d%d",&x,&y);
z=x*y;
printf("The product is %d",z);
getch();
}
```

Output

```
Enter two number:3
4
The product is 12
```

➢Program 2.13.3 :Write a program to calculate the simple interest taking principal, rate of interest and number of years as inputs from user.

- Step-form Algorithm

Step I :Declare the required variables.

Step II :Indicate the user to enter the principal amount, rate of interest and period by displaying suitable sentence using printf() function.

Step III :Wait using the scanf() function for the user to enter the data.

Step IV :Calculate the simple interest using the formula.

Step V :Display the calculated result using printf() statement.

Step VI :Wait for user to press a key using getch() function.

Step VII:Stop

- Pseudo code Algorithm

Step I :START.

Step II :PRINT "Enter principal amount, rate of interest and no. of years".

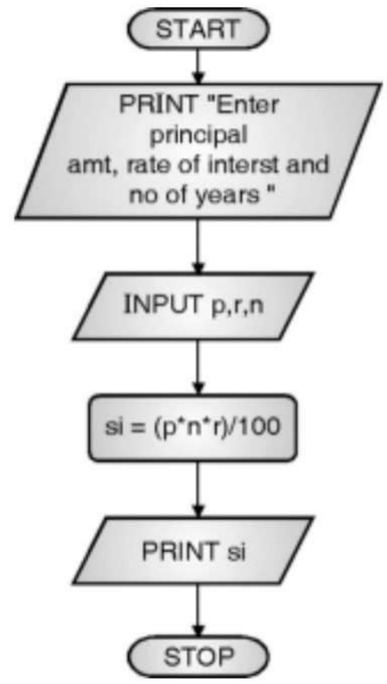
Step III :INPUT p, r, n.

Step IV :si = (p * r * n) /100.

Step V :PRINT si.

Step VI:Stop

- Flowchart : (Refer Flowchart 4)



Flowchart 4

➢Program

```
#include<stdio.h>
#include<conio.h>
void main ()
{
float rateOfInterest, years, principal, simpleInterest;
clrscr();
printf("Enter the principal amount, rate of interest and no. of years:");
scanf("%f%f%f", &principal, &rateOfInterest, &years);
simpleInterest = principal * rateOfInterest*years / 100;
printf("The simple interest is %f", simpleInterest) ;
getch();
}
```

Output

```
Enter the principal amount, rate of interest and no. of years:1000
9.5
10
The simple interest is 950.000000
```

➢Program 2.13.4 :Write a program to accept basic salary from the keyboard. Calculate the gross salary that includes basic salary, 50% DA and 40% HRA.

- Step-form Algorithm

Step I:Declare the required variables.

Step II:Indicate the user to enter the basic salary as input by displaying suitable sentence using printf() function.

Step III:Wait using the scanf() function for the user to enter the input.

Step IV:Calculate the HRA and DA of the user entered basic salary.

Step V:Calculate the total salary i.e. the sum of basic, HRA RA and DA

Step VI:Display the calculated gross salary using printf() statement.

Step VII:Wait for user to press a key using getch() function.

Step VIII:Stop

- Pseudo code Algorithm

Step I:START.

Step II:PRINT "Enter basic salary".

Step III:INPUT basic.

Step IV:HRA = 0.4 * basic

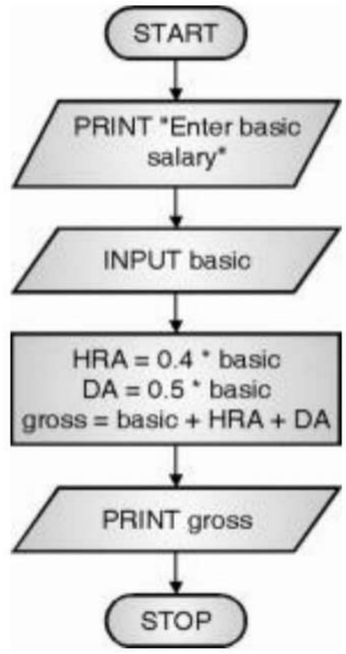
Step V:DA = 0.5 * basic

Step VI:gross = basic + HRA + DA.

Step VII:PRINT gross.

Step VIII:STOP

- Flowchart : (Refer Flowchart 5)



Flowchart 5

➢Program

```
#include<stdio.h>
#include<conio.h>
void main ()
{
float basic,hra,da,gross;
clrscr();
printf("Enter the basic salary:");
scanf("%f", &basic);
hra=40*basic/100;
da = 50*basic/100;
gross=basic+da+hra;
```

```
printf("The total salary is %f",gross);
getch();
}
```

Output

```
Enter the basic salary:10000
The total salary is 19000.000000
```

➢Program 2.13.5 :Write a program to accept the distance between two cities in kilometers from the keyboard. Calculate and display this distance in meters, feet, centimeters and inches.

- Step-form Algorithm

Step I :Declare the required variables.

Step II :Indicate the user to enter the distance between two cities in kilometers by displaying suitable sentence using printf() function.

Step III:Wait using the scanf() function for the user to enter the input.

Step IV:Calculate the distance in meters, centimeters and inches using proper formulae.

Step V:Display the calculated result using printf() statement.

Step VI:Wait for user to press a key using getch() function.

Step VII : Stop

- Pseudo code Algorithm

Step I :START.

Step II :PRINT "Enter distance in kilometers between the two cities".

Step III:INPUT km.

Step IV:m = km * 1000

Step V:cm = m * 100

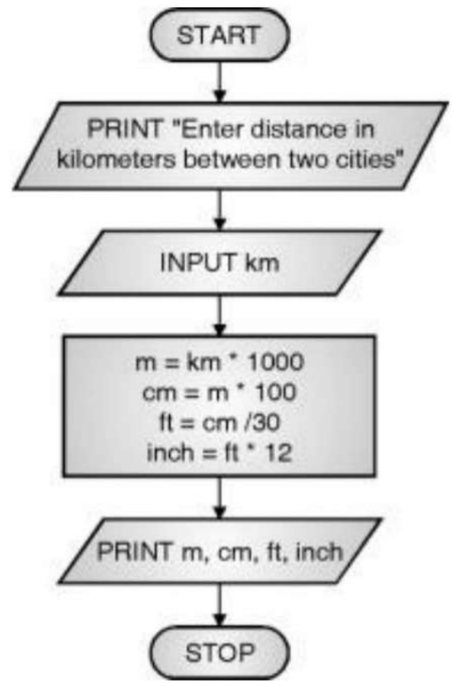
Step VI:ft = cm / 30

Step VII:inch = ft * 12

Step VIII:PRINT m, cm, ft, inch.

Step IX:STOP

- Flowchart : (Refer Flowchart 6)



Flowchart 6

➢Program

```
#include<stdio.h>
#include<conio.h>
void main ()
{
float km, mt, inch, ft, cm;
clrscr();
printf("Enter the distance between two cities in kilometers:");
scanf("%f",&km);
mt =km*1000;
ft = mt * 3.33;
cm = mt * 100;
inch = ft *12;
printf("The distance is= %f mts = %f ft = %f  cm = %f inches.", mt, ft, cm,  inch);
getch();
}
```

Output

```
Enter the distance between two cities in kilometers:25
The distance is= 25000.000000 mts = 83250.000000 ft = 2500000.000000 cm = 999000.000000 inches.
```

➢Program 2.13.6 :Write a program to accept the length and breadth of a rectangle from the user. Calculate and display the area and perimeter.

- Step-form Algorithm

Step I:Declare the required variables.

Step II:Indicate the user to enter the length and breadth of the rectangle as input from user by displaying suitable sentence using printf() function.

Step III:Wait using the scanf() function for the user to enter the input.

Step IV:Calculate the area and perimeter using the corresponding formulae.

Step V:Display the calculated area and perimeter using printf() statement.

Step VI:Wait for user to press a key using getch() function.

Step VII:Stop

- Pseudo code Algorithm

Step I:START

Step II:PRINT "Enter the length and breadth of the rectangle".

Step III:INPUT *l*, b.

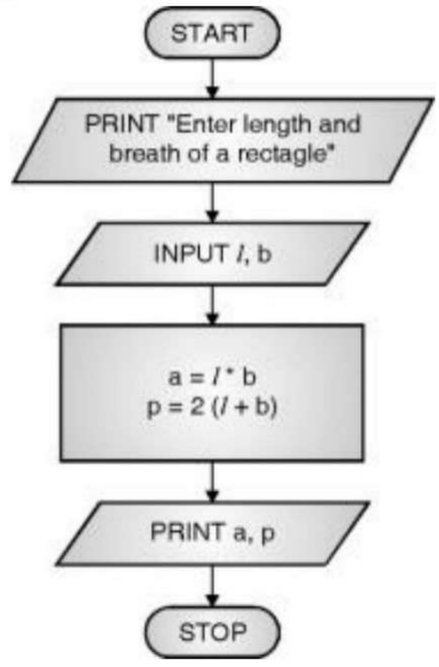
Step IV:a = *l* * b

Step V:p = 2 (*l* + b)

Step VI:PRINT a, p.

Step VII:STOP

- Flowchart : (Refer Flowchart 7)



Flowchart 7

➢Program

```
#include<stdio.h>
#include<conio.h>
void main ()
{
float length, breadth, area, perimeter;
clrscr();
printf("Enter the length and breadth of the rectangle:");
scanf("%f%f", &length, &breadth);
area= length * breadth;
perimeter= 2* (length + breadth);
printf("The area of rectangle is= %f and its perimeter is = %f", area, perimeter);
```

```
getch();
}
```

Output

```
Enter the length and breadth of the rectangle:10
15
The area of rectangle is= 150.000000 and its perimeter is = 50.000000
```

➢Program 2.13.7 :Write a program to separate the digits of a three digit number and display the three digits separately.

- Step form Algorithm

Step I:Declare the required variables.

Step II:Indicate the user to enter a three digit number by displaying suitable sentence using printf() function.

Step III:Wait using the scanf() function for the user to enter the input.

Step IV:Get the last digit of the user entered number into a variable using the modulus operator ('%'). And by the quotient operator ('/') get the remaining part of the number into the same variable as that of the original number.

Step V:Repeat the step IV to get the remaining two digits.

Step VI:Display the separated digits using printf() statement.

Step VII:Wait for user to press a key using getch() function.

Step VIII:Stop

- Pseudo code Algorithm

Step I :START

Step II:PRINT "Enter a three digit number".

Step III:INPUT n.

Step IV:d1 = n mod 10

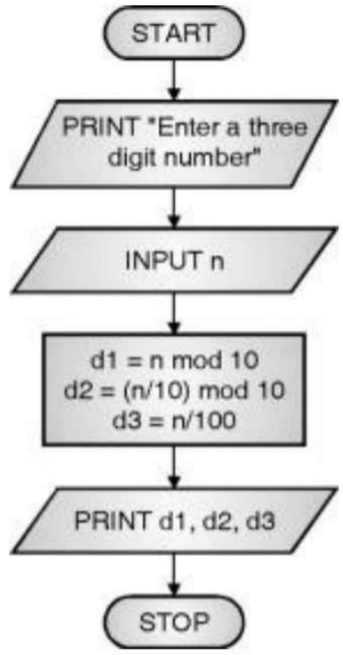
STEP V:d2 = (n / 10) mod 10

STEP VI:d3 = n / 100

Step VII:PRINT d1, d2, d3

Step VIII:STOP

- Flowchart (Refer Flowchart 8)



Flowchart 8

➢Program

```
#include<stdio.h>
#include<conio.h>
void main ()
{
int d1, d2, d3, number;
clrscr();
printf("Enter a three digit number:");
scanf("%d",&number);
d3=number%10;
number=number/10;
d2= number%10;
number =number / 10;
d1=number%10;
printf("The digits are :%d\n%d\n%d",d1,d2,d3);
getch();
}
```

Output

```
Enter a three digit number:345
The digits are :3
4
5
```

➢Program 2.13.8 :Write a program to accept one int type data and one float type data. Multiply the two numbers and display the result.

- Step form Algorithm

Step I :Declare the required variables.

Step II:Indicate the user to enter one data of int type and another of float type by displaying suitable sentence using printf() function.

Step III:Wait using the scanf() function for the user to enter the input.

Step IV:Multiply the user entered numbers and store it in a float type variable.

Step V:Display the product using printf() statement.

Step VI:Wait for user to press a key using getch() function.

Step VII:Stop

- Pseudo code Algorithm

Step I :START.

Step II:PRINT "Enter an integer and a float number".

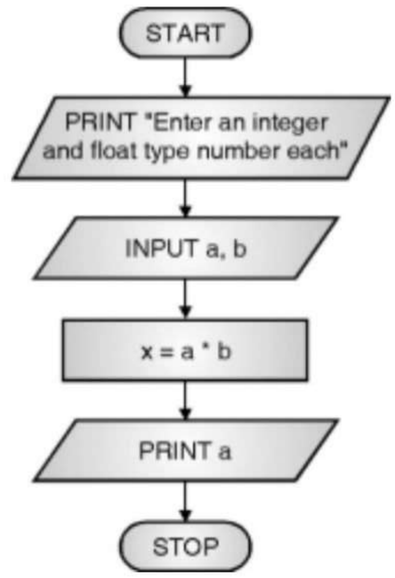
Step III:INPUT a, b

Step IV:x = a * b.

Step V:PRINT x

Step VI:Stop

- Flowchart : (Refer Flowchart 9)



Flowchart 9

➢Program

```
#include<stdio.h>
#include<conio.h>
void main ()
{
int n1;
float n2, result;
clrscr();
printf("Enter one integer and one float type number each:");
scanf("%d %f",&n1,&n2);
result=n1*n2;
printf("The product is :%f",result);
```

```
getch();
}
```

Output

```
Enter one integer and one float type number each:2
3.9
The product is :7.8
```

➢Program 2.13.9 :Write a program to accept two numbers from user and display the greatest of two using the conditional operator.

- Step form Algorithm

Step I:Declare the required variables.

Step II:Indicate the user to enter two numbers by displaying suitable sentence using printf() function.

Step III:Wait using the scanf() function for the user to enter the input.

Step IV:Using the proper syntax of the conditional or ternary operator, find the largest of the two numbers and store in another variable.

Step V:Display the greatest number using printf() statement.

Step VI:Wait for user to press a key using getch() function.

Step VII:Stop

- Pseudo code Algorithm

Step I:START.

Step II:PRINT "Enter two numbers".

Step III:INPUT a, b.

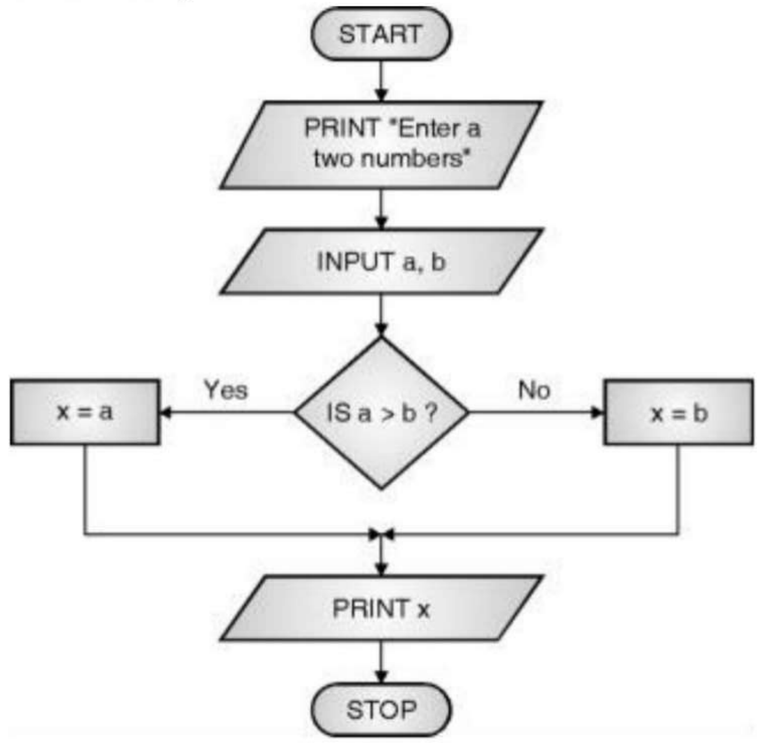
Step IV:IF a > b, THEN x = a

ELSE x = b.

Step V:PRINT x.

Step VI:STOP

- Flowchart : (Refer Flowchart 10)



Flowchart 10

➢Program

```
#include<stdio.h>
#include<conio.h>
void main ()
{
int n1, n2, greater;
clrscr();
printf("Enter two numbers:");
scanf("%d %d",&n1,&n2);
greater=(n1>n2)?n1:n2;
printf("The larger number is :%d",greater);
getch();
}
```

Output

```
Enter two numbers:35
12
The larger number is :35
```

➢Program 2.13.10 :Write a program to accept three numbers from user and display the greatest of three using the conditional operator. ***(Dec. 2014, May 2016)***

- Step form Algorithm

Step I:Declare the required variables.

Step II:Indicate the user to enter three numbers by displaying suitable sentence using printf() function.

Step III:Wait using the scanf() function for the user to enter the input.

Step IV:Using the proper syntax of the conditional or ternary operator, find the largest of the three numbers and store in another variable.

Step V:Display the greatest number using printf() statement.

Step VI:Wait for user to press a key using getch() function.

Step VII:Stop

- Pseudo code Algorithm

Step I:START.

Step II:PRINT "Enter three numbers".

Step III:INPUT a, b, c.

Step IV:IF a > b AND a > c THEN x = a

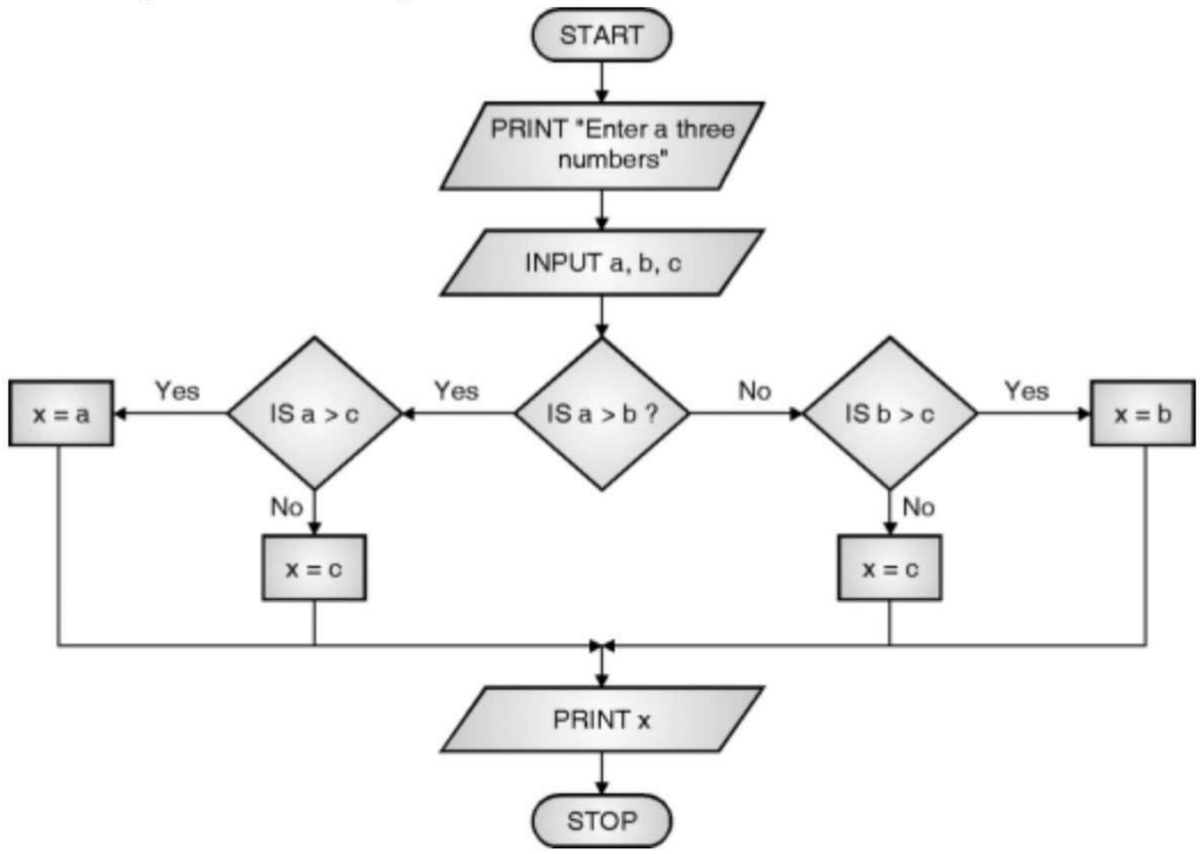
ELSE b > c THEN x = b

ELSE x = c.

Step V:PRINT x

Step VI:STOP

- Flowchart : (Refer Flowchart 11)



Flowchart 11

➢Program

```
#include<stdio.h>
#include<conio.h>
void main ()
{
int n1, n2, n3, greater;
clrscr();
printf("Enter three numbers:");
scanf("%d %d %d", &n1,&n2,&n3);
greater=(n1>n2)?((n1>n3)?n1:n3):((n2>n3)?n2:n3);
printf("The largest number is :%d",greater);
getch();
}
```

Output

```
Enter three numbers:234
23
33
The largest number is :234
```

➢Program 2.13.11 :Write a program to find the area of a circle.

- Step form Algorithm

Step I :Declare the required variables.

Step II:Indicate the user to enter the radius of the circle by displaying suitable sentence using printf() function.

Step III:Wait using the scanf() function for the user to enter the input.

Step IV:Using the proper formula calculate the area of the circle and store in another variable.

Step V:Display the area of the circle using printf() statement.

Step VI:Wait for user to press a key using getch() function.

Step VII:Stop

- Pseudo code Algorithm

Step I :START.

Step II:PRINT "Enter the radius of a circle".

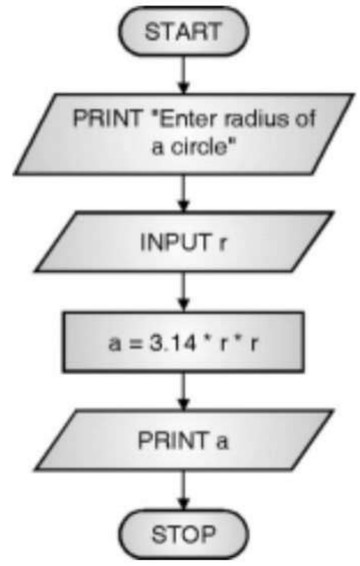
Step III:INPUT R.

Step IV:a = 3.14 * r * r.

Step V:PRINT a.

Step VI:STOP

- Flowchart : (Refer Flowchart 12)



Flowchart 12

➢Program

```
#include<stdio.h>
#include<conio.h>
void main ()
{
float r, area;
clrscr();
printf("Enter the radius of a circle:");
scanf("%f",&r);
area=3.14 *r*r;
printf("The area is equal:%f",area);
getch();
```

```
}
```

Output

```
Enter the radius of a circle:2.5
The area is equal:19.625000
```

➢Program 2.13.12 :Write a program to convert temperature in Fahrenheit to Celsius.

- Step form Algorithm

Step I:Declare the required variables.

Step II:Indicate the user to enter the temperature from user in Fahrenheit by displaying suitable sentence using printf() function.

Step III:Wait using the scanf() function for the user to enter the input.

Step IV:Using the proper formula calculate the temperature to degree Celsius..

Step V:Display the temperature in Celcius using printf() statement.

Step VI:Wait for user to press a key using getch() function.

Step VII:Stop

- Pseudo code Algorithm

Step I:START.

Step II:PRINT "Enter the temperature in Fahrenheit".

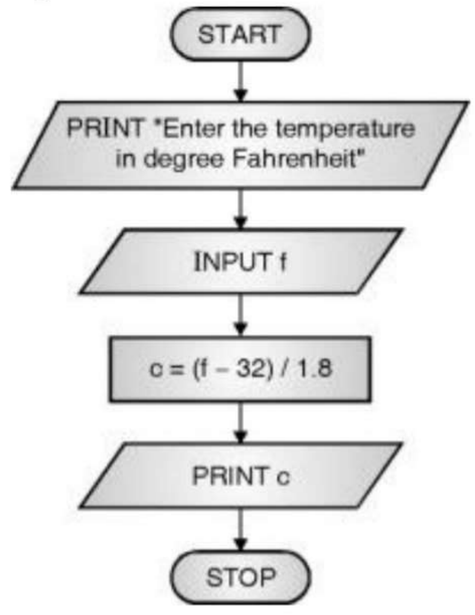
Step III:INPUT f.

Step IV:c = (f – 32) /1.8.

Step V:PRINT c.

Step VI:STOP

- Flowchart : (Refer Flowchart 13)



Flowchart 13

➢Program

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

```
{
float f, c;
clrscr();
printf("Enter the temperature in fahrenheit:");
scanf("%f",&f);
c=(f – 32)/1.8;
printf("The temperature in degree celsius is equal:%f",c);
getch();
}
```

Output

```
Enter the temperature in fahrenheit:96.6
The temperature in degree celsius is equal:35.888889
```

➢Program 2.13.13 :Write a program to accept three numbers and find their average.

- Step form Algorithm

Step I:Declare the required variables.

Step II:Indicate the user to enter the three numbers by displaying suitable sentence using printf() function.

Step III:Wait using the scanf() function for the user to enter the input.

Step IV:Add the three numbers and divide the sum by three..

Step V:Display the average calculated using printf() statement.

Step VI:Wait for user to press a key using getch() function.

Step VII:Stop

- Pseudo code Algorithm

Step I:START.

Step II:PRINT "Enter three numbers".

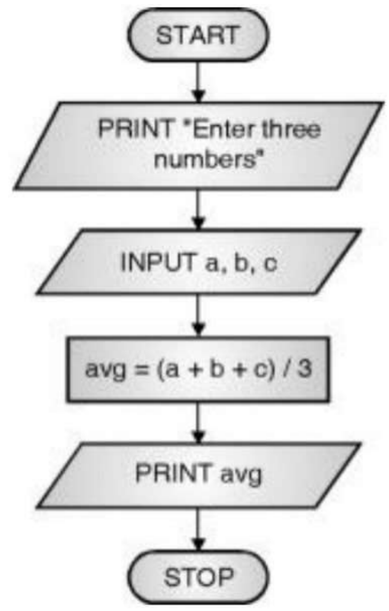
Step III:INPUT a, b, c

Step IV:avg = (a + b + c) / 3

Step V:PRINT avg.

Step VI:STOP

- Flowchart : (Refer Flowchart 14)



Flowchart 14

➢Program

```
#include<stdio.h>
#include<conio.h>
void main ()
{
int a, b, c, sum;
float avg;
clrscr();
printf("Enter three numbers:");
scanf("%d %d %d",&a,&b,&c);
sum= a+b+c;
avg=sum/3.0;
printf("The average is equal to:%f",avg);
getch();
}
```

Output

```
Enter three numbers:3
4
6
The average is equal to:4.333333
```

➢Program 2.13.14 :Write a program to swap two integers.

- Step form Algorithm

Step I:Declare the required variables.

Step II:Indicate the user to enter two numbers by displaying suitable sentence using printf() function.

Step III:Wait using the scanf() function for the user to enter the input.

Step IV:Using a temporary variable swap the two numbers. The logic to swap the numbers using the following logic

(i)Put the first number in the temporary variable.

(ii)Put the second number in the first number

(iii)Put the temporary variable in the second number

Step V:Display the two numbers indicating the two numbers are swapped using printf() statement.

Step VI:Wait for user to press a key using getch() function.

Step VII:Stop

- Pseudo code Algorithm

Step I:START

Step II:PRINT "Enter two numbers".

Step III:INPUT a, b.

Step IV:temp = a.

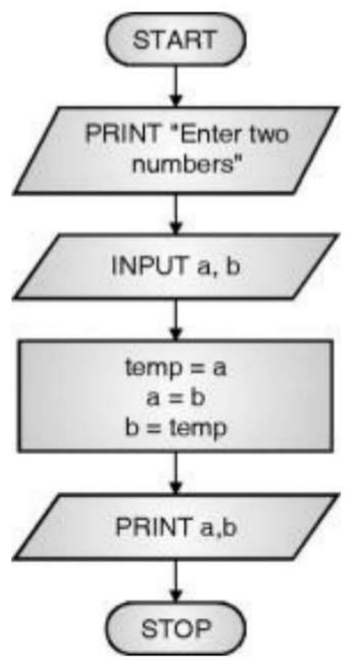
Step V:a = b

Step VI:b = temp

Step VII:PRINT a , b.

Step VIII:STOP

- Flowchart : (Refer Flowchart 15)



Flowchart 15

➢Program

```
#include<stdio.h>
#include<conio.h>
void main ()
{
int a, b, temp;
clrscr();
printf("Enter two numbers:");
```

```
scanf("%d %d",&a,&b);
temp=a;
a=b;
b=temp;
printf("After swapping the values are a=%d and b=%d",a,b);
getch();
}
```

Output

```
Enter two numbers:3
4
After swapping the values are a=4 and b=3
```

➢Program 2.13.15 :Write a program to shift a number three bits left and display the result.

- Step form Algorithm

Step I:Declare the required variables.

Step II:Indicate the user to enter a number from user by displaying suitable sentence using printf() function.

Step III:Wait using the scanf() function for the user to enter the input.

Step IV:Using the shift operator shift the number three times.

Step V:Display the result using printf() statement.

Step VI:Wait for user to press a key using getch() function.

Step VII:Stop

- Pseudo code Algorithm

Step I:START.

Step II:PRINT "Enter a number"

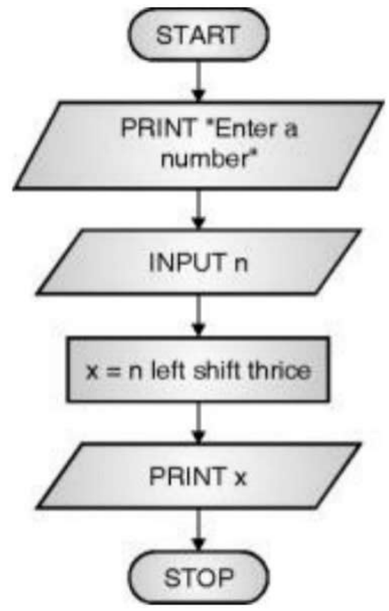
Step III:INPUT n.

Step IV:x = n LEFT SHIFT thrice.

Step V:PRINT x.

Step VI:STOP

- Flowchart : (Refer Flowchart 16)



Flowchart 16

➢Program

```
#include<stdio.h>
#include<conio.h>
void main ()
{
int a, b;
clrscr();
printf("Enter a number to be shifted left:");
scanf("%d",&a);
b=a<<3;
printf("After shifting three times left the value is %d",b);
getch();
}
```

Output

```
Enter a number to be shifted left:5
After shifting three times left the value is 40
```

➢Program 2.13.16 :Write a program to accept a number from user and find the remainder after dividing it by 2 and 3.

- Step form Algorithm

Step I:Declare the required variables.

Step II:Indicate the user to enter a number from user by displaying suitable sentence using printf() function.

Step III:Wait using the scanf() function for the user to enter the input.

Step IV:Using the modulus operator ('%') find the remainder by dividing the user entered number by 2 and 3.

Step V:Display the remainders using printf() statement.

Step VI:Wait for user to press a key using getch() function.

Step VII:Stop

- Pseudo code Algorithm

Step I:START.

Step II:PRINT "Enter a number".

Step III:INPUT n.

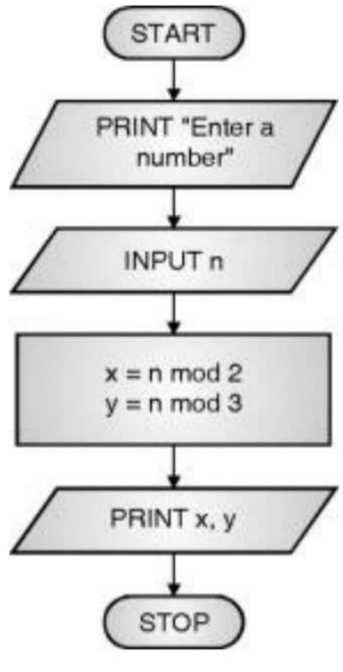
Step IV:x = n mod 2.

Step V:y = n mod 3

Step VI:PRINT x, y.

Step VII:STOP

- Flowchart : (Refer Flowchart 17)



Flowchart 17

➢Program

```
#include<stdio.h>
#include<conio.h>
void main ()
{
int a, rem1, rem2;
clrscr();
printf("Enter a number:");
scanf("%d",&a);
rem1=a%2;
rem2=a%3;
printf("Remainder after dividing by 2 and 3 are %d and %d respectively", rem1, rem2);
getch();
}
```

Output

```
Enter a number:23
```

```
Remainder after dividing by 2 and 3 are 1 and 2 respectively
```

➢Program 2.13.17 : Write a program to accept a two digit number and display it in reversed form.

- Step form Algorithm

Step I:Declare the required variables.

Step II:Indicate the user to enter a two digit number from user by displaying suitable sentence using printf() function.

Step III:Wait using the scanf() function for the user to enter the input.

Step IV:Using the modulus operator ('%') find the lower digit and quotient operator ('/') to find the upper digit. Multiply the units digit by 10 and add it to the tens digit, hence reversing the number.

Step V:Display the calculated reverse number using printf() statement.

Step VI:Wait for user to press a key using getch() function.

Step VII:Stop

- Pseudo code Algorithm

Step I:START.

Step II:PRINT "Enter a two digit number".

Step III:INPUT n.

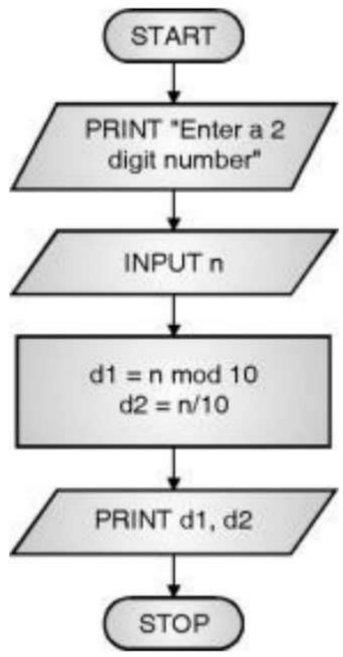
Step IV:d1 = n mod 10

Step V:d2 = n / 10

Step VI:PRINT d1, d2.

Step VII:STOP

- Flowchart : (Refer Flowchart 18)



Flowchart 18

➢Program

```
#include<stdio.h>
#include<conio.h>
void main ()
{
int a, digit1, digit2;
```

```
clrscr();
printf("Enter a number:");
scanf("%d",&a);
digit1=a%10;
a=a/10;
digit2=a%10;
printf("Reversed no is %d%d",digit1,digit2);
getch();
}
```

Output

```
Enter a number:67
Reversed no is 76
```

➢Program 2.13.18 :Write a program to check if the entered number is even or odd.

- Step form Algorithm

Step I :Declare the required variables.

Step II :Indicate the user to enter a number from user by displaying suitable sentence using printf() function.

Step III:Wait using the scanf() function for the user to enter the input.

Step IV:Using the modulus operator ('%') find the remainder. If the remainder is one, then the number entered is odd and if the remainder is zero, the number entered is even.

Step V:Display the statement if the number is even or odd using printf() statement.

Step VI:Wait for user to press a key using getch() function.

Step VII:Stop.

- Pseudo code Algorithm

Step I :START.

Step II :PRINT "Enter a number".

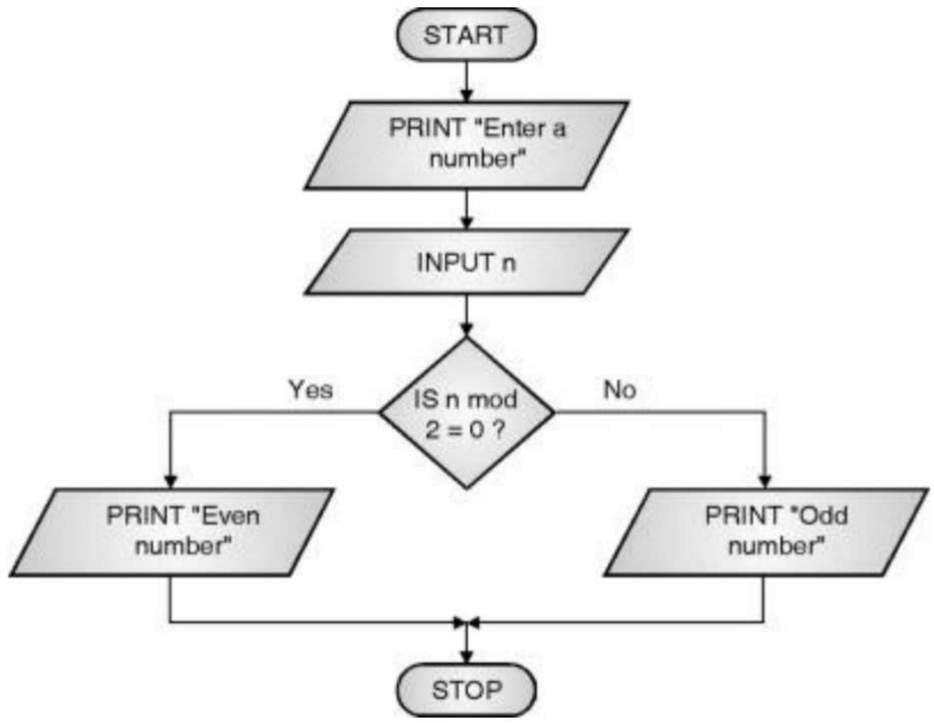
Step III:INPUT n.

Step IV:IF n mod 2 = 0 THEN PRINT "Even Number"

ELSE PRINT "Odd Number".

Step V:STOP.

- Flowchart : (Refer Flowchart 19)



Flowchart 19

➢Program

```
#include<stdio.h>
#include<conio.h>
void main ()
{
int a, rem;
clrscr();
printf("Enter a number:");
scanf("%d",&a);
rem=a%2;
(rem==0)?printf("Even number"):printf("Odd number");
getch();
}
```

Output

```
Enter a number:23
Odd number
```

➢Program 2.13.19 :Write a program to accept a float number and display the integer part using type casting operator.

- Step form Algorithm

Step I:Declare the required variables.

Step II:Indicate the user to enter a float type number from user by displaying suitable sentence using printf() function.

Step III:Wait using the scanf() function for the user to enter the input.

Step IV:Using the type casting operator find the integer type value of the user entered number.

Step V:Display the integer obtained in step IV, using printf() statement.

Step VI:Wait for user to press a key using getch() function.

Step VII:Stop.

- Pseudo code Algorithm

Step I:START.

Step II:PRINT "Enter a float type number".

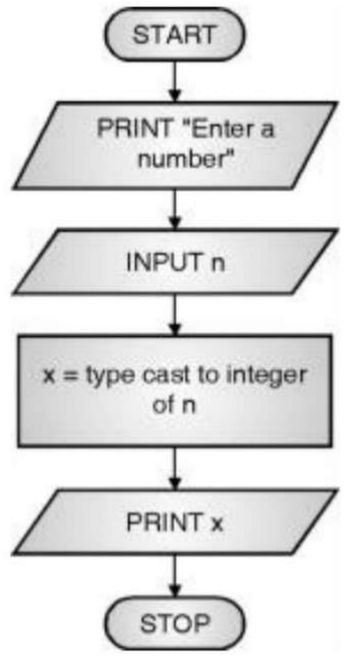
Step III:INPUT n.

Step IV:x = type cast to integer of n.

Step V:PRINT x.

Step VI:STOP.

- Flowchart : (Refer Flowchart 20)



Flowchart 20

➢Program

```
#include<stdio.h>
#include<conio.h>
void main ()
{
int a;
float b;
clrscr();
printf("Enter a float number:");
scanf("%f",&b);
a=(int)b;
printf("The integer part of the number is:%d",a);
getch();
}
```

Output

```
Enter a float number:3.14
```

```
The integer part of the number is:3
```

➢Program 2.13.20 : Write a program to accept two numbers and display the result of their AND, OR, EXOR and NOT operations.

- Step form Algorithm

Step I:Declare the required variables.

Step II:Indicate the user to enter two numbers from user by displaying suitable sentence using printf() function.

Step III:Wait using the scanf() function for the user to enter the input.

Step IV:Using the bitwise operators calculate the results according to the said operations.

Step V:Display the results calculated using printf() statement.

Step VI:Wait for user to press a key using getch() function.

Step VII:Stop.

- Pseudo code Algorithm

Step I:START.

Step II:PRINT "Enter two numbers".

Step III:INPUT a, b.

Step IV:x = a AND b.

Step V:y = a OR b.

Step VI:z = a EXOR b

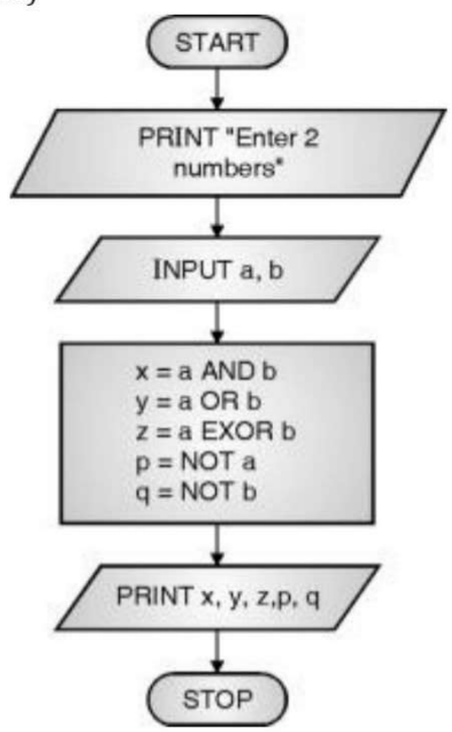
Step VII:p = NOT a.

Step VIII:q = NOT b.

Step IX:PRINT x, y, z, p, q

Step X:STOP

- Flowchart : (Refer Flowchart 21)



Flowchart 21

➢Program

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

```
#include<conio.h>
void main ()
{
int a,b;
clrscr();
printf("Enter two numbers:");
scanf("%d %d",&a,&b);
printf("The AND of %d and %d is %d\n",a,b,(a & b));
printf("The OR of of %d and %d is %d\n",a,b,(a | b));
printf("The EXOR of of %d and %d is %d\n",a,b,(a ^ b));
printf("The NOT of %d is %d\n",a,~a);
printf("The NOT of %d is %d\n",b,~b);
getch();
}
```

Output

```
Enter two numbers:5
9
The AND of 5 and 9 is 1
The OR of 5 and 9 is 13
The EXOR of 5 and 9 is 12
The NOT of 5 is – 6
The NOT of 9 is – 10
```

➢Program 2.13.21 :Write a program to accept a number and display its equivalent ASCII using type casting.

- Step form Algorithm

Step I:Declare the required variables.

Step II:Indicate the user to enter a number from user by displaying suitable sentence using printf() function.

Step III:Wait using the scanf() function for the user to enter the input.

Step IV:Using the type casting operator find the char equivalent of the user entered number.

Step V:Display the char type data calculated using printf() statement.

Step VI:Wait for user to press a key using getch() function.

Step VII:Stop.

- Pseudo code Algorithm

Step I:START.

Step II:PRINT "Enter an integer number".

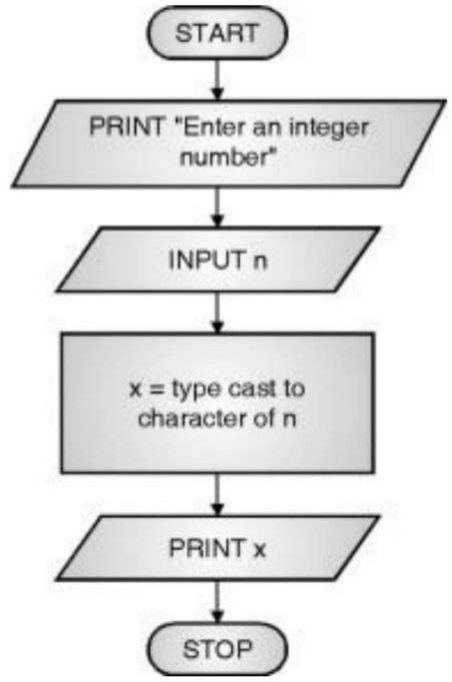
Step III:INPUT n.

Step IV:x = type casting to character of n.

Step V:PRINT x

Step VI:STOP.

- Flowchart : (Refer Flowchart 22)



Flowchart 22

➢Program

```
#include<stdio.h>
#include<conio.h>
void main ()
{
int a;
clrscr();
printf("Enter a number:");
scanf("%d",&a);
printf("This is ASCII value of: %c",(char)a);
getch();
}
```

Output

```
Enter a number:65
This is ASCII value of: A
```

➢Program 2.13.22 : Find the output of the following program.

```
#include<stdio.h>
#include<conio.h>
void main ()
{
int a=2,b=3,ab=4;
int i;
```

```
int in='2'*2;
char ch='c';
clrscr();
printf("%c %c\n",ch,(++ch));
printf("%d %d %d\n",a,a,++a);
printf("%d %d %d\n",b,b,++b);
printf("%d %d %d\n",ab,ab,++ab);
printf("%d %d\n",a,!!a);
getch();
}
```

Output

```
d d
3 3 3
4 4 4
5 5 5
3 1
```

➢Program 2.13.23 :Find the output of the following program.

```
#include<stdio.h>
#include<conio.h>
void main ()
{
int a=3;
clrscr();
printf(" %d\n",a);
printf(" %d\n",a++);
printf(" %d\n",++a);
getch();
}
```

Output

```
3
3
5
```

➢Program 2.13.24 : Find the output of the following program.

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

```
void main ()
{
int x=4,y=9;
int z;
clrscr();
z=(x++)+(– –y)+y;
printf("Value=%d\n",z);
z=(–– x)+x+(y ––);
printf("Value=%d\n",z);
getch();
}
```

Output

```
Value=20
Value=16
```

➢Program 2.13.25 : Find the output of the following program.

```
#include<stdio.h>
#include<conio.h>
void main ()
{
int a,b,c;
clrscr();
a=2;b=5;c=10;
printf("Value=%d\n",(a+b*– c));
printf("Value=%d\n",(– c/b*c – a));
printf("Value=%d\n",(– a+ ++b %a));
getch();
}
```

Output

```
Value=–48
Value=–22
Value=–2
```

➢Program 2.13.26 : Find the output of the following program.

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

```
{
int a=5,b=3;
float c;
clrscr();
c=a/b;
printf("%f",c);
getch();
}
```

Output

```
1.000000
```

➢Program 2.13.27 : Find the output of the following program.

```
#include<stdio.h>
#include<conio.h>
void main ()
{
int a=10,b,c;
clrscr();
c=b=a;
b– =a – –;
c– =– – a;
a – =–– a-a– –;
printf("a=%d\nb=%d\nc=%d\n",a,b,c);
getch();
}
```

Output

```
a=6
b=0
c=2
```

➢Program 2.13.28 : Find the output of the following program.

```
#include<stdio.h>
#include<conio.h>
void main ()
{
int k=3,l=4,m;
clrscr();
```

```
m=++k +l– –;
printf("Value of m %d\n",m);
m=k++ + – –l;
printf("Value of m %d\n",m);
getch();
}
```

Output

```
Value of m 8
Value of m 6
```

➢Program 2.13.29 :Write a program that calculates the roots of quadratic equation. ***(May 2013)***

- Step form Algorithm

Step I:Declare the required variables.

Step II:Indicate the user to enter the coefficients of the quadratic equation by displaying suitable sentence using printf() function.

Step III:Wait using the scanf() function for the user to enter the input.

Step IV:Calculate the roots of quadratic equation using the proper formulae.

Step V:Display the result i.e. the roots calculated using printf() statement.

Step VI:Wait for user to press a key using getch() function.

Step VII:Stop.

- Pseudo code Algorithm

Step I:START.

Step II:PRINT "Enter the coefficients of the quadratic equation"

Step III:INPUT a, b, c.

Step IV:x_1 = (– b + pow ((b * b – 4 * a * c) , 0.5))/(2 * a)

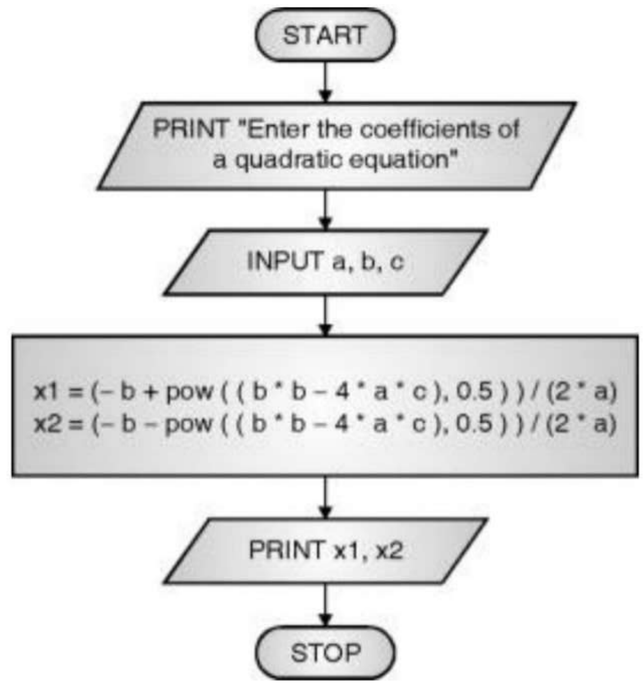
x_2 = (– b – pow ((b * b – 4 * a * c) , 0.5))/(2 * a)

Note : pow() is an algorithm which finds the value of x^y with x and y passed to it

Step V:PRINT x1 and x2.

Step VI:STOP.

- Flowchart : (Refer Flowchart 23)



Flowchart 23

➢Program

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
void main ()
{
int a,b,c;
float x1,x2;
clrscr();
printf("Enter the coefficients of the quadratic equation:");
scanf("%d %d %d",&a,&b,&c);
x1=(–b+pow((b*b–4*a*c),0.5))/(2*a);
x2=(–b–pow((b*b–4*a*c),0.5))/(2*a);
printf("The roots of quadratic equation are:%f and %f",x1,x2);
getch();
}
```

Output

```
Enter the coefficients of the quadratic equation:1
2
1
The roots of quadratic equation are: –1.000000 and –1.000000
```

➢Program 2.13.30 : Find the output of the following program.

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

```
void main ()
{
int a=1,b=2,c=3,d=4.75,x;
clrscr();
x=++a + b++ * ++c % d++;
printf("%d %d %d %d %d",a,b,c,d,x);
getch();
}
```

Output

```
2 3 4 5 2
```

➢Program 2.13.31 : Find the output of the following program.

```
#include<stdio.h>
#include<conio.h>
void main ()
{
int x=1;
clrscr();
printf("%d%d%d\n",x,(x=x+2),(x<<2));
x<<2;
printf("%d%d%d\n",++x,x++,++x);
getch();
}
```

Output

```
334
644
```

➢Program 2.13.32 : Find the output of the following program.*(May 2013)*

```
#include<stdio.h>
#include<conio.h>
void main ()
{
int x=10,y,z;
clrscr();
z=y=x;
y-=- -x;
z-=x- -;
```

```
x-=- -x-x- -;
printf("x = %d y = %d z = %d",x,y,z);
getch();
}
```

Output

```
x=6 y= 1 z= 1
```

❑❑❑

CHAPTER

3

Control Structures : Looping

Syllabus Topic : Control Structures - for Looping

➢Program 3.2.1 :Write a program to display the word "Computer" five times using for loop.

➢Algorithm :

Step I:START

Step II:i = 1.

Step III:IF i > 5 THEN GOTO step VII

Step IV:PRINT "Computer".

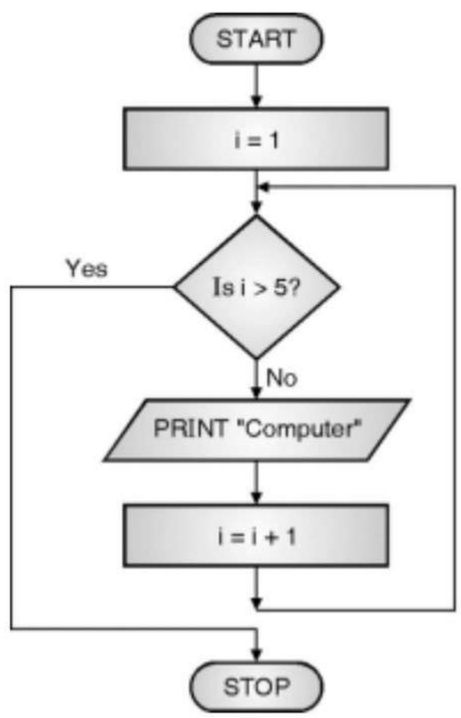
Step V:i = i + 1.

Step VI:GOTO step III

Step VII:STOP.

Note : Notice the right shifted statements in algorithm i.e. Step III to VI. The shift indicates that these statements are in a loop or are to be executed multiple times.

➢Flowchart 1 : (Refer Flowchart 1)



Flowchart 1

➢Program

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

```
{
int i;
clrscr();
for(i=1;i<=5;i++)
{
printf("Computer\n");
}
getch();
}
```

Output

```
Computer
Computer
Computer
Computer
Computer
```

➢Program 3.2.2 : Write a program to display the first ten natural numbers.

➢Algorithm

Step I:START

Step II:i = 1

Step III:IF i > 10 THEN GOTO step VII.

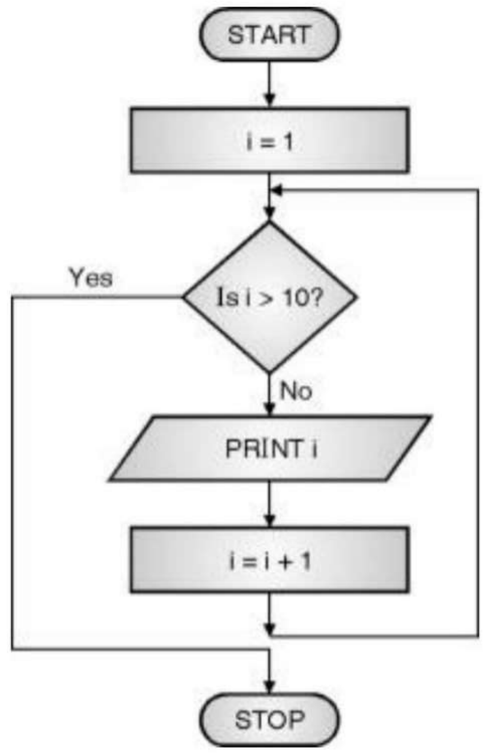
Step IV:PRINT i.

Step V:i = i + 1

Step VI:GOTO step III

Step VII:STOP.

➢Flowchart : (Refer Flowchart 2)



Flowchart 2

➢Program

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

Output

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

➢Program 3.2.3 :Write a program to display the first n natural numbers, where the value of n is taken from user.

➢Algorithm

Step I:START

Step II:PRINT "Enter a number".

Step III:INPUT n.

Step IV:i = 1.

Step V:IF i > n THEN GOTO step IX.

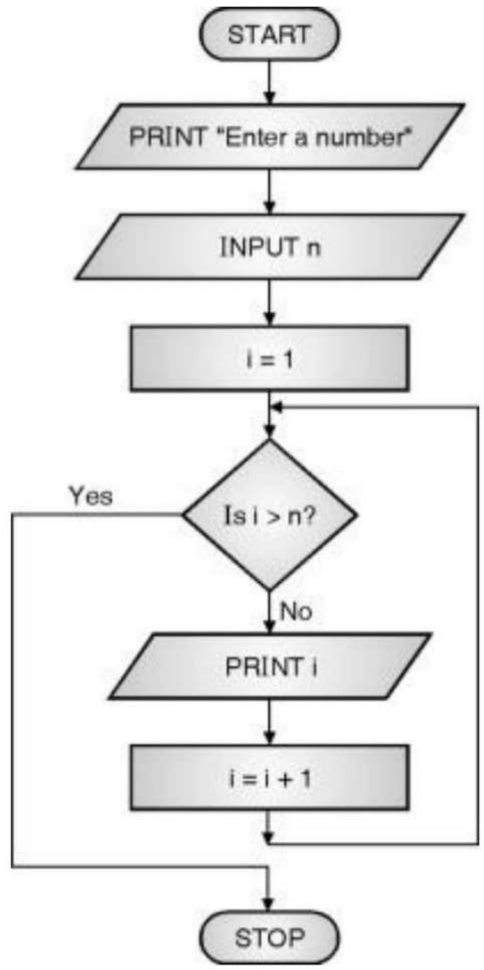
Step VI:PRINT i.

Step VII:i = i + 1.

Step VIII:GOTO step V

Step IX:STOP.

➢Flowchart : (Refer Flowchart 3)



Flowchart 3

➢Program

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

Output

```
Enter a number: 7
1
2
3
```

```
4
5
6
7
```

➢Program 3.2.4 :Write a program to find the factorial of a number.

➢Algorithm

STEP I:START

Step II:PRINT "Enter a number".

Step III:INPUT n.

Step IV:i = 1, fact = 1

Step V:IF i > n THEN GOTO step IX.

Step VI:fact = fact * i.

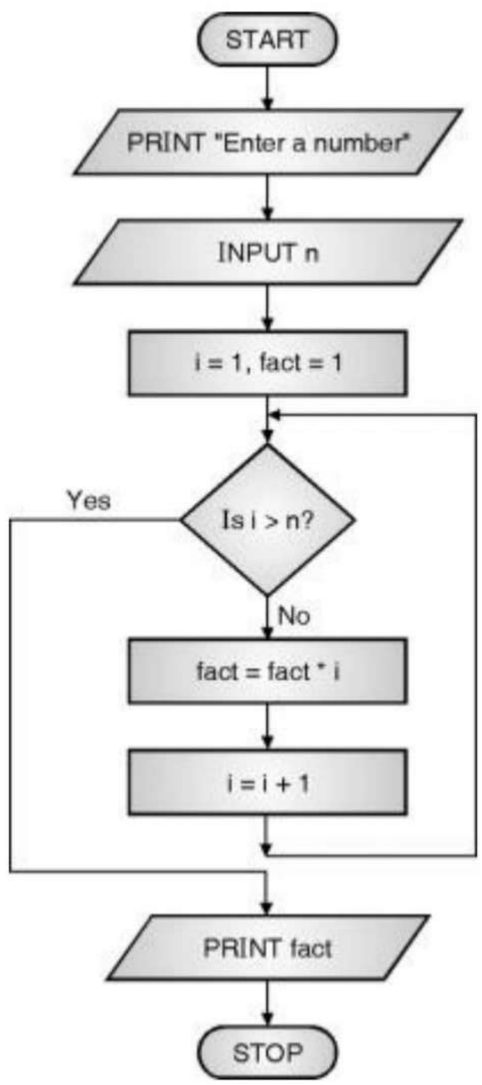
Step VII:i = i + 1

Step VIII:GOTO step V

Step IX:PRINT fact

Step X:STOP

➢Flowchart : (Refer Flowchart 4)



Flowchart 4

➢Program

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

```
#include<conio.h>
void main()
{
int i,n,fact;
clrscr();
printf("Enter a number: ");
scanf("%d",&n);
for(i=1,fact=1;i<=n;i++)
{
fact=fact * i;
}
printf("Factorial of this number is:%d\n",fact);
getch();
}
```

Output

```
Enter a number: 4
Factorial of this number is:24
```

➢Program 3.2.5 :Write a program to display the multiplication table of a user entered number. The table must be upto 10.

➢Algorithm

Step I :START.

Step II :PRINT "Enter a number".

Step III:INPUT n.

Step IV :i = 1.

Step V :IF i > 10 THEN GOTO step IX.

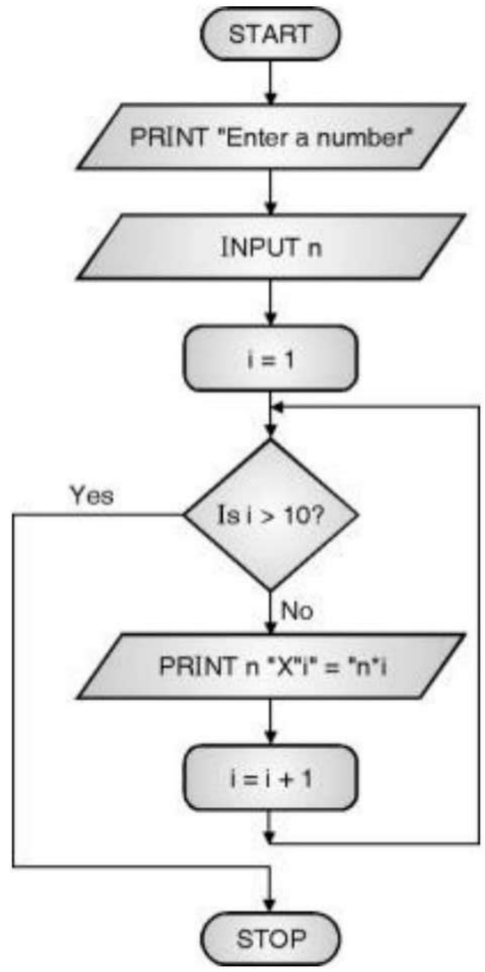
Step VI :PRINT n " X " i " = " n * i.

Step VII :i = i + 1

Step VIII:GOTO step V

Step IX :STOP.

➢Flowchart : (Refer Flowchart 5)



Flowchart 5

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int i,n;
clrscr();
printf("Enter a number: ");
scanf("%d",&n);
for(i=1;i<=10;i++)
{
printf("%d X %d = %d\n",n,i,(n*i));
}
getch();
}
```

Output

```
Enter a number: 5
5 X 1 = 5
5 X 2 = 10
5 X 3 = 15
5 X 4 = 20
```

```
5 X 5 = 25
5 X 6 = 30
5 X 7 = 35
5 X 8 = 40
5 X 9 = 45
5 X 10 = 50
```

➢Program 3.2.6 :Write a program to display first n odd numbers.

➢Algorithm

Step I :START.

Step II :PRINT "Enter a number".

Step III :INPUT n.

Step IV :i = 0

Step V :IF i > n THEN GOTO step IX.

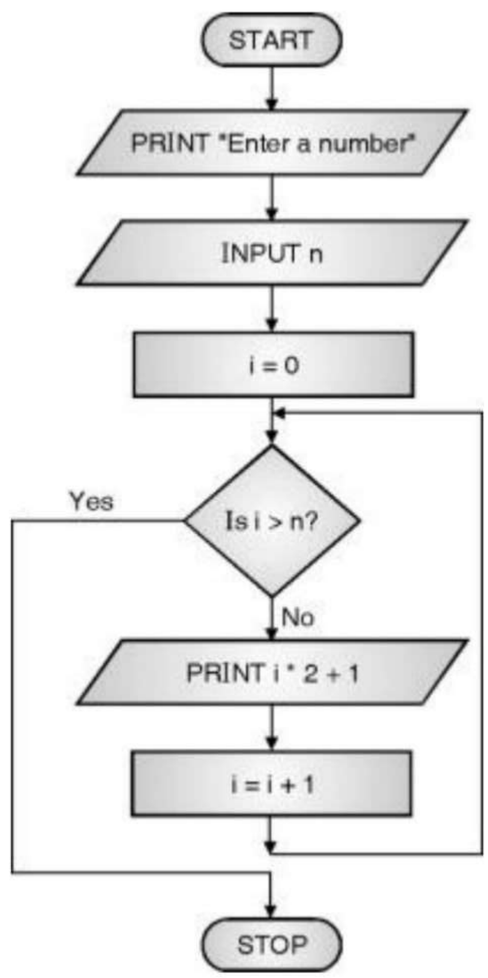
Step VI :PRINT i * 2 + 1.

Step VII :i = i + 1

Step VIII :GOTO step V

Step IX :STOP.

➢Flowchart : (Refer Flowchart 6)



Flowchart 6

➢Program

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

```
void main()
{
int i,n;
clrscr();
printf("Enter a number: ");
scanf("%d",&n);
for(i=0;i<=n-1;i++)
{
printf("%d\n",2*i+1);
}
getch();
}
```

Output

```
Enter a number: 6
1
3
5
7
9
11
```

➢Program 3.2.7 : Write a program to display odd numbers upto n.

➢Algorithm

Step I :START.

Step II :PRINT "Enter a number".

Step III :INPUT n.

Step IV :i = 1

Step V :IF i > n THEN GOTO step IX.

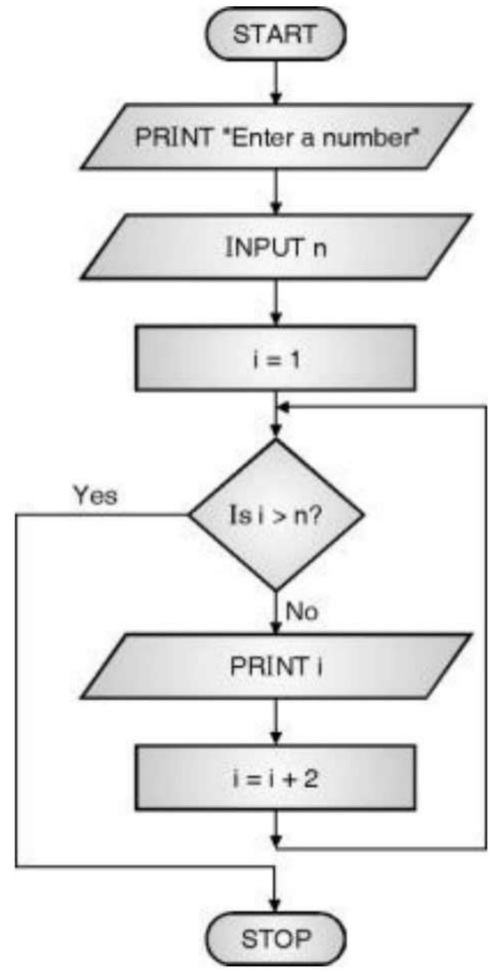
Step VI :PRINT i.

Step VII :i = i + 2

Step VIII :GOTO step V

Step IX :STOP.

➢Flowchart : (Refer Flowchart 7)



Flowchart 7

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int i,n;
clrscr();
printf("Enter a number: ");
scanf("%d",&n);
for(i=1;i<=n;i+=2)
{
printf("%d\n",i);
}
getch();
}
```

Output

```
Enter a number: 10
1
3
5
7
```

```
9
```

➢Program 3.2.8 :Write a program to calculate and display the sum of first n natural numbers.

➢Algorithm

Step I :START.

Step II:PRINT "Enter a number".

Step III :INPUT n.

Step IV :i = 1, sum = 0.

Step V :IF i > n, THEN GOTO step IX.

Step VI :sum = sum + i.

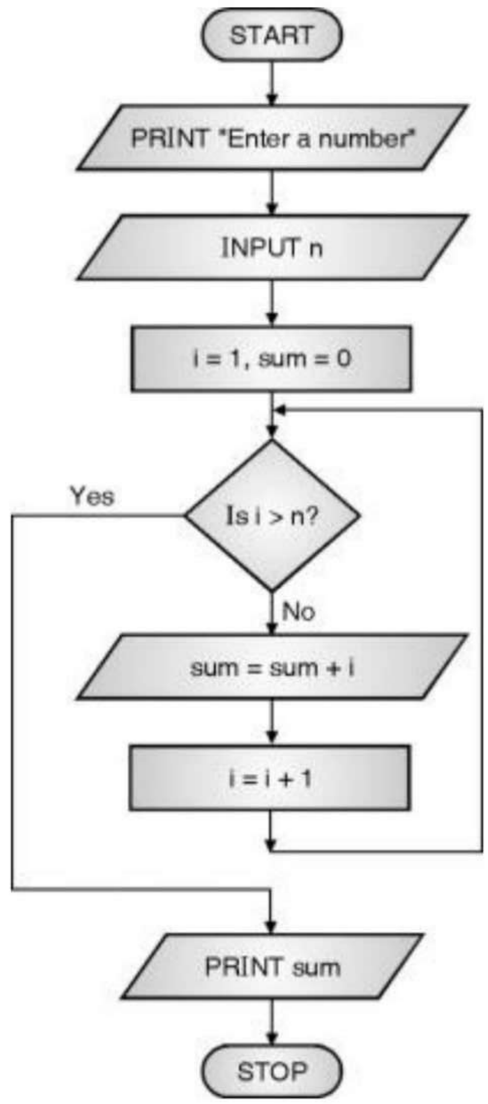
Step VII :i = i + 1

Step VIII :GOTO step V.

Step IX :PRINT sum.

Step X :STOP.

➢Flowchart : (Refer Flowchart 8)



Flowchart 8

➢Program

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

```
int i,n,sum=0;
clrscr();
printf("Enter a number: ");
scanf("%d",&n);
for(i=1;i<=n;i++)
{
sum=sum+i;
}
printf("Sum= %d\n",sum);
getch();
}
```

Output

```
Enter a number: 10
Sum= 55
```

➢Program 3.2.9 :Write a program to calculate and display the sum of the square of first n natural numbers.

➢Algorithm

Step I :START.

Step II:PRINT "Enter a number".

Step III :INPUT n.

Step IV :i = 1, sum = 0.

Step V :IF i > n, THEN GOTO step IX.

Step VI :sum = sum + i * i.

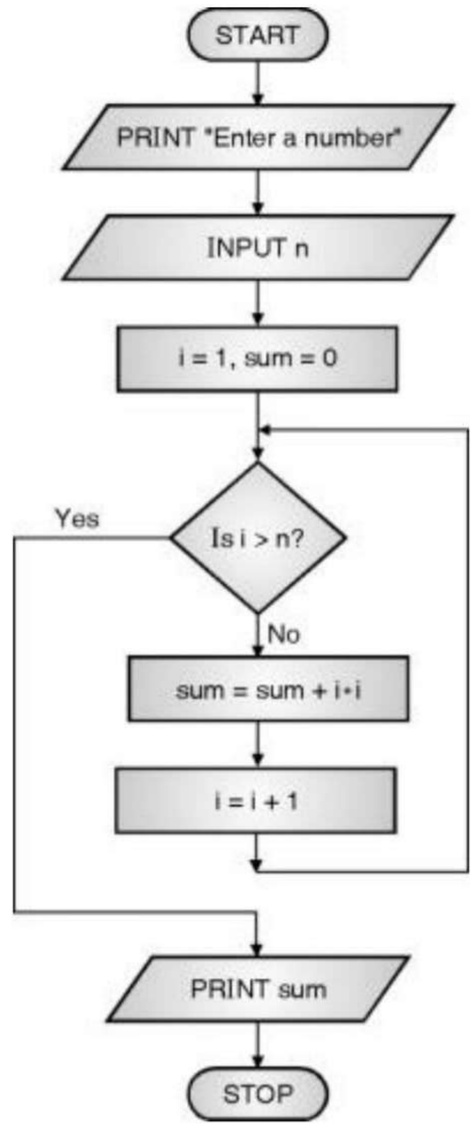
Step VII :i = i + 1

Step VIII :GOTO step V.

Step IX :PRINT sum.

Step X :STOP.

➢Flowchart : (Refer Flowchart 9)



Flowchart 9

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int i,n,sum=0;
clrscr();
printf("Enter a number: ");
scanf("%d",&n);
for(i=1;i<=n;i++)
{
sum=sum+i*i;
}
printf("Sum= %d\n",sum);
getch();
}
```

Output

```
Enter a number: 5
```

Sum= 55

➢Program 3.2.10 :Write a program to display first n elements of Fibonacci series.

Note : Fibonacci series is a series wherein the first two elements are 0 and 1. Thereafter the next values are the sum of previous two elements. Hence it can be said that the series is 0,1,1,2,3,5,8,13,21,34,55 . . .

➢Algorithm

Step I :START.

Step II:PRINT "Enter a number".

Step III :INPUT n.

Step IV :i = 1, a = 0, b = 1

Step V:PRINT a, b

Step VI :IF i > n – 2, THEN GOTO step XII.

Step VII :c = a + b.

Step VIII :PRINT c

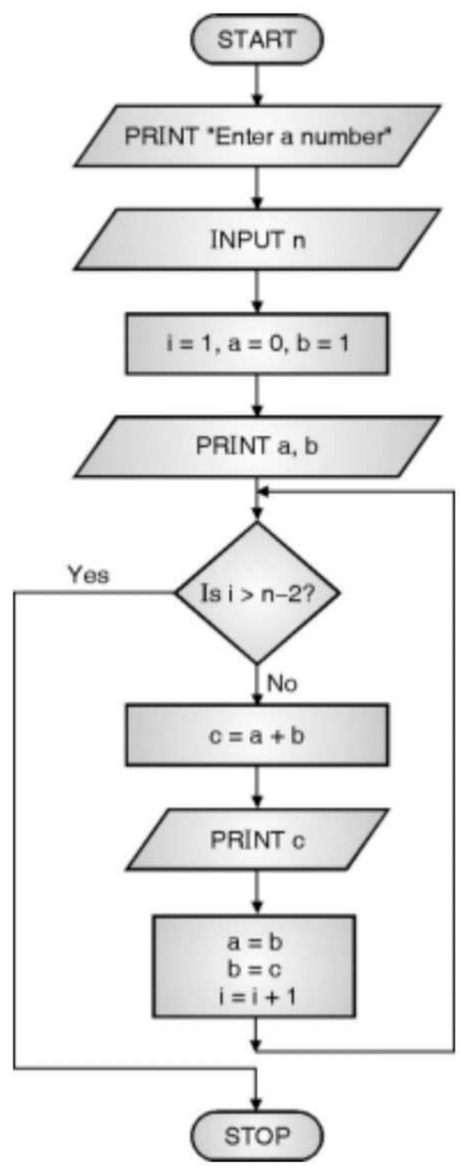
Step IX :a = b, b = c.

Step X :i = i + 1

Step XI :GOTO step VI.

Step XII :STOP.

➢Flowchart : (Refer Flowchart 10)



Flowchart 10

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int a=0,b=1,c,i,n;
clrscr();
printf("Enter a number: ");
scanf("%d",&n);
printf("Fibonacci Series\n0\n1\n");
for(i=1;i<=n–2;i++)
{
c=a+b;
printf("%d\n",c);
a=b;
b=c;
}
getch();
}
```

Output

```
Enter a number: 10
Fibonacci Series
1
1
2
3
5
8
13
21
34
```

➢Program 3.2.11 :Write a program to calculate the value of the following series.

$$1 + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \ldots + \frac{1}{n}$$

➢Algorithm

Step I :START.

Step II :PRINT "Enter the value of n".

Step III :INPUT n.

Step IV :i = 1, sum = 0.

Step V :IF i > n, THEN GOTO step IX.

Step VI:sum = sum + 1 / i.

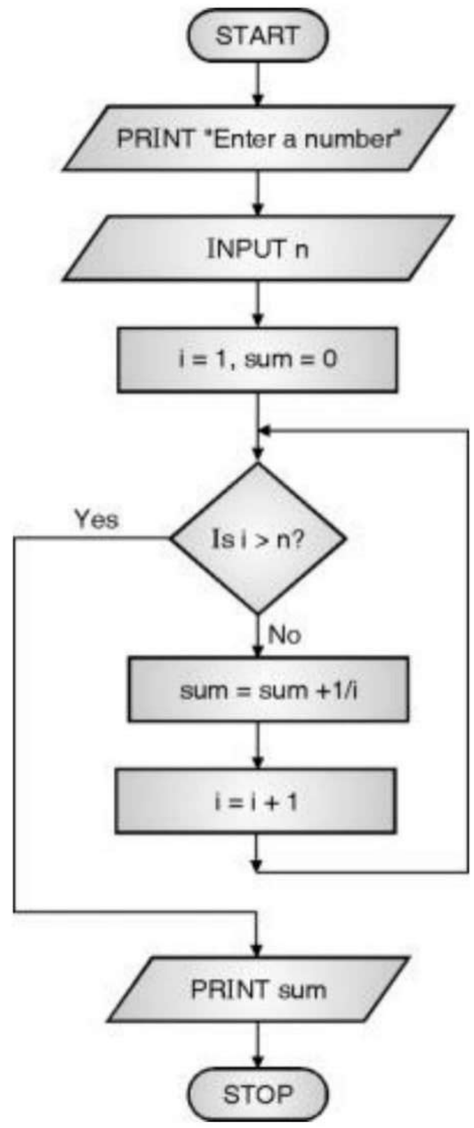
Step VII :i = i + 1

Step VIII:GOTO step V.

Step IX :PRINT sum.

Step X :STOP.

➢Flowchart : (Refer Flowchart 11)



Flowchart 11

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int i,n;
float sum=0.0;
clrscr();
printf("Enter a number: ");
scanf("%d",&n);
```

```
for(i=1;i<=n;i++)
{
sum=sum+1.0/i;
}
printf("The value of the series is:%f",sum);
getch();
}
```

Output

```
Enter a number: 4
The value of the series is:2.083333
```

➢Program 3.2.12 :Write a program to calculate the value of the following series.

$$1 + \frac{1}{2!} + \frac{1}{3!} + \frac{1}{4!} + \ldots + \frac{1}{n!}$$

➢Algorithm

Step I :START.

Step II :PRINT "Enter the value of n".

Step III :INPUT n.

Step IV :i = 1, fact = 1, sum = 0.

Step V :IF i > n, THEN GOTO step IX.

Step VI :fact = fact * i

sum = sum + 1 / fact.

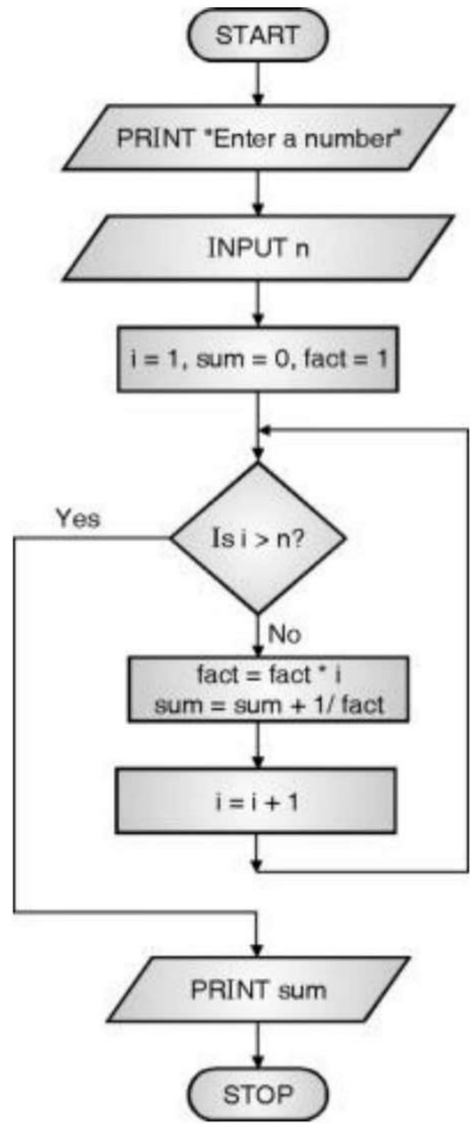
Step VII :i = i + 1

Step VIII :GOTO step V.

Step IX :PRINT sum.

Step X :STOP.

➢Flowchart : (Refer Flowchart 12)



Flowchart 12

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int i,n,fact=1;
float sum=0.0;
clrscr();
printf("Enter a number: ");
scanf("%d",&n);
for(i=1;i<=n;i++)
{
fact=fact*i;
sum=sum+1.0/fact;
}
printf("The value of the series is:%f",sum);
getch();
}
```

Output

```
Enter a number: 5
The value of the series is:1.716667
```

➢Program 3.2.13 :Write a program to calculate the sine of an angle using the following series for x as the angle in radians.

$$\text{Sin } x = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \dots \frac{x^n}{n!}$$

Accept the angle from user in degrees and display the result correct upto 6 decimal places

➢Algorithm

Step I :START.

Step II :PRINT "Enter the value of x in degrees".

Step III :INPUT x.

Step IV :x = x / 180 * 3.14

Step V :term = sum = x.

Step VI :i = 1, fact = 1, sign = – 1.

Step VII :IF term < 0.0000001, THEN GOTO step XIV.

Step VIII :fact = fact * i * i – 1.

Step IX :num = x^i, term = num / fact

Step X :sum = sum + term * sign.

Step XI :sign = sign * – 1

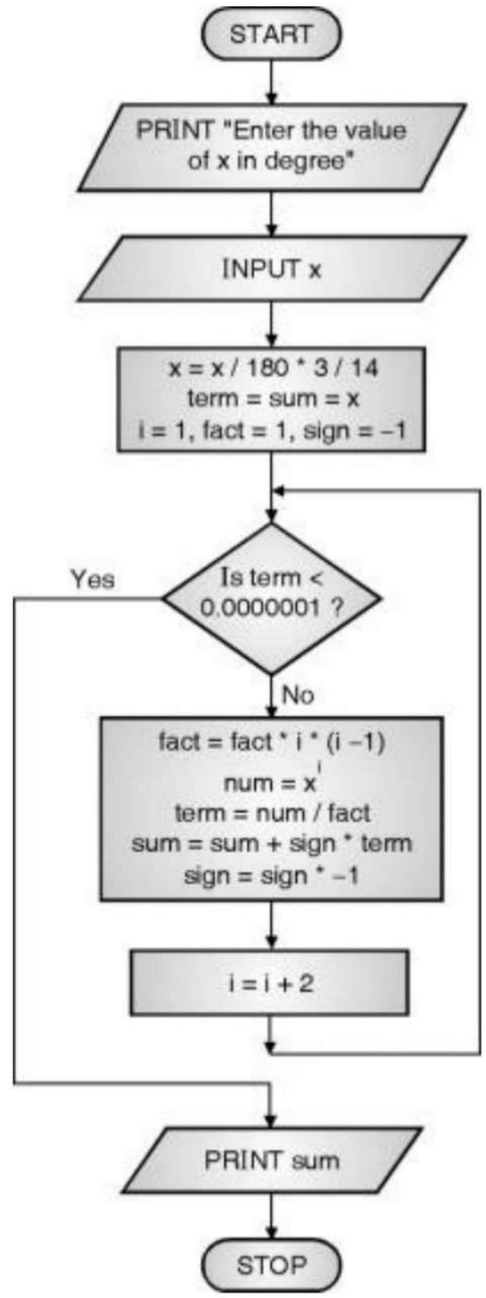
Step XII :i = i + 2.

Step XIII :GOTO step VII.

Step XIV :PRINT sum.

Step XV :STOP.

➢Flowchart : (Refer Flowchart 13)



Flowchart 13

➢Program

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

```
#include<conio.h>
```

```
void main()
{
int i, fact=1,sign=–1;
float x,numerator,sum,term;
clrscr();
printf("Enter an angle in degrees: ");
scanf("%f",&x);
x=x*3.14/180;
term=x;
sum=term;
for(i=3;term>=0.0000001;i=i+2)
```

```
{
fact=fact*i*(i-1);
numerator=pow(x,i);
term=numerator/fact;
sum=sum + sign * term;
sign=sign*-1;
}
printf("The value of the series is:%f",sum);
getch();
}
```

Output

```
Enter an angle in degrees: 45
The value of the series is:0.706821
```

➢Program 3.2.14 :Write a program to calculate the sine of an angle using the following series for x as the angle in radians.

$$\sin x = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \dots \frac{x^n}{n!}$$

Accept the angle in degrees and value of n from user.

➢Algorithm

Step I :START.

Step II :PRINT "Enter the value of x in degrees and the value of n".

Step III :INPUT x, n.

Step IV :x = x / 180 * 3.14

Step V :term = sum = x.

Step VI :i = 1, fact = 1, sign = – 1.

Step VII :IF i > n THEN GOTO step XIV.

Step VIII :fact = fact * i * i – 1.

Step IX :num = x^i, term = num / fact

Step X :sum = sum + term * sign.

Step XI :sign = sign * – 1

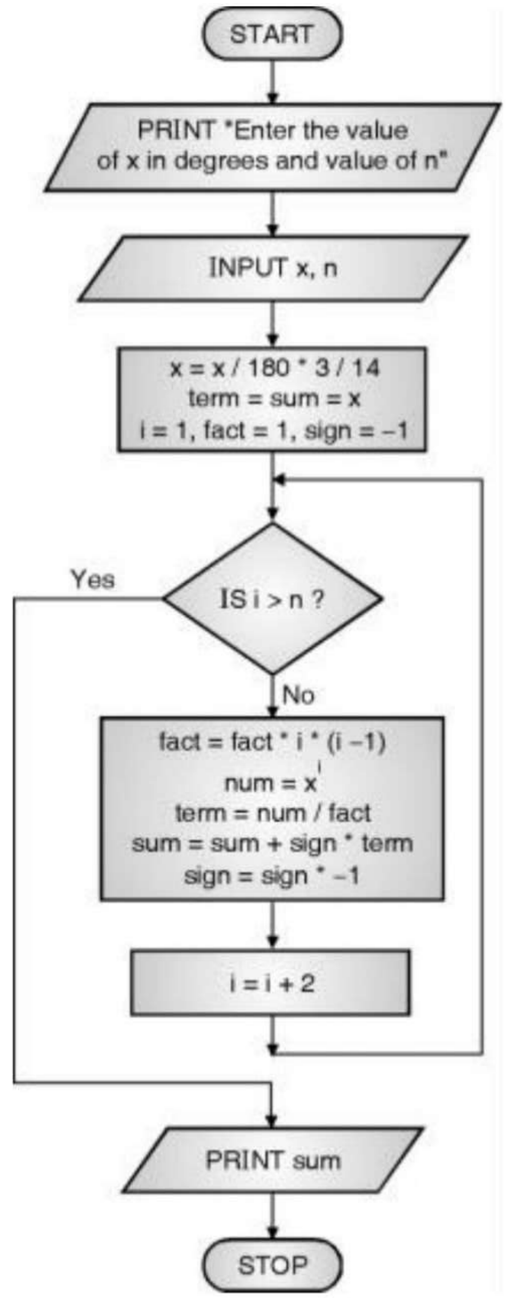
Step XII :i = i + 2.

Step XIII :GOTO step VII.

Step XIV :PRINT sum.

Step XV :STOP.

➢Flowchart : (Refer Flowchart 14)



Flowchart 14

➢Program

```
#include<stdio.h>
#include<math.h>
#include<conio.h>
void main()
{
int i, n, fact=1,sign=–1;
float x,numerator,sum,term;
clrscr();
printf("Enter an angle in degrees and the
value of n:  ");
scanf("%f %d",&x,&n);
x=x*3.14/180;
term=x;
sum=term;
for(i=3;i<=n;i=i+2)
{
```

```
fact=fact*i*(i-1);
numerator=pow(x,i);
term=numerator/fact;
sum=sum + sign * term;
sign=sign*-1;
}
printf("The value of the series is:%f",sum);
getch();
}
```

Output

```
Enter an angle in degrees and the value of n: 45
10
The value of the series is:0.706821
```

➢Program 3.2.15 :Write a program to display the square, cube, fourth power, ...nth power of x, where x is an integer taken from user.

➢Algorithm

Step I :START.

Step II:PRINT "Enter the values of x and n".

Step III :INPUT x, n.

Step IV :i = 1.

Step V :IF i > n, THEN GOTO step IX .

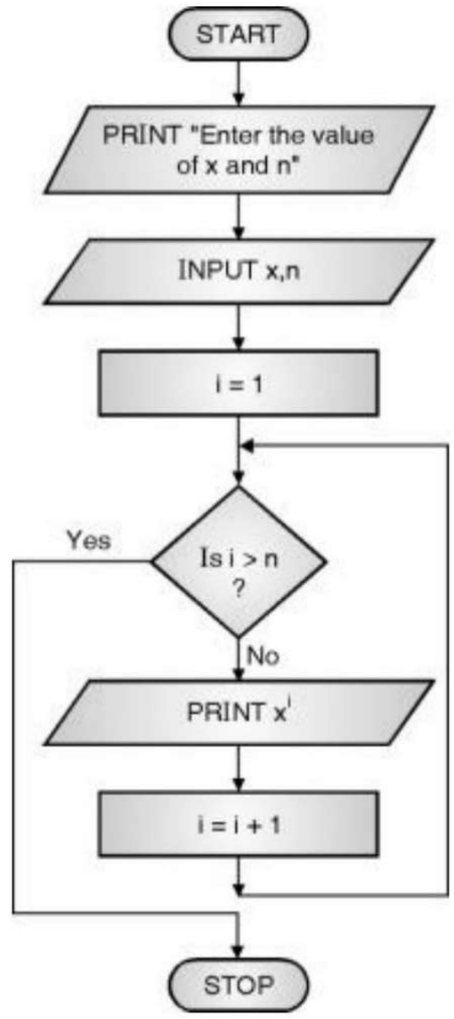
Step VI :PRINT x^i.

Step VII :i = i + 1.

Step VIII :GOTO step V.

Step IX :STOP.

➢Flowchart : (Refer Flowchart 15)



Flowchart 15

➢Program

```
#include<stdio.h>
#include<math.h>
#include<conio.h>
void main()
{
int i,n;
float x;
clrscr();
printf("Enter the value of x and the value of n: ");
scanf("%f %d",&x,&n);
for(i=2;i<=n;i++)
{
printf("%f raised to %d is %f\n",x,i,pow(x,i));
}
getch();
}
```

Output

```
Enter the value of x and the value of n: 3
5
```

```
3 raised to 2 is 9
3 raised to 3 is 27
3 raised to 4 is 81
3 raised to 5 is 243
```

➢Program 3.2.16 :Write a program to display the value of s for t=1, 5, 10, 15, 20, ...100.

$$s = s_0 + v_0 + \frac{1}{2} a t^2$$

➢Algorithm

Step I :START.

Step II :PRINT "Enter the values of s0, v0 and a (i.e. initial displacement, velocity and acceleration)".

Step III:INPUT s0, v0, a.

Step IV :s = s0 + v0 + 0.5 * a

Step V:PRINT "1", s

Step VI :t = 5.

Step VII :IF t > 100, THEN GOTO step XII .

Step VIII :s = s0 + v0 + 0.5 * a * t * t.

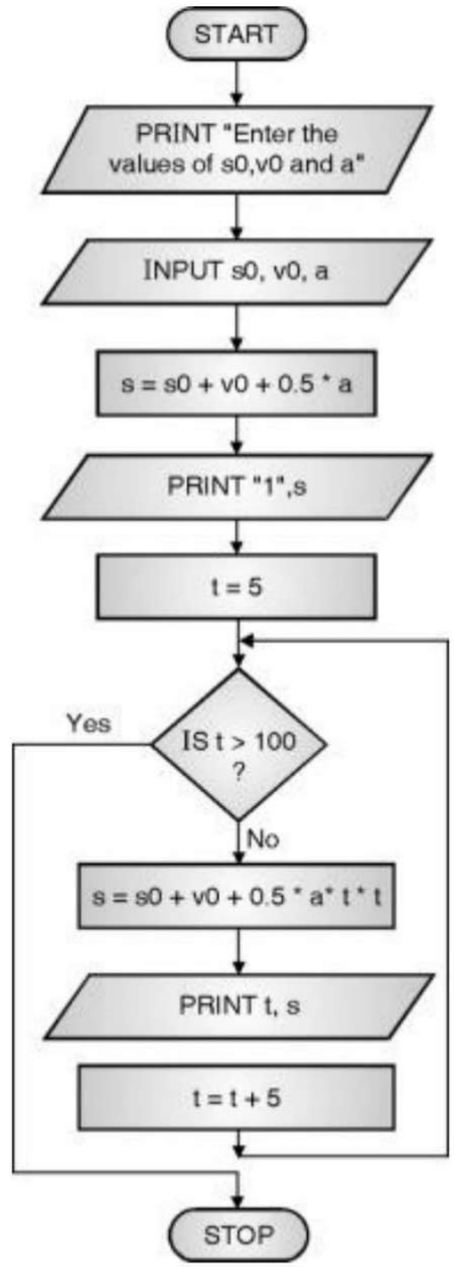
Step IX:PRINT t, s

Step X :t = t + 5.

Step XI:GOTO step VII.

Step XII :STOP.

➢Flowchart : (Refer Flowchart 16)



Flowchart 16

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int t;
float s,s0,v0,a;
clrscr();
printf("Enter the values of s0, v0 and a: ");
scanf("%f %f %f",&s0,&v0,&a);
printf("t\tS\n1\t%d\n",s0+v0+0.5*a);
for(t=5;t<=100;t+=5)
{
printf("%d\t%f\n",t,s0+v0+0.5*a*t*t);
}
getch();
}
```

Output

```
Enter the values of s0, v0 and a: 2 3 9.8
t     S
1     9.9
5     127.500002
10    495.00001
15    1107.500021
20    1965.000038
25    3067.50006
30    4415.000086
35    6007.500117
40    7845.000153
45    9927.500193
50    12255.000238
55    14827.500288
60    17645.000343
65    20707.500403
70    24015.000467
75    27567.500536
80    31365.00061
85    35407.500689
90    39695.000772
95    44227.500861
100   49005.000954
```

➢Program 3.2.17 : Write a program to display "Hi" twice in a line and five such lines.

➢Algorithm

Step I :START.

Step II :i = 1.

Step III :IF i > 5, THEN GOTO step XII.

Step IV :j = 1.

Step V :IF j > 2, THEN GOTO step IX

Step VI :PRINT "Hi"

Step VII :j = j + 1

Step VIII :GOTO step V

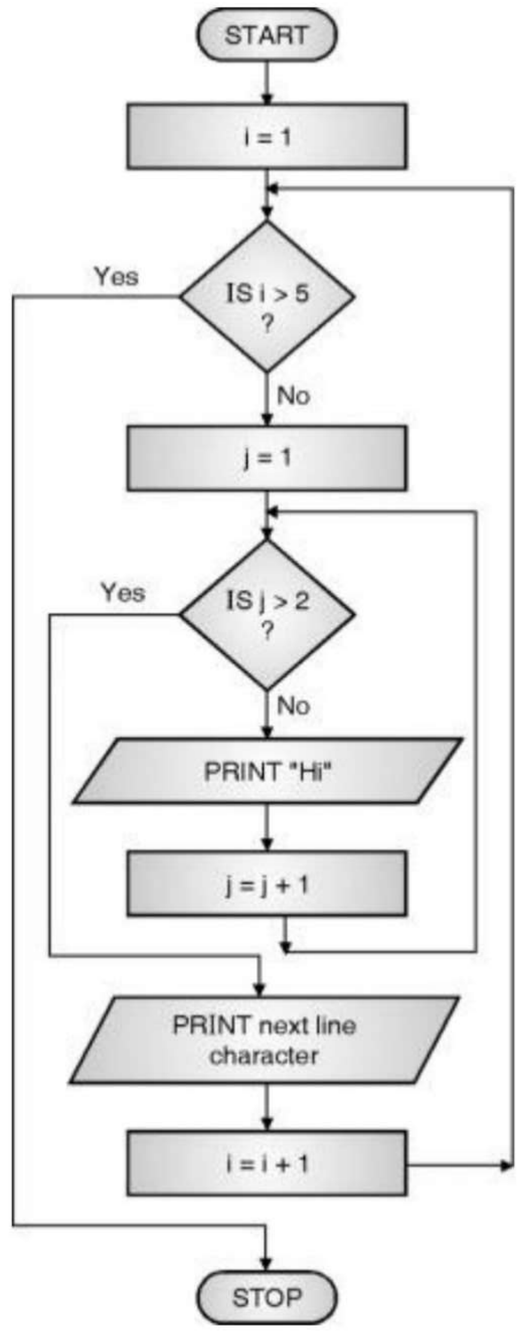
Step IX :PRINT next line character.

Step X :i = i + 1.

Step XI :GOTO step III.

Step XII :STOP.

➢Flowchart : (Refer Flowchart 17)



Flowchart 17

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int i,j;
clrscr();
for(i=1;i<=5;i++)
{
for(j=1;j<=2;j++)
{
printf("Hi ");
}
printf("\n");
```

```
}
getch();
}
```

Output

```
Hi Hi
Hi Hi
Hi Hi
Hi Hi
Hi Hi
```

➢Program 3.2.18 : Find the output of the following program.

```
#include<stdio.h>
#include<conio.h>
void main()
{
int i,j;
clrscr();
for(i=1;i<=5;i++)
{
for(j=1;j<=2;j++)
{
printf("Hello ");
}
printf("Hi \n");
}
getch();
}
```

Output

```
Hello Hello Hi
Hello Hello Hi
Hello Hello Hi
Hello Hello Hi
Hello Hello Hi
```

➢Program 3.2.19 : Write a program to display the following

Hello Hello Hello

Hello Hello Hello

➢Algorithm

Step I :START.

Step II :i = 1.

Step III :IF i > 2, THEN GOTO step XII .

Step IV :j = 1.

Step V :IF j > 3, THEN GOTO step IX

Step VI :PRINT "Hello"

Step VII :j = j + 1

Step VIII:GOTO step V

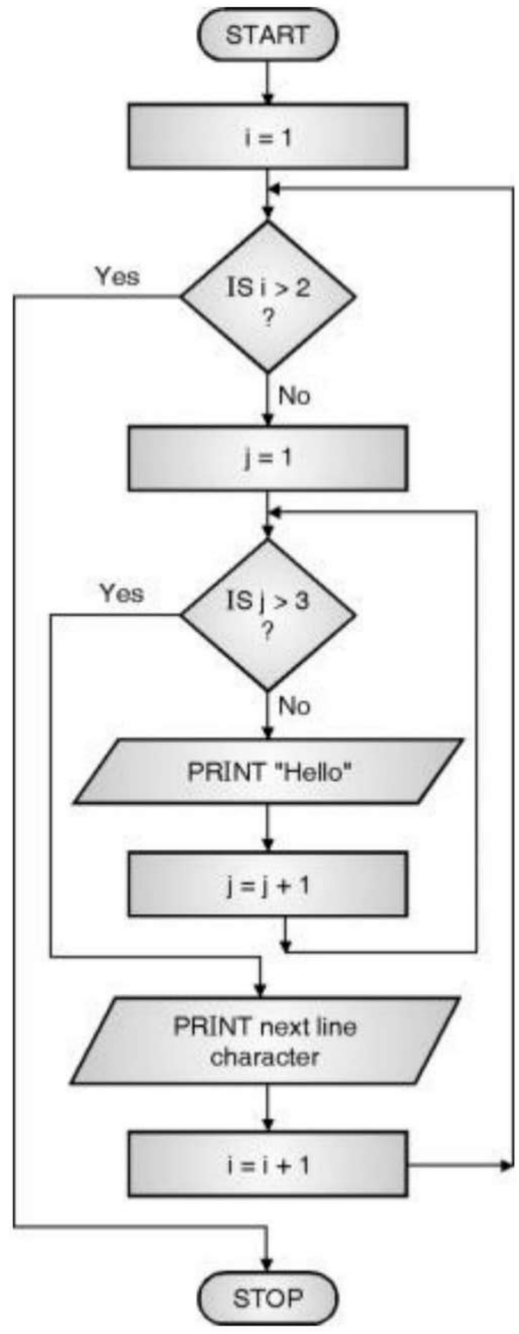
Step IX :PRINT next line character.

Step X :i = i + 1.

Step XI :GOTO step III.

Step XII :STOP.

➢Flowchart : (Refer Flowchart 18)



Flowchart 18

➢Program

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

```
#include<conio.h>
void main()
{
int i,j;
clrscr();
for(i=1;i<=2;i++)
{
for(j=1;j<=3;j++)
{
printf("Hello ");
}
printf("\n");
}
getch();
}
```

Output

```
Hello Hello Hello
Hello Hello Hello
```

➢Program 3.2.20 : Write a program to display the following

1

11

111

1111

11111 *(May 2014)*

➢Algorithm

Step I :START.

Step II :i = 1.

Step III :IF i > 5, THEN GOTO step XII .

Step IV :j = 1.

Step V :IF j > i, THEN GOTO step IX

Step VI :PRINT "1"

Step VII :j = j + 1

Step VIII :GOTO step V

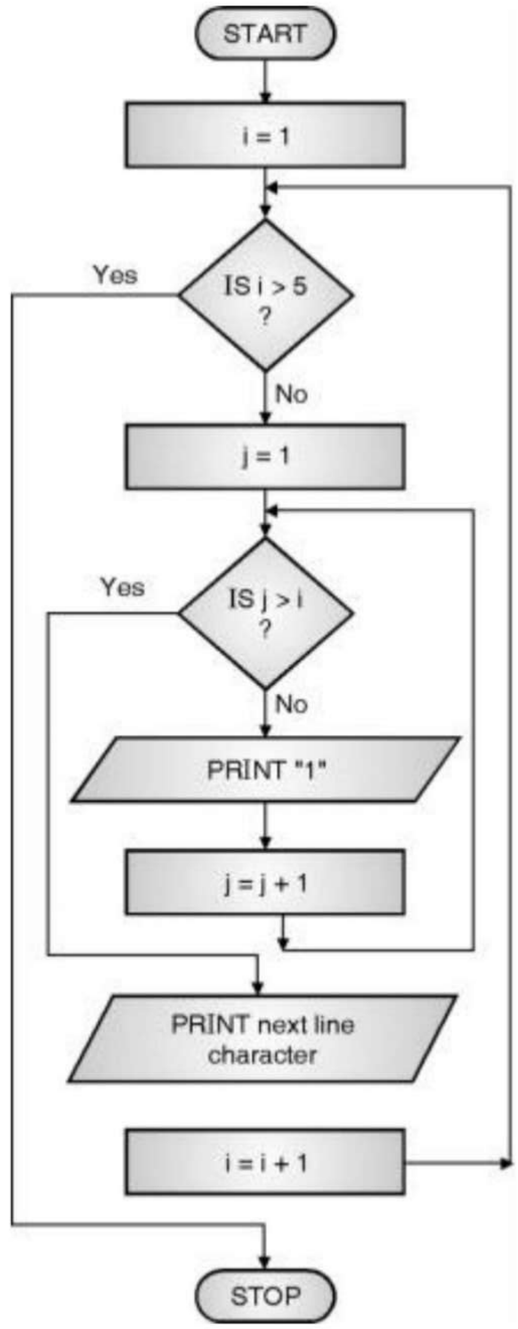
Step IX :PRINT next line character.

Step X :i = i + 1.

Step XI :GOTO step III.

Step XII :STOP.

➢Flowchart : (Refer Flowchart 19)



Flowchart 18

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int i,j;
clrscr();
for(i=1;i<=5;i++)
{
for(j=1;j<=i;j++)
{
printf("1");
}
printf("\n");
}
```

```
getch();
}
```

Output

```
1
11
111
1111
11111
```

➢Program 3.2.21 : Write a program to display the following :

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

(Dec. 2015)

➢Algorithm

Step I :START.

Step II :i = 1.

Step III :IF i > 5, THEN GOTO step XII .

Step IV :j = 1.

Step V :IF j > i, THEN GOTO step IX

Step VI :PRINT "*"

Step VII :j = j + 1

Step VIII :GOTO step V

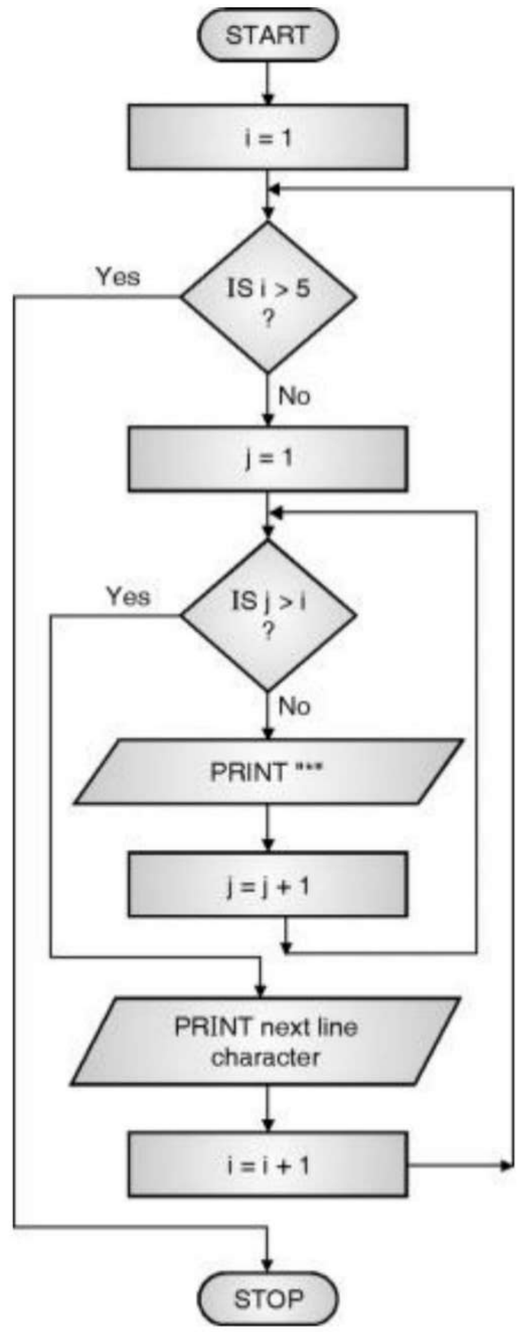
Step IX :PRINT next line character.

Step X :i = i + 1.

Step XI :GOTO step III.

Step XII :STOP.

➢Flowchart : (Refer Flowchart 20)



Flowchart 20

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int i,j;
clrscr();
for(i=1;i<=5;i++)
{
for(j=1;j<=i;j++)
{
printf("*");
}
printf("\n");
}
getch();
```

```
}
```

Output

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

➢Program 3.2.22 :Write a program to display the following for the user specified number of lines

```
*
**
***
****
*****
|
|
n lines
```

➢Algorithm

Step I :START.

Step II:PRINT "Enter the number of lines"

Step III:INPUT n

Step IV :i = 1.

Step V :IF i > n, THEN GOTO step XIV .

Step VI :j = 1.

Step VII :IF j > i, THEN GOTO step XI

Step VIII :PRINT "*"

Step IX :j = j + 1

Step X :GOTO step VII

Step XI :PRINT next line character.

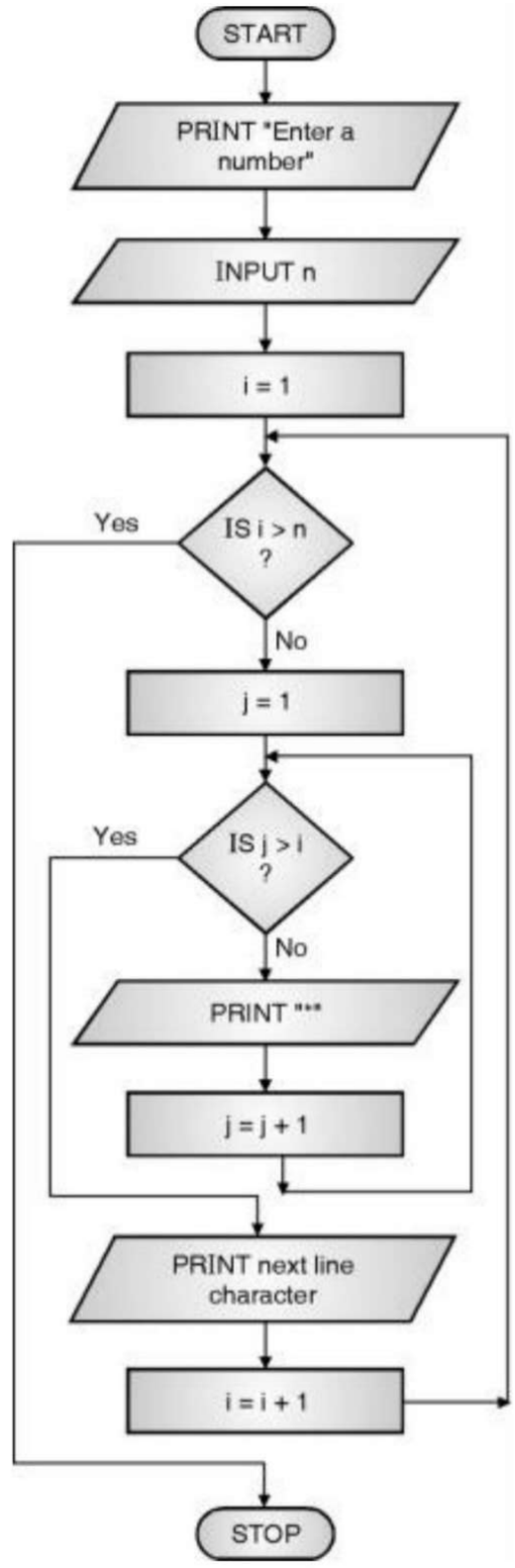
Step XII :i = i + 1.

Step XIII :GOTO step V.

Step XIV :STOP.

For other patterns the flowchart will be in similar manner.

➢Flowchart : (Refer Flowchart 21)



Flowchart 21

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int i,j,n;
clrscr();
printf("Enter the number of lines:");
scanf("%d",&n);
for(i=1;i<=n;i++)
{
for(j=1;j<=i;j++)
```

```
{
printf("*");
}
printf("\n");
}
getch();
}
```

Output

```
Enter the number of lines:7
*
**
***
****
*****
******
*******
```

➢Program 3.2.23 : Write a program to display the following for the user specified number of lines

```
*
**
***
****
*****
******
*******
********
|
|
n lines
```

➢Algorithm

Step I :START.

Step II:PRINT "Enter the number of lines"

Step III:INPUT n

Step IV :i = 1.

Step V :IF i > n, THEN GOTO step XIX.

Step VI:j = 1

Step VII:IF j > n – i, THEN GOTO step XI

Step VIII:PRINT " "

Step IX:j = j + 1

Step X:GOTO step VII

Step XI :j = 1.

Step XII :IF j > i, THEN GOTO step XVI

Step XIII :PRINT "*"

Step XIV :j = j + 1

Step XV :GOTO step XII

Step XVI :PRINT next line character.

Step XVII :i = i + 1.

Step XVIII :GOTO step V.

Step XIX :STOP.

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int i,j,n;
clrscr();
printf("Enter the number of lines:");
scanf("%d",&n);
for(i=1;i<=n;i++)
{
for(j=1;j<=n–i;j++)
{
printf(" ");
}
for(j=1;j<=i;j++)
{
printf("*");
}
printf("\n");
}
getch();
}
```

Output

```
Enter the number of lines:8
*
```

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

➢Program 3.2.24 :Write a program to display the following asking the user for the number of "*" in the largest line

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

➢Algorithm

Step I :START.

Step II:PRINT "Enter the number of lines"

Step III:INPUT n

Step IV :i = 1.

Step V :IF i > n, THEN GOTO step XIX.

Step VI:j = 1

Step VII:IF j > n – i, THEN GOTO step XI

Step VIII:PRINT " "

Step IX:j = j + 1

Step X:GOTO step VII

Step XI :j = 1.

Step XII :IF j > i, THEN GOTO step XVI

Step XIII :PRINT "*"

Step XIV :j = j + 1

Step XV :GOTO step XII

Step XVI :PRINT next line character.

Step XVII :i = i + 1.

Step XVIII:GOTO step V.

Step XIX :i = n – 1.

Step XX :IF i < 1, THEN GOTO step XXXIV.

Step XXI:j = 1

Step XXII:IF j > n – i, THEN GOTO step XXVI

Step XXIII:PRINT " "

Step XXIV:j = j + 1

Step XXV:GOTO step XXII

Step XXVI :j = 1.

Step XXVII:IF j > i, THEN GOTO step XXXI

Step XXVIII :PRINT "*"

Step XXIX :j = j + 1

Step XXX :GOTO step XXVII

Step XXXI :PRINT next line character.

Step XXXII:i = i – 1.

Step XXXIII:GOTO step XX.

Step XXXIV:STOP.

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int i,j,n;
clrscr();
printf("Enter the number of * in the centre line:");
scanf("%d",&n);
for(i=1;i<=n;i++)
{
for(j=1;j<=n–i;j++)
{
printf(" ");
}
for(j=1;j<=i;j++)
{
printf("*");
}
printf("\n");
}
```

```
for(i=n-1;i>=1;i- -)
{
for(j=1;j<=n-i;j++)
{
printf(" ");
}
for(j=1;j<=i;j++)
{
printf("*");
}
printf("\n");
}
getch();
}
```

Output

```
Enter the number of * in the centre line:6
*
**
***
***
****
*****
******
*****
****
***
**
*
```

➢Program 3.2.25 :Write a program to display the following asking the user for the number of "*" in the largest line ***(May 2014)***

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

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

➢Algorithm

Step I :START.

Step II:PRINT "Enter the number of lines"

Step III:INPUT n

Step IV :i = 1.

Step V :IF i > n, THEN GOTO step XIX.

Step VI:j = 1

Step VII:IF j > n – i, THEN GOTO step XI

Step VIII:PRINT " "

Step IX:j = j + 1

Step X:GOTO step VII

Step XI :j = 1.

Step XII :IF j > i, THEN GOTO step XVI

Step XIII :PRINT "* "

Step XIV :j = j + 1

Step XV :GOTO step XII

Step XVI :PRINT next line character.

Step XVII :i = i + 1.

Step XVIII :GOTO step V.

Step XIX :i = n – 1.

Step XX :IF i < 1, THEN GOTO step XXXIV.

Step XXI:j = 1

Step XXII:IF j > n – i, THEN GOTO step XXVI

Step XXIII:PRINT " "

Step XXIV:j = j + 1

Step XXV:GOTO step XXII

Step XXVI :j = 1.

Step XXVII:IF j > i, THEN GOTO step XXXI

Step XXVIII:PRINT "* "

Step XXIX :j = j + 1

Step XXX :GOTO step XXVII

Step XXXI :PRINT next line character.

Step XXXII:i = i – 1.

Step XXXIII:GOTO step XX.

Step XXXIV:STOP.

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int i,j,n;
clrscr();
printf("Enter the number of * in the centre line:");
scanf("%d",&n);
for(i=1;i<=n;i++)
{
for(j=1;j<=n–i;j++)
{
printf(" ");
}
for(j=1;j<=i;j++)
{
printf("* ");
}
printf("\n");
}
for(i=n–1;i>=1;i– –)
{
for(j=1;j<=n–i;j++)
{
printf(" ");
}
for(j=1;j<=i;j++)
{
printf("* ");
}
printf("\n");
}
getch();
}
```

Output

```
Enter the number of * in the centre line:7
*
 **
 ***
 ****
*****
 ******
 *******
 ******
  *****
 ****
 ***
 **
 *
```

➢Program 3.2.26 : Write a program to display the following asking the user for the number of lines in the upper half.

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

➢Algorithm

Step I :START.

Step II:PRINT "Enter the number of lines"

Step III:INPUT n

Step IV :i = 1.

Step V :IF i > n, THEN GOTO step XIX.

Step VI:j = 1

Step VII:IF j > n – i, THEN GOTO step XI

Step VIII:PRINT " "

Step IX:j = j + 1

Step X:GOTO step VII

Step XI :j = 1.

Step XII :IF j > 2 * i – 1, THEN GOTO step XVI

Step XIII :PRINT "*"

Step XIV :j = j + 1

Step XV :GOTO step XII

Step XVI :PRINT next line character.

Step XVII :i = i + 1.

Step XVIII :GOTO step V.

Step XIX :i = n – 1.

Step XX :IF i < 1, THEN GOTO step XXXIV.

Step XXI:j = 1

Step XXII:IF j > n – i, THEN GOTO step XXVI

Step XXIII:PRINT " "

Step XXIV:j = j + 1

Step XXV:GOTO step XXII

Step XXVI :j = 1.

Step XXVII:IF j > 2 * i – 1, THEN GOTO step XXXI

Step XXVIII:PRINT "*"

Step XXIX :j = j + 1

Step XXX :GOTO step XXVII

Step XXXI :PRINT next line character.

Step XXXII:i = i – 1.

Step XXXIII:GOTO step XX.

Step XXXIV : STOP.

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int i,j,n;
clrscr();
printf("Enter the number of lines in the upper half:");
scanf("%d",&n);
for(i=1;i<=n;i++)
{
for(j=1;j<=n–i;j++)
{
printf(" ");
}
```

```
for(j=1;j<=2*i–1;j++)
{
printf("*");
}
printf("\n");
}
for(i=n–1;i>=1;i– –)
{
for(j=1;j<=n–i;j++)
{
printf(" ");
}
for(j=1;j<=2*i–1;j++)
{
printf("*");
}
printf("\n");
}
getch();
}
```

Output

```
Enter the number of lines in the upper half:5
*
 ***
    *****
   *******
  *********
   *******
    *****
     ***
      *
```

➢Program 3.2.27 :Write a program to display the following asking the user for the number of lines.

1

12

123

1234

12345

123456

➢Algorithm

Step I :START.

Step II:PRINT "Enter the number of lines"

Step III:INPUT n

Step IV :i = 1.

Step V :IF i > n, THEN GOTO step XIV .

Step VI :j = 1.

Step VII :IF j > i, THEN GOTO step XI

Step VIII :PRINT j

Step IX :j = j + 1

Step X :GOTO step VII

Step XI :PRINT next line character.

Step XII :i = i + 1.

Step XIII :GOTO step V.

Step XIV :STOP.

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int i,j,n;
clrscr();
printf("Enter the number of lines:");
scanf("%d",&n);
for(i=1;i<=n;i++)
{
for(j=1;j<=i;j++)
{
printf("%d",j);
}
printf("\n");
}
getch();
}
```

Output

```
Enter the number of lines:6
```

```
1
12
123
1234
12345
123456
```

➢Program 3.2.28 : Write a program to display the following asking the user for the number of lines.

```
1
121
12321
1234321
123454321
12345654321
```

➢Algorithm

Step I :START.

Step II:PRINT "Enter the number of lines"

Step III:INPUT n

Step IV :i = 1.

Step V :IF i > n, THEN GOTO step XXIXV

Step VI:j = 1

Step VII:IF j > n – i, THEN GOTO step XI

Step VIII:PRINT " "

Step IX:j = j + 1

Step X:GOTO step VII

Step XI :j = 1.

Step XII :IF j > i, THEN GOTO step XVI

Step XIII :PRINT j

Step XIV :j = j + 1

Step XV :GOTO step XII

Step XVI:j = i – 1

Step XVII:IF j < 1, THEN GOTO step XXI

Step XVIII:PRINT j

Step XIX:j = j – 1

Step XX:GOTO step XVII

Step XXI :PRINT next line character.

Step XXII :i = i + 1.

Step XXIII :GOTO step V.

Step XXIV:STOP.

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int i,j,n;
clrscr();
printf("Enter the number of lines:");
scanf("%d",&n);
for(i=1;i<=n;i++)
{
for(j=1;j<=n–i;j++)
{
printf(" ");
}
for(j=1;j<=i;j++)
{
printf("%d",j);
}
for(j=i–1;j>=1;j– –)
{
printf("%d",j);
}
printf("\n");
}
getch();
}
```

Output

```
Enter the number of lines:6
1
121
12321
1234321
123454321
12345654321
```

Explanation

- Here we have to perform 3 operations in each line and then next line i.e. printf("\n").
- The first operation is blank spaces. The second operation is printing j i.e. numbers from 1 to line number. The third operation is to print the numbers from i-1 (where i is line number) to 1. For each of these three operations there is an inner for loop.

➢Program 3.2.29 :Write a program to display the following asking the user for the number of lines.

A

ABA

ABCBA

ABCDCBA

ABCDEDCBA

ABCDEFEDCBA

➢Algorithm

Step I :START.

Step II:PRINT "Enter the number of lines"

Step III:INPUT n

Step IV :i = 1.

Step V :IF i > n, THEN GOTO step XXIXV

Step VI:j = 1

Step VII:IF j > n – i, THEN GOTO step XI

Step VIII:PRINT " "

Step IX:j = j + 1

Step X:GOTO step VII

Step XI :j = 1.

Step XII :IF j > i, THEN GOTO step XVI

Step XIII :PRINT (j + 64) type casted to character type data

Step XIV :j = j + 1

Step XV :GOTO step XII

Step XVI:j = i – 1

Step XVII:IF j < 1, THEN GOTO step XXI

Step XVIII:PRINT (j + 64) type casted to character type data

Step XIX:j = j – 1

Step XX:GOTO step XVII

Step XXI :PRINT next line character.

Step XXII :i = i + 1.

Step XXIII :GOTO step V.

Step XXIV:STOP.

➢Program

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

```
#include<conio.h>
void main()
{
int i,j,n;
clrscr();
printf("Enter the number of lines:");
scanf("%d",&n);
for(i=1;i<=n;i++)
{
for(j=1;j<=n–i;j++)
{
printf(" ");
}
for(j=1;j<=i;j++)
{
printf("%c",(char)(j+64));
}
for(j=i–1;j>=1;j– –)
{
printf("%c",(char)(j+64));
}
printf("\n");
}
getch();
}
```

Output

```
Enter the number of lines:6
A
ABA
ABCBA
ABCDCBA
ABCDEDCBA
ABCDEFEDCBA
```

➢Program 3.2.30 : Write a program to display the following asking the user for the number of lines.

```
1
2 3
4 5 6
```

7 8 9 10

Step I :START.

Step II:PRINT "Enter the number of lines"

Step III:INPUT n

Step IV :i = 1, k = 1.

Step V :IF i > n, THEN GOTO step XIV .

Step VI :j = 1.

Step VII :IF j > i, THEN GOTO step XI

Step VIII :PRINT k

Step IX :j = j + 1, k = k + 1

Step X :GOTO step VII

Step XI :PRINT next line character.

Step XII :i = i + 1.

Step XIII :GOTO step V.

Step XIV :STOP.

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int i,j,n,k;
clrscr();
printf("Enter the number of lines:");
scanf("%d",&n);
for(i=1,k=1;i<=n;i++)
{
for(j=1;j<=i;j++)
{
printf("%d ",k++);
}
printf("\n");
}
getch();
}
```

Output

```
Enter the number of lines:8
1
```

```
2 3
4 5 6
7 8 9 10
11 12 13 14 15
16 17 18 19 20 21
22 23 24 25 26 27 28
29 30 31 32 33 34 35 36
```

➢Program 3.2.31 :Write a program to display the following asking the user for the number of lines.

```
A
B C
D E F
G H I J
K L M N O
```

Step I :START.

Step II:PRINT "Enter the number of lines"

Step III:INPUT n

Step IV :i = 1, k = 1.

Step V :IF i > n, THEN GOTO step XIV .

Step VI :j = 1.

Step VII :IF j > i, THEN GOTO step XI

Step VIII :PRINT (k + 64) type casted to character type data

Step IX :j = j + 1, k = k + 1

Step X :GOTO step VII

Step XI :PRINT next line character.

Step XII :i = i + 1.

Step XIII :GOTO step V.

Step XIV :STOP.

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int i,j,n,k;
clrscr();
printf("Enter the number of lines:");
scanf("%d",&n);
for(i=1,k=1;i<=n;i++)
```

```
{
for(j=1;j<=i;j++)
{
printf("%c ",64+k++);
}
printf("\n");
}
getch();
}
```

Output

```
Enter the number of lines:6
A
B C
D E F
G H I J
K L M N O
P Q R S T U
```

➢Program 3.2.32 :Write a program to generate following patterns.

A

C B

F E D

J I H G

O N M L K *(May 2016)*

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int n,i,j,m=65,k=64;
printf("Enter n");
scanf("%d",&n);
for(i=1;i<=n;i++)
{
k=k+i;
m=k;
for(j=0;j<=n-i;j++)
printf(" ");
```

```
for(j=1;j<=i;j++)
printf("%c", m--);
printf("\n");
}
}
```

➢Program 3.2.33 :Write a program to generate following patterns.

```
1
2 1
1 2 3
4 3 2 1
1 2 3 4 5
```
(May 2016)

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int i,j;
for (i=1; i<=5;i++)
{
if(i%2 !=0)
{
for(j=1;j<=i;j++)
printf("%d ",j);
}
else
{
for(j=i;j>=1;j--)
printf("%d ",j);
}
}
```

Syllabus Topic : Control Structures - while and do-while Looping

Q.Explain difference between for, while and do while loop. *(May 2013, Dec. 2013, May 2015, Dec. 2015)*

Ans. :

Table 3.2.1 : Differences between while and do-while loop

Sr. No.	while loop	do-while loop	for loop

1.	Syntax of while loop:
	while(condition)
	{
	-
	-
	statements;
	-
	-
	-
	}
		Syntax of do-while loop:
		do
		{
		-
		-
		statements;
		-
		-
		-
		} while(condition);
			Syntax of for loop:
			for (initializations; condition; updations)
			{
			-
			-
			statements;
			-
			-
			-
			}

2.	This is called as a entry controlled loop, as the entry inside the loop is possible only if the condition is true.	This is called as exit controlled loop, as the entry inside this loop is sure i.e. no condition is checked to enter inside the loop. Nut the exit is possible only if the condition is false.	This is called as a entry controlled loop, as the entry inside the loop is possible only if the condition is true.
3.	If the condition is not true for the first time, the control will never enter into the loop. Hence there is a possibility that the control never enters into the loop and the statements inside the loop are never executed.	Even if the condition is not true for the first time the control will enter into the loop. Hence the statements inside the loop will be executed atleast once even if the condition is not true for the first time.	If the condition is not true for the first time, the control will never enter into the loop. Hence there is a possibility that the control never enters into the loop and the statements inside the loop are never executed.
4.	There is no semicolon (;) after the condition in the syntax of the while loop	There is a semicolon (;) after the condition in the syntax of the do-while loop.	There is no semicolon (;) after the condition in the syntax of the for loop
5.	The initializations and the updating is not a part of the syntax.	The initializations and the updating is not a part of the syntax.	The initializations and the updating is a part of the syntax.

➢Program 3.2.34 : Write a program to display first n natural numbers using the while loop.

➢Algorithm

Step I:START

Step II:PRINT "Enter a number".

Step III:INPUT n.

Step IV:i = 1.

Step V:IF i > n THEN GOTO step IX.

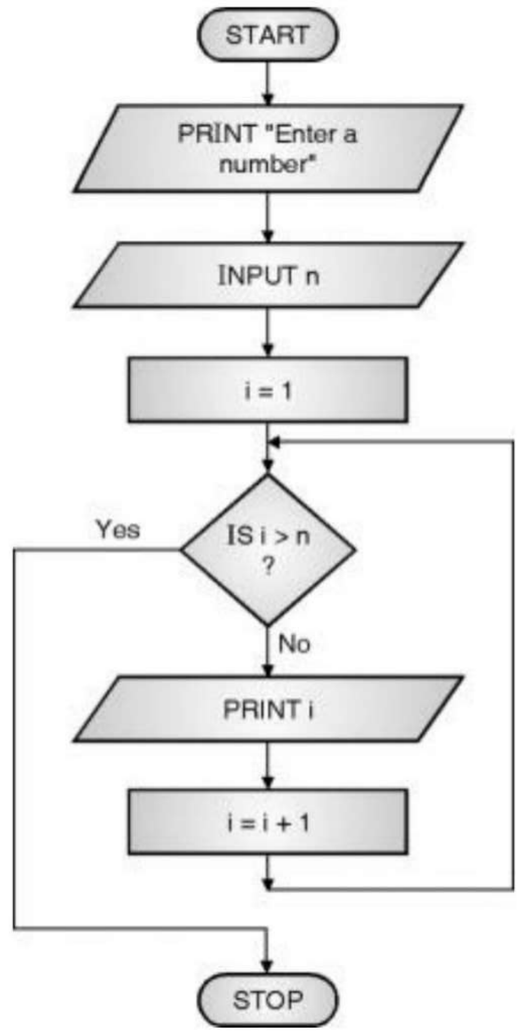
Step VI:PRINT i.

Step VII:i = i + 1.

Step VIII:GOTO step V

Step IX:STOP.

➢Flowchart : (Refer Flowchart 22)



Flowchart 22

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int i,n;
clrscr();
printf("Enter the value of n:");
scanf("%d",&n);
i=1;
while(i<=n)
```

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

Output

```
Enter the value of n:5
1
2
3
4
5
```

➢Program 3.2.35 :Write a program to display first n natural numbers using the do-while loop.

➢Algorithm

Step I:START

Step II:PRINT "Enter a number".

Step III:INPUT n.

Step IV:i = 1.

Step V:IF i > n THEN GOTO step IX.

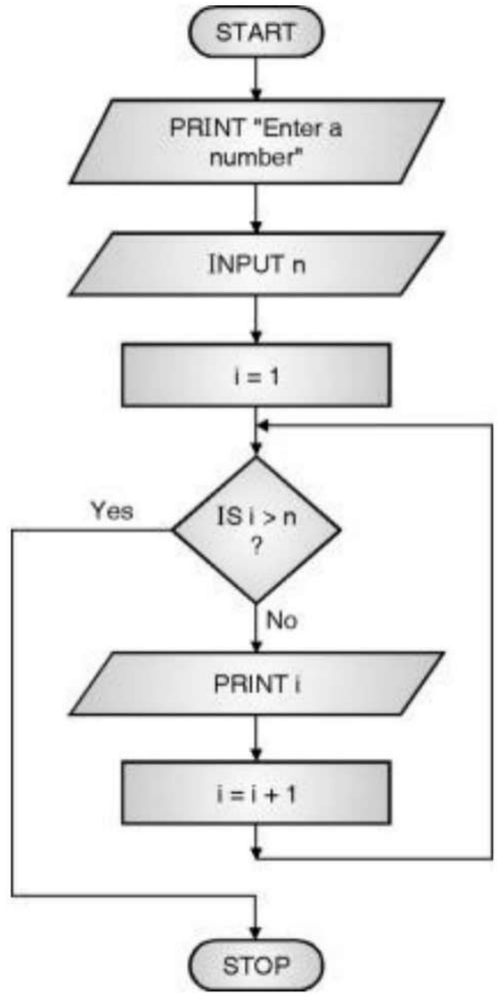
Step VI:PRINT i.

Step VII:i = i + 1.

Step VIII:GOTO step V

Step IX:STOP.

➢Flowchart : (Refer Flowchart 23)



Flowchart 23

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int i,n;
clrscr();
printf("Enter the value of n:");
scanf("%d",&n);
i=1;
do
{
printf("%d\n",i);
i++;
}while(i<=n);
getch();
}
```

Output

```
Enter the value of n:7
1
```

```
2
3
4
5
6
7
```

➢Program 3.2.36 :Write a program to find the sum and product of all the digits of a user entered number using the while loop.

➢Algorithm

Step I:START.

Step II:PRINT "Enter a number".

Step III :INPUT n.

Step IV :sum = 0, prod = 1

Step V:IF n = 0, THEN GOTO step X.

Step VI :digit = n mod 10.

Step VII :sum = sum + digit

prod = prod * digit

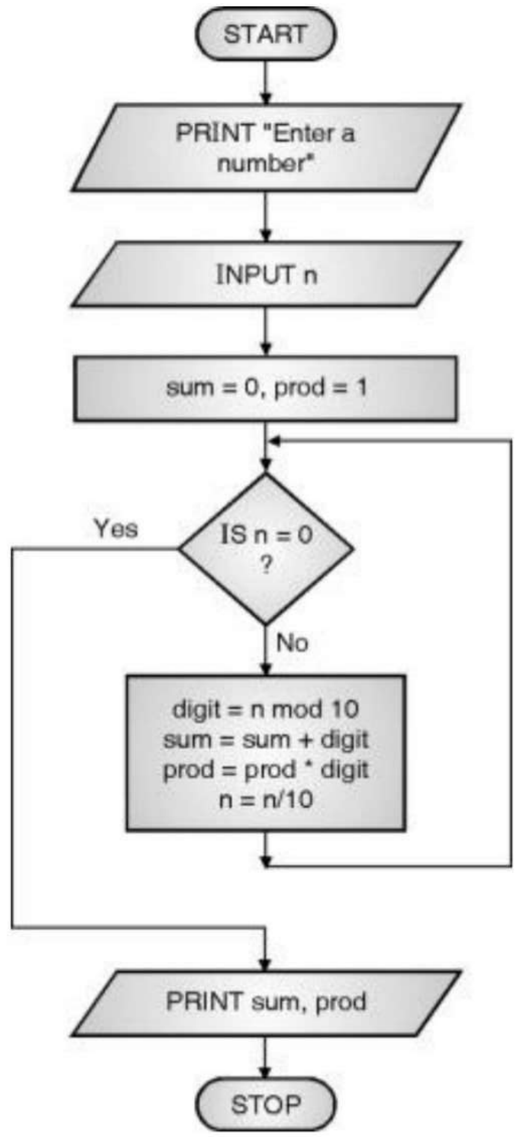
Step VIII :n = n / 10

Step IX :GOTO step V

Step X :PRINT sum, prod.

Step XI :STOP.

➢Flowchart : (Refer Flowchart 24)



Flowchart 24

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int n,digit,sum=0,product=1;
clrscr();
printf("Enter a number:");
scanf("%d",&n);
while(n!=0)
{
digit=n%10;
sum=sum+digit;
product=product*digit;
n=n/10;
}
printf("Sum=%d\nProduct=%d\n",sum,product);
getch();
}
```

Output

```
Enter a number:2354
Sum=14
Product=120
```

➢Program 3.2.37 :Write a program to find the sum and product of all the digits of a user entered number using the do-while loop.

➢Algorithm

Step I:START.

Step II:PRINT "Enter a number".

Step III :INPUT n.

Step IV :sum = 0, prod = 1

Step V:IF n = 0, THEN GOTO step X.

Step VI :digit = n mod 10.

Step VII :sum = sum + digit

prod = prod * digit

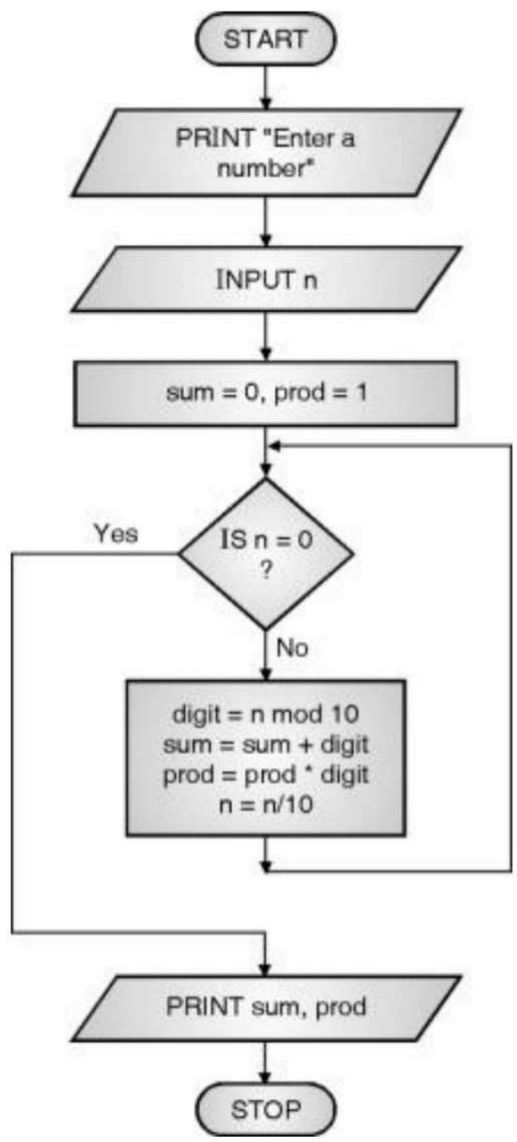
Step VIII :n = n / 10

Step IX :GOTO step V

Step X :PRINT sum, prod.

Step XI :STOP.

➢Flowchart : (Refer Flowchart 25)



Flowchart 25

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int n,digit,sum=0,product=1;
clrscr();
printf("Enter a number:");
scanf("%d",&n);
do
{
digit=n%10;
sum=sum+digit;
product=product*digit;
n=n/10;
}while(n!=0);
printf("Sum=%d\nProduct=%d\n",sum,product);
getch();
}
```

Output

```
Enter a number:1345
Sum=13
Product=60
```

➢Program 3.2.38 :Write a program to count the number of digits in a user entered number.

➢Algorithm

Step I:START.

Step II:PRINT "Enter a number".

Step III :INPUT n.

Step IV :count = 0

Step V:IF n = 0, THEN GOTO step IX.

Step VI :count = count + 1

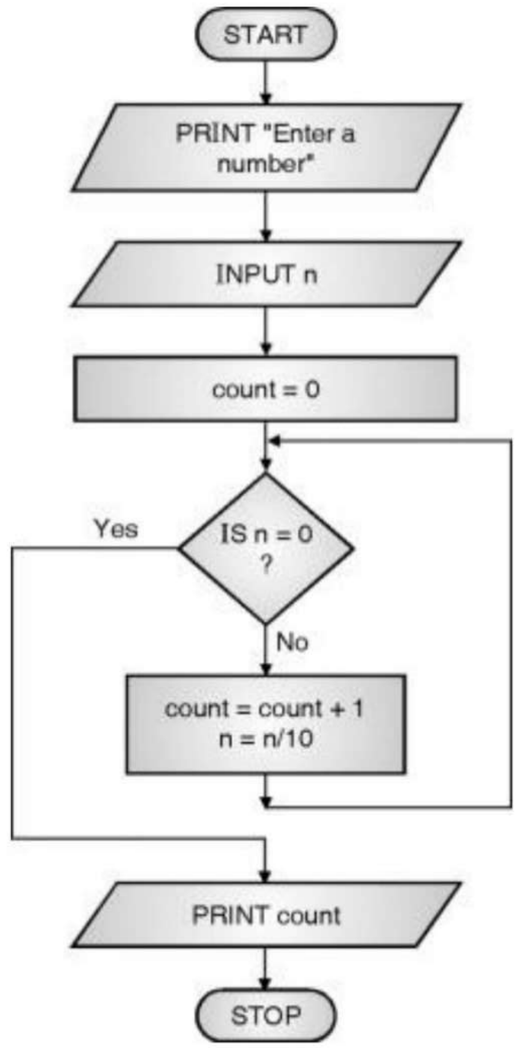
Step VII :n = n / 10

Step VIII :GOTO step V

Step IX :PRINT count.

Step X :STOP.

➢Flowchart : (Refer Flowchart 26)



Flowchart 26

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int n,count=0;
clrscr();
printf("Enter a number:");
scanf("%d",&n);
while(n!=0)
{
n=n/10;
count++;
}
printf("Number of digits=%d",count);
getch();
}
```

Output

```
Enter a number:7567
Number of digits=4
```

➢Program 3.2.39 :Write a program to find the number of digits before and after the decimal point in a floating point number.

➢Algorithm

Step I:START.

Step II:PRINT "Enter a number".

Step III :INPUT n.

Step IV:x = n type casted to integer type data

Step V :count = 0

Step VI:IF x = 0, THEN GOTO step X.

Step VII :count = count + 1

Step VIII :x = x / 10

Step IX :GOTO step V

Step X :count1 = 0

Step XI:x = n type casted to integer type data

Step XII:n = n – x

Step XIII:IF x = 0 THEN GOTO step

Step XIV:count1 = count1 + 1

Step XV:n = n * 10

Step XVI:x = n type casted to integer type data

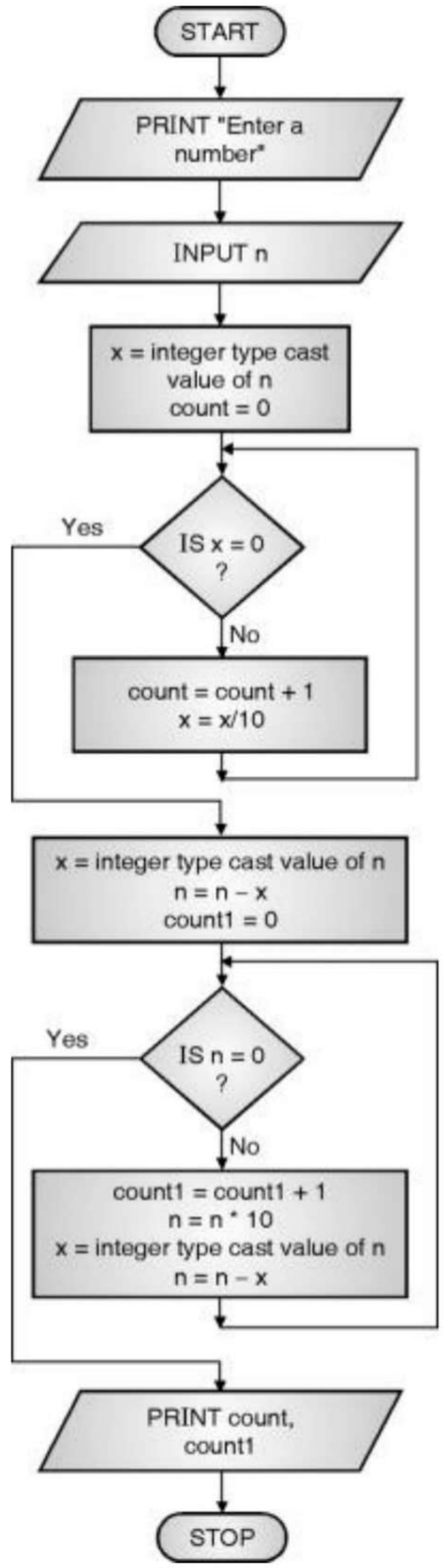
Step XVII:n = n – x

Step XVIII:GOTO step XIII

Step XIX:PRINT count, count1.

Step XX :STOP.

➢Flowchart : (Refer Flowchart 27)



Flowchart 27

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int n,countBefore=0,countAfter=0;
float x;
clrscr();
printf("Enter a number:");
scanf("%f",&x);
```

```
n=(int)(x);
x=x–n;
while(n!=0)
{
n=n/10;
countBefore++;
}
while(x!=0)
{
x=x*10;
n=(int)x;
x=x–n;
countAfter++;
}
printf("Number of digits before decimal
point=%d\n",countBefore);
printf("Number of digits before decimal point=%d",countAfter);
getch();
}
```

Output

```
Enter a number:1234.625
Number of digits before decimal point=4
Number of digits before decimal point=3
```

➢Program 3.2.40 :Write a program to reverse the digits of a user entered number and store in another variable.

➢Algorithm

Step I:START.

Step II:PRINT "Enter a number".

Step III :INPUT n.

Step IV :rev = 0

Step V:IF n = 0, THEN GOTO step X.

Step VI :digit = n mod 10.

Step VII :rev = rev * 10 + digit

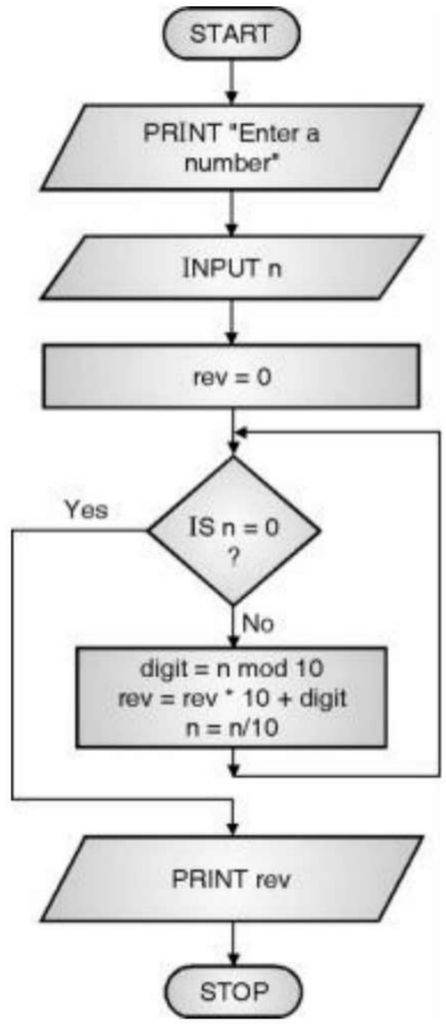
Step VIII :n = n / 10

Step IX :GOTO step V

Step X :PRINT rev.

Step XI :STOP.

➢Flowchart : (Refer Flowchart 28)



Flowchart 28

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int n,reverse=0,digit;
clrscr();
printf("Enter a number:");
scanf("%d",&n);
while(n!=0)
{
digit=n%10;
n=n/10;
reverse=reverse*10+digit;
}
printf("The reverse number is:%d",reverse);
getch();
}
```

Output

```
Enter a number:1234
```

```
The reverse number is:4321
```

➢Program 3.2.41 :Write a program to convert a decimal number to binary format. (May 2015)

➢Algorithm

Step I:START.

Step II:PRINT "Enter a number".

Step III :INPUT n.

Step IV :i = 15

Step V:IF i < 0, THEN GOTO step X.

Step VI :digit = n / 2^i.

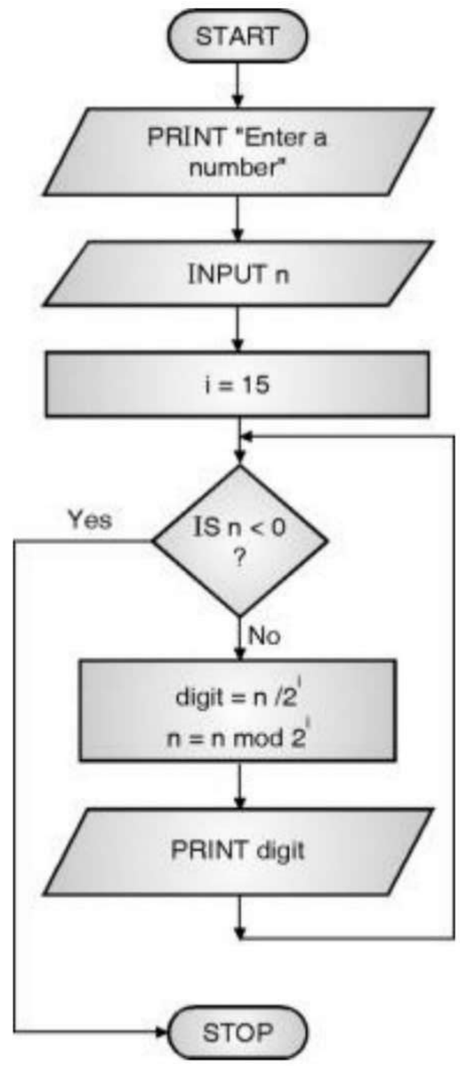
Step VII :PRINT n / 2^i

Step VIII :n = n mod 2^i

Step IX :GOTO step V

Step X :STOP.

➢Flowchart : (Refer Flowchart 29)



Flowchart 29

➢Program

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
void main()
```

```
{
int n,i;
clrscr();
    printf("Enter a number :");
scanf("%d",&n);
printf("Binary form is:");
for(i=15;i>=0;i– –)
{
printf("%d",n/(int)pow(2,i));
n=n%(int)(pow(2,i));
}
getch();
}
```

Output 1

```
Enter a number :12
Binary form is:0000000000001100
```

Output 2

```
Enter a number : 8
Binary form is:0000000000001000
```

➢Program 3.2.42 :Write a program to display the following pattern.

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

➢Algorithm

Step I :START.

Step II :i = 4.

Step III :IF i < 1, THEN GOTO step XVII.

Step IV:j = 1

Step V:IF j > 4 – i, THEN GOTO step IX

Step VI:PRINT " "

Step VII:j = j + 1

Step VIII:GOTO step V

Step IX :j = 1.

Step X:IF j > i, THEN GOTO step XIV

Step XI:PRINT "*"

Step XII :j = j + 1

Step XIII :GOTO step X

Step XIV :PRINT next line character.

Step XV:i = i – 1.

Step XVI:GOTO step III.

Step XVII:STOP.

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int i,j;
clrscr();
    for(i=4;i>=1;i– –)
{
for(j=1;j<=4–i;j++)
{
printf(" ");
}
for(j=1;j<=i;j++)
{
printf("*");
}
printf("\n");
}
getch();
}
```

Output

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

➢Program 3.2.43 :Write a C program to generate all combination of 1,2 & 3 using for loop.

(1 Row), 112 (2 Row), 113 (Row 3), 121 (Row 4), 122 (Row 5)……333(Last Row)

➢Algorithm

Step I :START.

Step II :i = 1.

Step III :IF i > 3, THEN GOTO step XV.

Step IV :j = 1.

Step V :IF j > 3, THEN GOTO step XIII.

Step VI :k = 1.

Step VII :IF k > 3, THEN GOTO step XI

Step VIII :PRINT i, j, k and next line character.

Step IX :k = k + 1.

Step X :GOTO step VII.

Step XI :j = j + 1

Step XII:GOTO step V.

Step XIII :i = i + 1.

Step XIV :GOTO step III

Step XV :STOP.

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int i,j,k;
clrscr();
    for(i=1;i<=3;i++)
{
for(j=1;j<=3;j++)
{
for(k=1;k<=3;k++)
{
printf("%d%d%d\n",i,j,k);
}
}
}
getch();
}
```

Output

```
111
112
113
121
```

```
122
123
131
132
133
211
212
213
221
222
223
231
232
233
311
312
313
321
322
323
331
332
333
```

➢Program 3.2.44 :Write a program to display the following pattern.

1

22

333

4444

55555

➢Algorithm

Step I :START.

Step II :i = 1.

Step III :IF i > 5, THEN GOTO step XVII.

Step IV:j = 1

Step V:IF j > 4 – i, THEN GOTO step IX

Step VI:PRINT " "

Step VII:j = j + 1

Step VIII:GOTO step V

Step IX :j = 1.

Step X :IF j > i, THEN GOTO step XIV

Step XI :PRINT i

Step XII :j = j + 1

Step XIII :GOTO step X

Step XIV :PRINT next line character.

Step XV :i = i + 1.

Step XVI :GOTO step III.

Step XVII :STOP.

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int i,j;
clrscr();
    for(i=1;i<=5;i++)
{
for(j=1;j<=5-i;j++)
{
printf(" ");
}
for(j=1;j<=i;j++)
{
printf("%d",i);
}
printf("\n");
}
getch();
}
```

Output

```
    1
   22
  333
 4444
55555
```

➢Program 3.2.45 :Write a program to display the following pattern.

P

PQ

PQR

PQRS

PQRST

➢Algorithm

Step I :START.

Step II :i = 1, x = 'P'.

Step III :IF i > n, THEN GOTO step XII .

Step IV :j = 1.

Step V :IF j > i, THEN GOTO step IX

Step VI :PRINT j + x – 1 type casted to character type data

Step VII :j = j + 1

Step VIII :GOTO step V

Step IX :PRINT next line character.

Step X :i = i + 1.

Step XI :GOTO step III.

Step XII :STOP.

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int i,j;
    char x='P';
clrscr();
for(i=1;i<=5;i++)
{
      for(j=1;j<=i;j++)
      {
printf("%c",(x+j–1));
      }
      printf("\n");
}
getch();
}
```

Output

```
P
```

```
PQ
PQR
PQRS
PQRST
```

➢Program 3.2.46 :Write a program to display the following pattern.

1

21A

321AB

4321ABC

54321ABCD

➢Algorithm

Step I :START.

Step II:PRINT "Enter the number of lines"

Step III:INPUT n

Step IV :i = 1.

Step V :IF i > n, THEN GOTO step XXIXV

Step VI:j = 1

Step VII:IF j > n – i, THEN GOTO step XI

Step VIII:PRINT " "

Step IX:j = j + 1

Step X:GOTO step VII

Step XI :j = i.

Step XII :IF j < 1, THEN GOTO step XVI

Step XIII :PRINT j

Step XIV :j = j – 1

Step XV :GOTO step XII

Step XVI:j = 1

Step XVII:IF j > i - 1, THEN GOTO step XXI

Step XVIII:PRINT j + 'A' – 1 type casted to character type data

Step XIX:j = j + 1

Step XX:GOTO step XVII

Step XXI :PRINT next line character.

Step XXII:i = i + 1.

Step XXIII:GOTO step V.

Step XXIV:STOP.

➢Program

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

```
#include<conio.h>
void main()
{
int i,j,n;
clrscr();
    printf("Enter the number of lines:");
scanf("%d",&n);
for(i=1;i<=n;i++)
{
for(j=1;j<=n–i;j++)
{
printf(" ");
}
for(j=i;j>=1;j– –)
{
printf("%d",j);
}
for(j=1;j<=i–1;j++)
{
printf("%c",(j+'A'–1));
}
printf("\n");
}
getch();
}
```

Output

```
Enter the number of lines:5
1
21A
321AB
4321ABC
54321ABCD
```

➢Program 3.2.47 :Write a program to calculate the value of

x=1! +3! + 5! +...(2*n–1)!.

➢Algorithm

Step I :START.

Step II :PRINT "Enter the value of n".

Step III :INPUT n.

Step IV :i = 1, sum = 0.

Step V :IF i > 2 * n – 1, THEN GOTO step XIII.

Step VI :j = 1, fact = 1

Step VII:IF j > i, THEN GOTO step X

Step VIII:fact = fact * j

Step IX:GOTO step VII

Step X:sum = sum + fact.

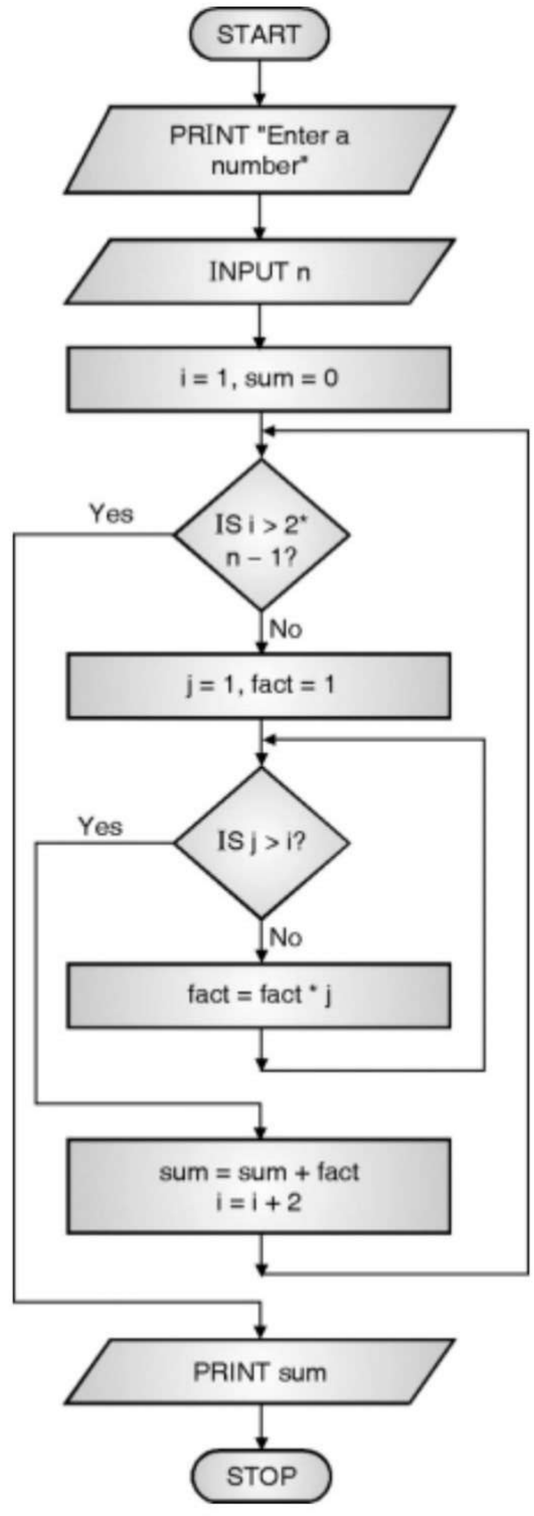
Step XI :i = i + 2

Step XII :GOTO step V.

Step XIII :PRINT sum.

Step XIV :STOP.

➢Flowchart : (Refer Flowchart 30)



Flowchart 30

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int i,j,sum=0,n,fact;
clrscr();
   printf("Enter a number:");
scanf("%d",&n);
for(i=1;i<=2 * n – 1;i+=2)
{
fact=1;
for(j=1;j<=i;j++)
{
fact=fact*j;
}
sum=sum+fact;
}
printf("The value of this series= %d",sum);
getch();
}
```

Output

```
Enter a number:4
The value of this series=5167
```

➢Program 3.2.48 :Write a program to display the following pattern.

A

AB

ABC

ABCD

ABCDE *(May 2013)*

➢Algorithm

Step I :START.

Step II :i = 1, x = 'A'.

Step III :IF i > n, THEN GOTO step XII .

Step IV :j = 1.

Step V :IF j > i, THEN GOTO step IX

Step VI :PRINT j + x – 1 type casted to character type data

Step VII :j = j + 1

Step VIII :GOTO step V

Step IX :PRINT next line character.

Step X :i = i + 1.

Step XI :GOTO step III.

Step XII :STOP.

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int i,j;
   char x='A';
clrscr();
for(i=1;i<=5;i++)
{
    for(j=1;j<=i;j++)
    {
printf("%c",(x+j-1));
    }
    printf("\n");
}
getch();
}
```

Output

```
A
AB
ABC
ABCD
ABCDE
```

➢Program 3.2.49 :Write a program to calculate sum of series

x=1/2 +3/4 + 5/6 +...n terms. ***(May 2013)***

➢Algorithm

Step I :START.

Step II :PRINT "Enter the value of n".

Step III :INPUT n.

Step IV :i = 1, j = 1, sum = 0.

Step V :IF i > n , THEN GOTO step X.

Step VI:sum = sum + j / (j + 1)

Step VII:j = j + 2

Step VIII:i = i + 1

Step IX:GOTO step VII

Step X :PRINT sum.

Step XI :STOP.

➢Program

```
#include<conio.h>
#include<stdio.h>
void main ()
{
   int i,j,n;
      float sum = 0;
      printf("Enter the value of n");
      scanf("%d",&n);
      for(i=1,j=1;i<=n; i++,j=j+2)
  {
         sum=sum+(float)(j)/(j+1);
      }
  printf("sum = %f",sum);
}
```

Output

```
Enter a number:3
The value of this series=2.083333
```

➢Program 3.2.50 : Write a program to generate following patterns.

5

44

333

2222

11111 ***(Dec. 2013)***

➢Program

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

```
void main()
{
int i,j,k;
clrscr();
    for(i=1,k=5;i<=5;i++,k– –)
{
for(j=1;j<=i;j++)
{
printf("%d",k);
}
printf("\n");
}
getch();
}
```

Output

```
5
44
333
2222
11111
```

➢Program 3.2.51 :Write a program to display the following pattern.

1

22

333

4444

55555 *(Dec. 2013)*

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int i,j;
clrscr();
    for(i=1;i<=5;i++)
{
for(j=1;j<=i;j++)
{
```

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

Output

```
1
22
333
4444
55555
```

➢Program 3.2.52 :Write a program to check whether the given number is palindrome or not. i.e. if no is 12421 it is palindrome. ***(May 2014)***

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int n,reverse=0,digit;
clrscr();
printf("Enter a number:");
scanf("%d",&n);
while(n!=0)
{
digit=n%10;
n=n/10;
reverse=reverse*10+digit;
}
if( n==rev) printf("Palindrome");
else printf("Not a Palindrome");
getch();
}
```

Output

```
Enter a number:1234
```

```
The reverse number is:4321
```

➢Program 3.2.53 :Write a program to calculate summation of series

1/2 – 3/4 + 5/6 – 7/8 + …….. upto n terms. ***(May 2014)***

➢Program

```
#include<conio.h>
#include<stdio.h>
void main ()
{
  int i,j,n;
     float sum = 0;
     printf("Enter the value of n");
     scanf("%d",&n);
     for(i=1,j=1;i<=n; i++,j=j+2)
  {
        sum=sum+(float)(j)/(j+1);
     }
  printf("sum = %f",sum);
}
```

Output

```
Enter a number:3
The value of this series=2.083333
```

➢Program 3.2.54 :Write a program to print the following pattern. (Note : Not only 4 lines, it should print n lines taken from user)

A

BB

CCC

DDD D

(Dec. 2014)

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int i,j,n;
clrscr();
```

```
printf("Enter the number of lines:");
scanf("%d",&n);
for(i=1;i<=n;i++)
{
for(j=1;j<=n-i;j++)
{
printf(" ");
}
for(j=1;j<=i;j++)
{
printf("%c ",(j+64));
}
printf("\n");
}
getch();
}
```

➢Program 3.2.55 :Write a Program to calculate summation of series.

$1 - x^2/2! + x^4/4! - x^6/6! + x^8/8! - \ldots$upto n terms***(Dec. 2015, May 2016)***

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
float x,sum,term,denr;
int i,n;
clrscr();
printf("enter the Value of x and (n) Number of terms to be sum\t :");scanf("%f%d",&x,&n);
sum =1; term = 1;
for (i=1;i<n;i++)
{
  denr = (2*i)*(2*i-1);
  term = -term*x*x/denr;
  sum =sum+ term;
}
printf("\nthe sum = %f\nNumber of terms = %d\nvalue of x = %f\n",sum,n,x);
getch();
}
```

➢Program 3.2.56 :Write a program to display following pattern. ***(May 2015)***

```
(i)  A B C D      (ii)        1
       A B C                1 2 A
         A B              1 2 3 A B
           A          1 2 3 4 A B C
```

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int i, j;
clrscr();
for (i=1; i<=4; i++)
{
for (j=1; j<=i–1; j++)
{
printf (“ ”);
}
for (j=1; j<=5–i; j++)
{
printf (“%c”,(char)(j+64));
}
printf (“\n”);
}
getch();
}
```

```
#include<stdio.h>
#include<conio.h>
void main()
{
int i, j;
clrscr();
for (i=1; i<=4; i++)
{
for (j=1; j<=4–i; j++)
{
```

```
printf (“ ”);
}
for (j=1; j<=i; j++)
{
printf (“%d”, j);
}
for (j = 1; j <=i–1; j++)
{
printf (“%c”, (char) (j + 64));
}
printf (“%n”);
}
getch();
}
```

➢Program 3.2.57 :Generate the following pattern of digits using nested loops.

1

2 3 2

3 4 5 4 3

4 5 6 7 6 5 4 ***(Dec. 2015)***

➢Program

```
#include <stdio.h>
int main()
{
  int i,space,rows,k=0,count=0,count1=0;
  printf("Enter the number of rows: ");
  scanf("%d",&rows);
  for(i=1;i<=rows;++i)
  {
     for(space=1;space<=rows-i;++space)
     {
       printf("  ");
       ++count;
     }
     while(k!=2*i-1)
     {
       if (count<=rows-1)
       {
         printf("%d ",(i+k));
         ++count;
       }
```

```
      else
      {
        ++count1;
         printf("%d ", (i+k-2*count1));
      }
      ++k;
    }
    count1=count=k=0;
    printf("\n");
  }
  return 0;
}
```

❑❑❑

CHAPTER 4

Control Structures : Branching and Nested Control Structure

Syllabus Topic : if statement and if-else Statement

➢Program 4.1.1 : Write a program to check if the user entered number is divisible by 10 on not.

➢Algorithm

Step I:START

Step II :PRINT "Enter a number".

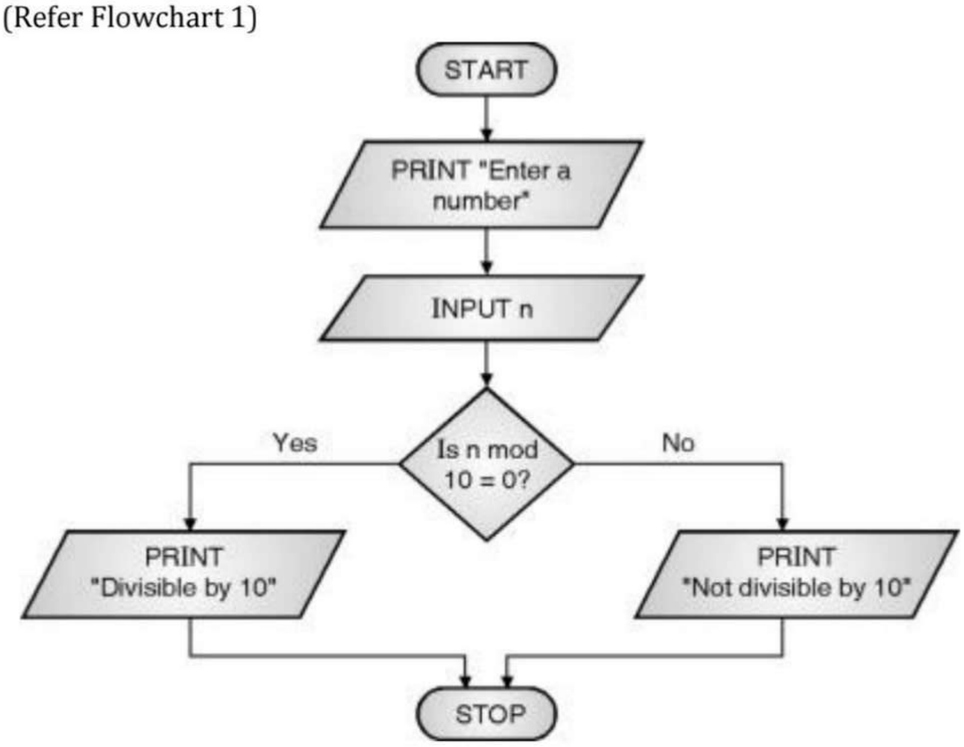
Step III:INPUT n.

Step IV:IF n mod 10 = 0 THEN PRINT "Divisible by 10"

ELSE PRINT "Not divisible by 10"

Step V:STOP.

Flowchart : (Refer Flowchart 1)



Flowchart 1

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int n;
clrscr();
```

```
   printf("Enter a number:");
scanf("%d",&n);
if(n%10==0)
{
printf("The number is divisible by 10");
}
else
{
printf("The number is not divisible by 10");
}
getch();
}
```

Output

```
Enter a number:400
The number is divisible by 10
```

➢Program 4.1.2 :Write a program to display first 10 numbers divisible by 5 and 8.

➢Algorithm

Step I:START.

Step II:i = 1, n = 1

Step III:IF i > 10, THEN GOTO step VII.

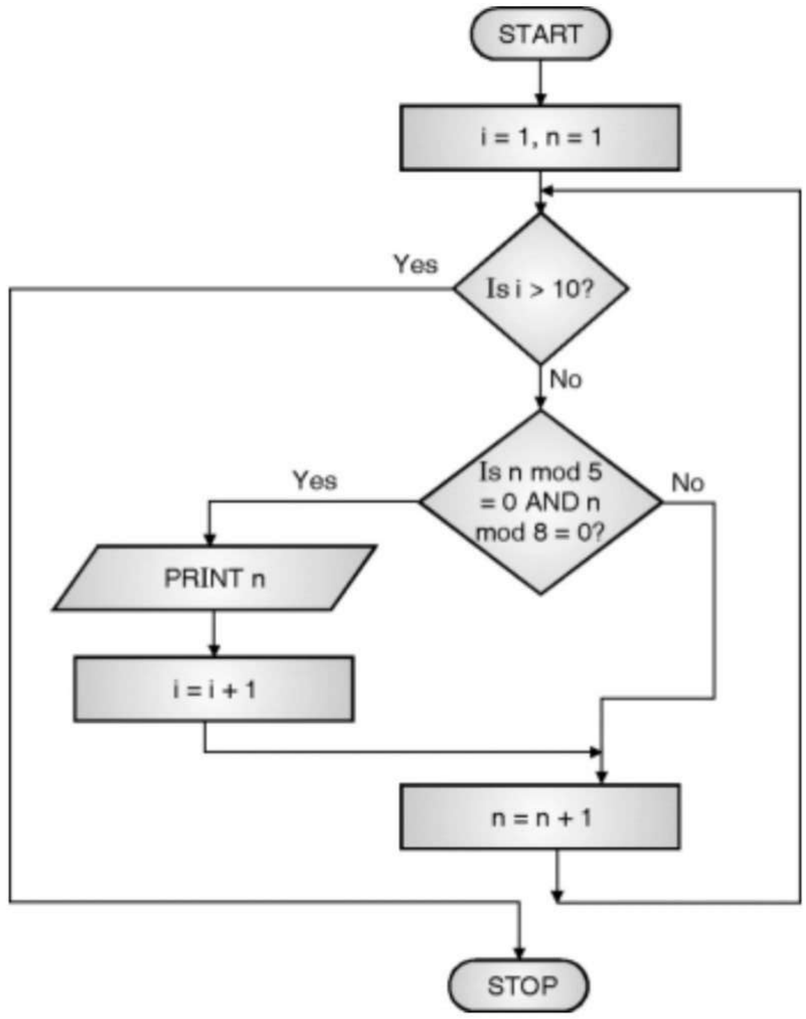
Step IV:IF n mod 5 = 0 AND n mod 10 = 0 THEN PRINT n AND i = i + 1

Step V:n = n + 1.

Step VI:GOTO step III

Step VII:Stop

➢Flowchart : (Refer Flowchart 2)



Flowchart 2

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int i=0,n=1;
clrscr();
    while(i<10)
{
if(n%5==0 && n%8==0)
{
printf("%d\n",n);
i++;
}
n++;
}
getch();
}
```

Output

```
40
```

```
80
120
160
200
240
280
320
360
400
```

➢Program 4.1.3 :Write a program to check if the entered number is prime number or not. ***(May 2015)***

➢Algorithm

Step I:START

Step II:PRINT "Enter a number".

Step III:INPUT n.

Step IV:x = 2.

Step V:IF (n mod x) = 0 THEN GOTO step VIII

Step VI:x = x + 1.

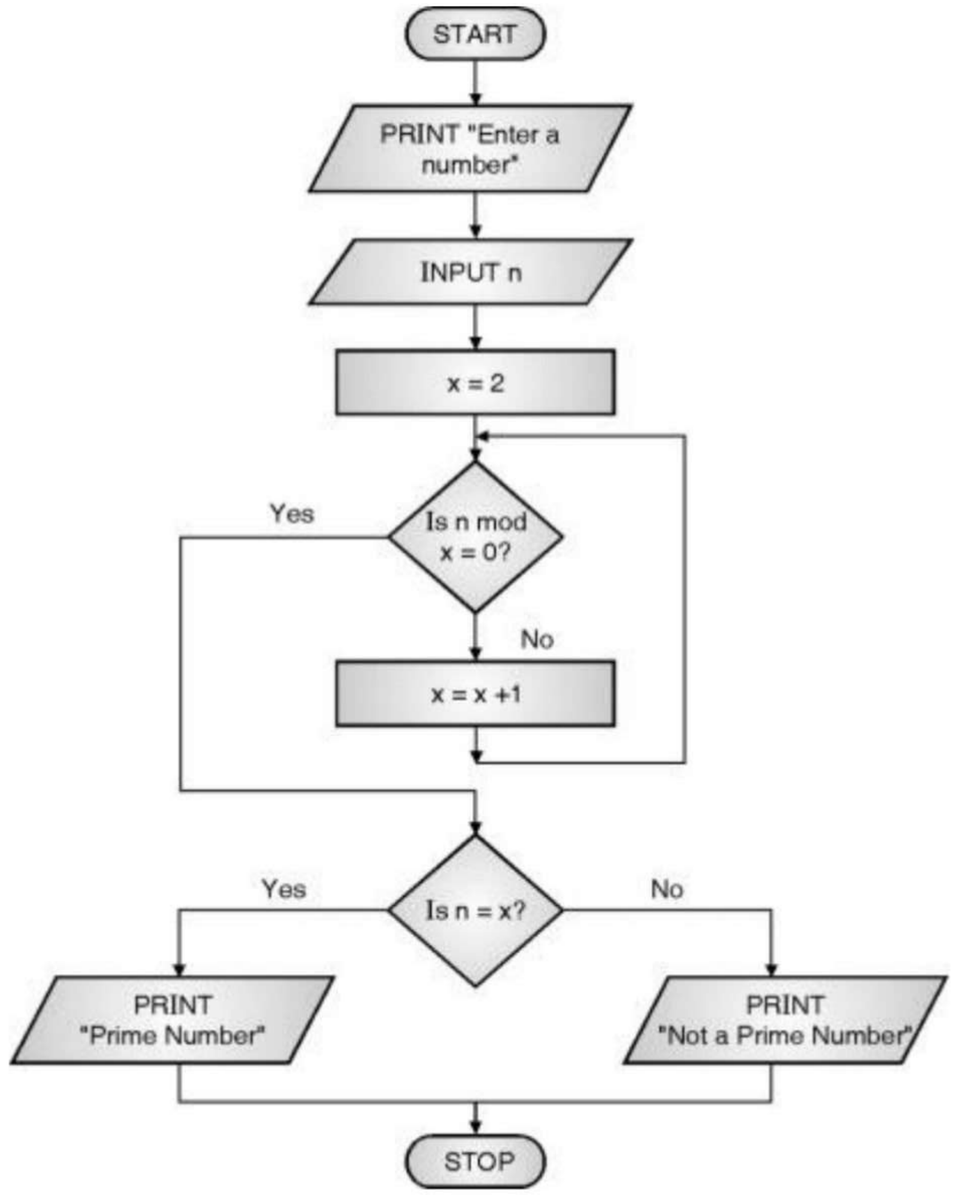
Step VII:GOTO step V

Step VIII:IF x = n THEN PRINT "Prime number"

ELSE PRINT "Not a prime number"

Step IX:STOP.

➢Flowchart : (Refer Flowchart 3)



Flowchart 3

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int i=2,n;
clrscr();
    printf("Enter a number:");
scanf("%d",&n);
while(n%i!=0)
{
i++;
}
if(n==i)
{
printf("Prime Number");
}
else
```

```
{
printf("Not a prime number");
}
getch();
}
```

Output

```
Enter a number:17
Prime Number
```

➢Program 4.1.4 :Write a program to display first n prime numbers where the value of n is taken from the user.

➢Algorithm

Step I:START

Step II:PRINT "Enter a number".

Step III:INPUT n.

Step IV:x = 1.

Step V :IF n = 0 THEN GOTO step XII.

Step VI:i = 2, x = x + 1.

tep VII:IF x mod i = 0, THEN GOTO step X.

tep VIII:i = i + 1.

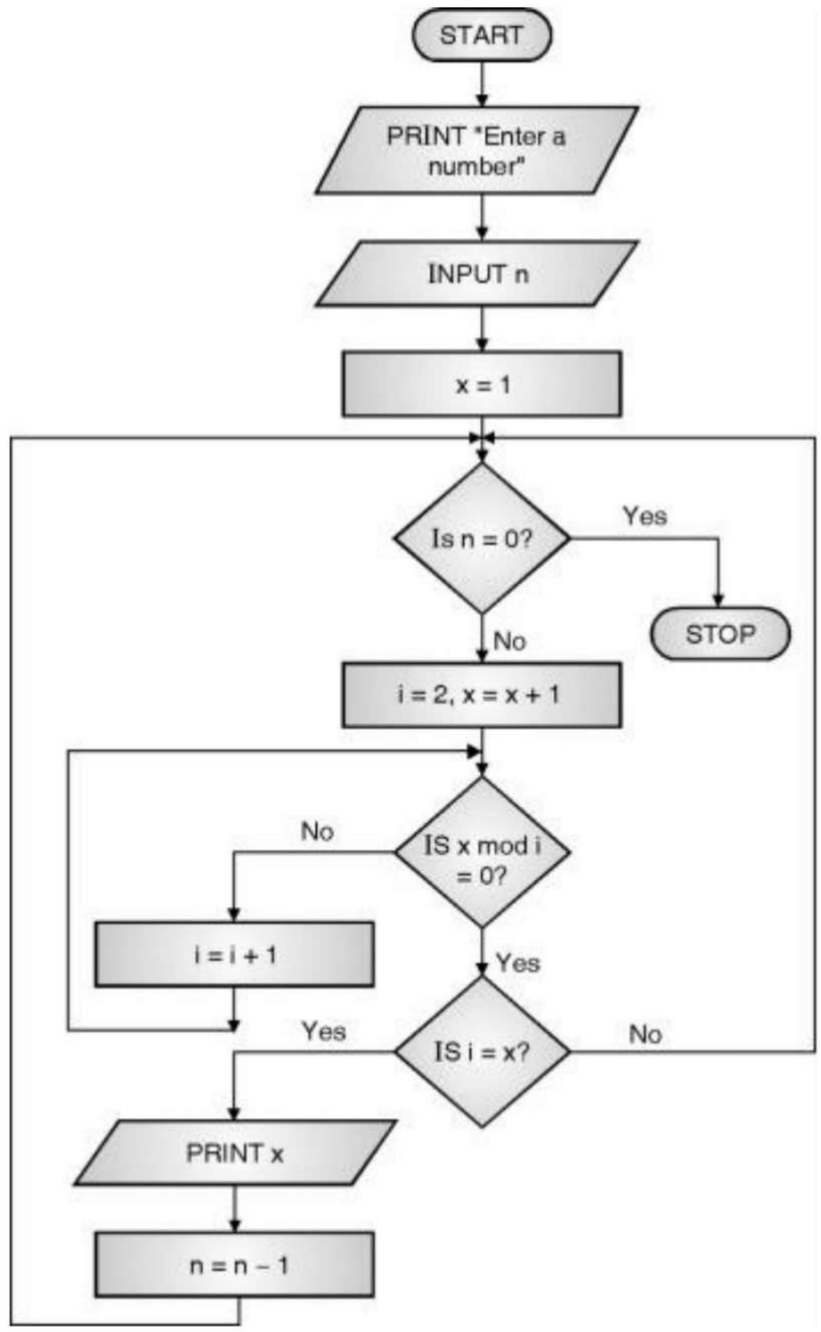
tep IX:GOTO step VII.

Step X:IF x = i, THEN PRINT x AND n = n – 1

Step XI:GOTO Step V.

Step XII:Stop.

➢Flowchart : (Refer Flowchart 4)



Flowchart 4

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int i,n,x=1;
clrscr();
    printf("Enter a number:");
scanf("%d",&n);
printf("The following are %d prime numbers\n",n);
while (n!=0)
{
x++;
i=2;
while(x%i!=0)
{
```

```
i++;
}
if(x==i)
{
printf("%d\n",x);
n--;
}
}
getch();
}
```

Output

```
Enter a number:10
The following are 10 prime numbers
2
3
5
7
11
13
17
19
23
29
```

➢Program 4.1.5 :Write a program to check if the entered number is Armstrong or not.***(Dec. 2015)***

Note : A number is said to be Armstrong number if the sum of the cube of its digits is equal to the number itself. For e.g. 153, the sum of the cubes is $1^3 + 5^3 + 3^3 = 1 + 125 + 27 = 153$ i.e. the original number itself.

➢Algorithm

Step I:START

Step II:PRINT "Enter a number".

Step III:INPUT n.

Step IV:copy = n, sum = 0.

Step V :IF n = 0, THEN GOTO step IX.

Step VI :digit = n mod 10

sum = sum + digit * digit * digit.

Step VII :n = n / 10.

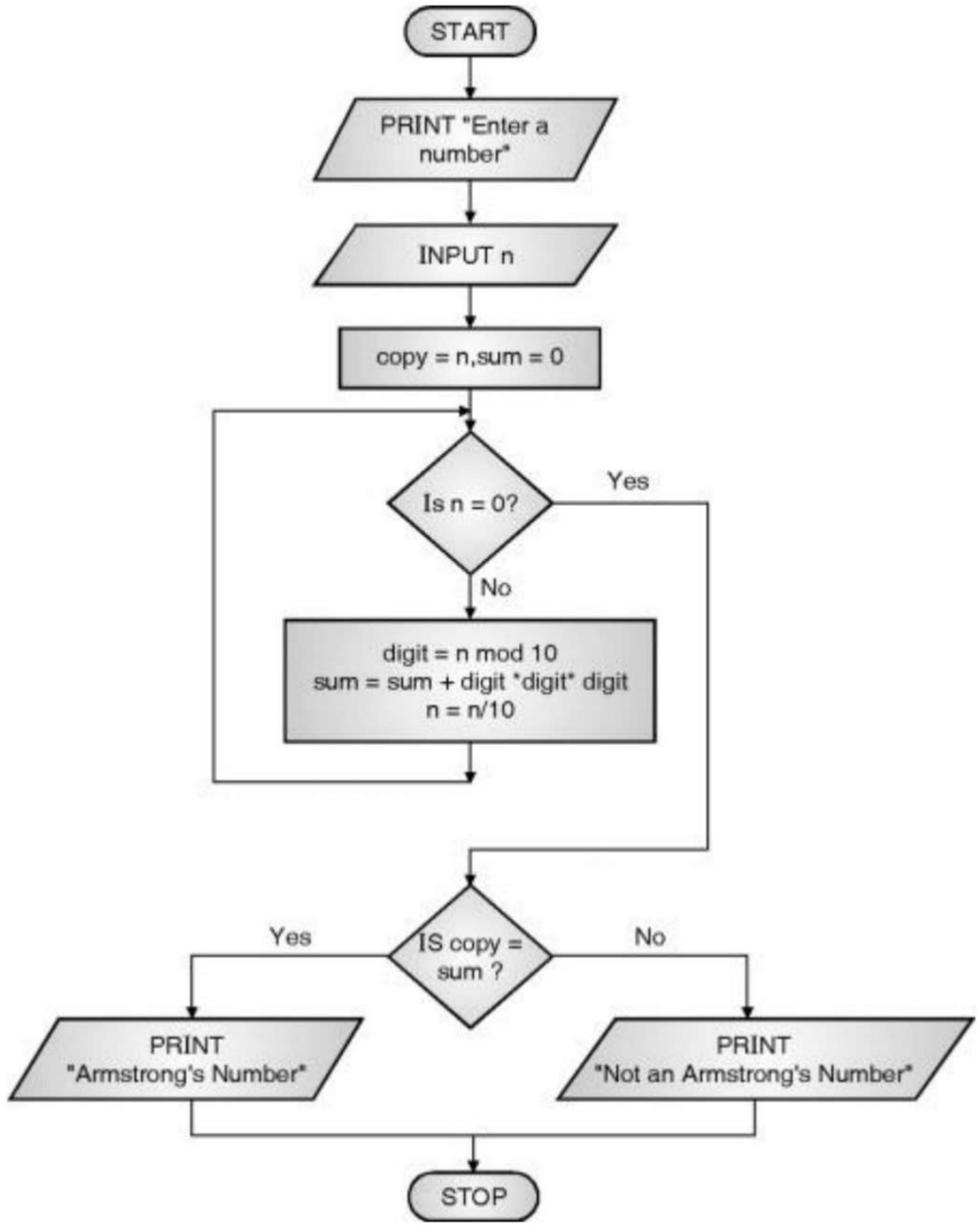
Step VIII :GOTO step V.

Step IX :IF copy = sum PRINT "Armstrong's number"

ELSE PRINT "Not Armstrong's number".

Step X :Stop.

➢Flowchart : (Refer Flowchart 5)



Flowchart 5

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int sum=0,digit,x,copy;
clrscr();
   printf("Enter a number:");
scanf("%d",&x);
copy=x;
while(x!=0)
{
digit=x%10;
```

```
sum=sum+digit*digit*digit;
x=x/10;
}
if(sum==copy)
{
printf("Armstrong Number");
}
else
{
printf("Not an Armstrong Number");
}
getch();
}
```

Output

```
Enter a number:789
Not an Armstrong Number
```

➢Program 4.1.6 :Write a program to display first n Armstrong numbers where the value of n is taken **from** the user.

➢Algorithm

Step I:START

Step II:PRINT "Enter a number".

Step III:INPUT n.

Step IV:x = 99.

Step V :IF n = 0, THEN GOTO step XIV.

Step VI :x = x + 1, copy = x, sum = 0.

tep VII :IF x = 0, THEN GOTO step XI.

tep VIII :digit = x mod 10, sum = sum + digit * digit * digit

tep IX :x = x / 10.

tep X :GOTO step VII.

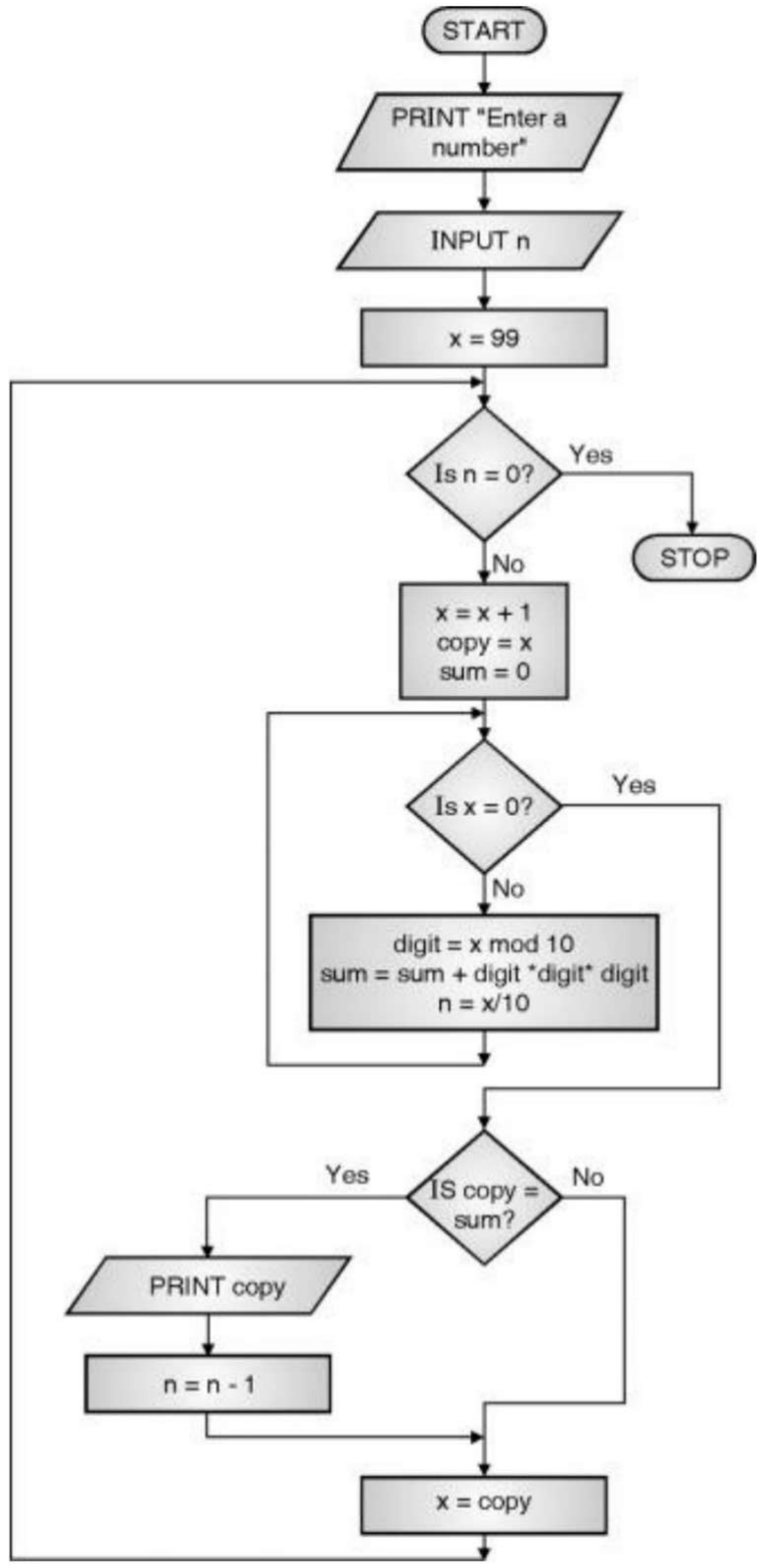
Step XI :IF sum = copy THEN PRINT copy AND n = n – 1.

Step XII :x = copy

Step XIII :GOTO Step V.

Step XIV :Stop.

➢Flowchart : (Refer Flowchart 6)



Flowchart 6

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int sum,digit,x=99,copy,n;
clrscr();
   printf("Enter a number:");
scanf("%d",&n);
printf("The list of %d Armstrong number is given below\n",n);
while(n!=0)
{
```

```
x++;
copy=x;
sum=0;
while(x!=0)
{
digit=x%10;
sum=sum+digit*digit*digit;
x=x/10;
}
if(sum==copy)
{
printf("%d\n",copy);
n--;
}
x=copy;
}
getch();
}
```

Output

```
Enter a number:4
The list of 4 Armstrong number is given below
153
370
371
407
```

➢Program 4.1.7 : Write a program to check if the year entered is leap year or not.

➢Algorithm

STEP I:START

Step II :PRINT "Enter a year number".

Step III:INPUT y.

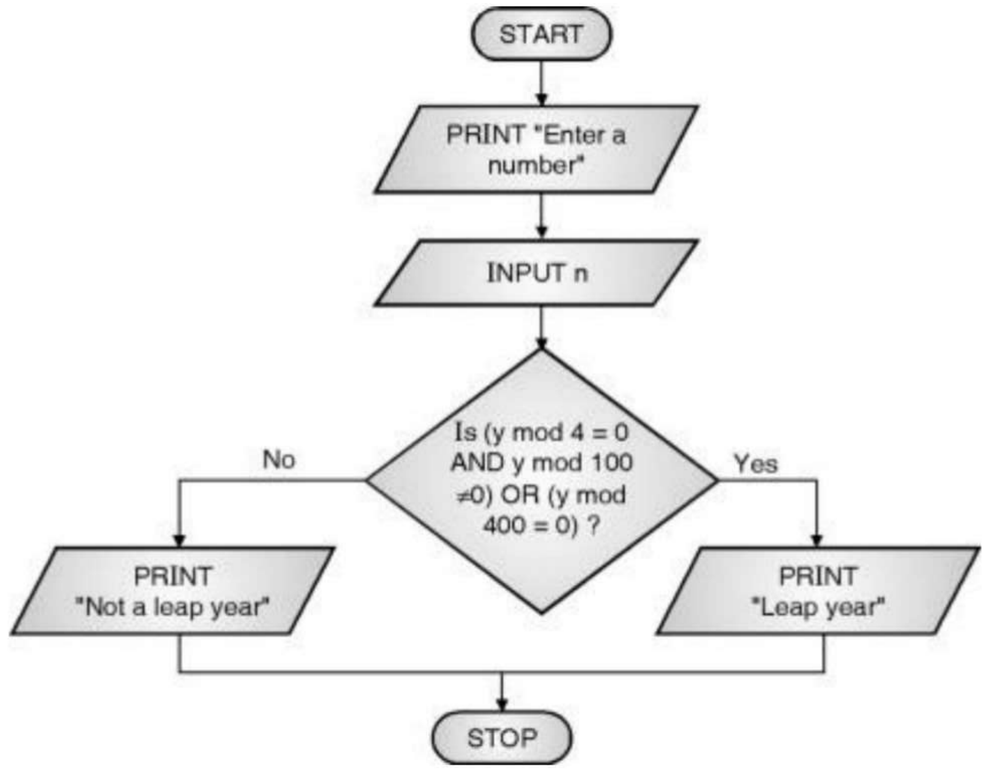
Step IV:IF (y mod 4 = 0 AND y mod 100 ≠ 0) OR (y mod 400 = 0) THEN

PRINT "It is a leap year"

ELSE PRINT "Not a leap year"

Step V:STOP.

➢Flowchart : (Refer Flowchart 7)



Flowchart 7

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int year;
clrscr();
    printf("Enter a year:");
scanf("%d",&year);
if(year%4==0 && year%100!=0 || year%100==0 && year%400==0)
printf("Leap Year");
else
printf("Not a leap year");
getch();
}
```

Output

```
Enter a year:1900
Not a leap year
```

➢Program 4.1.8 :Write a program to display the factors of a user entered number. ***(May 2015)***

➢Algorithm

Step I :START

Step II :PRINT "Enter a number".

Step III :INPUT n.

Step IV :i = 2.

Step V:IF n = i, THEN GOTO step IX.

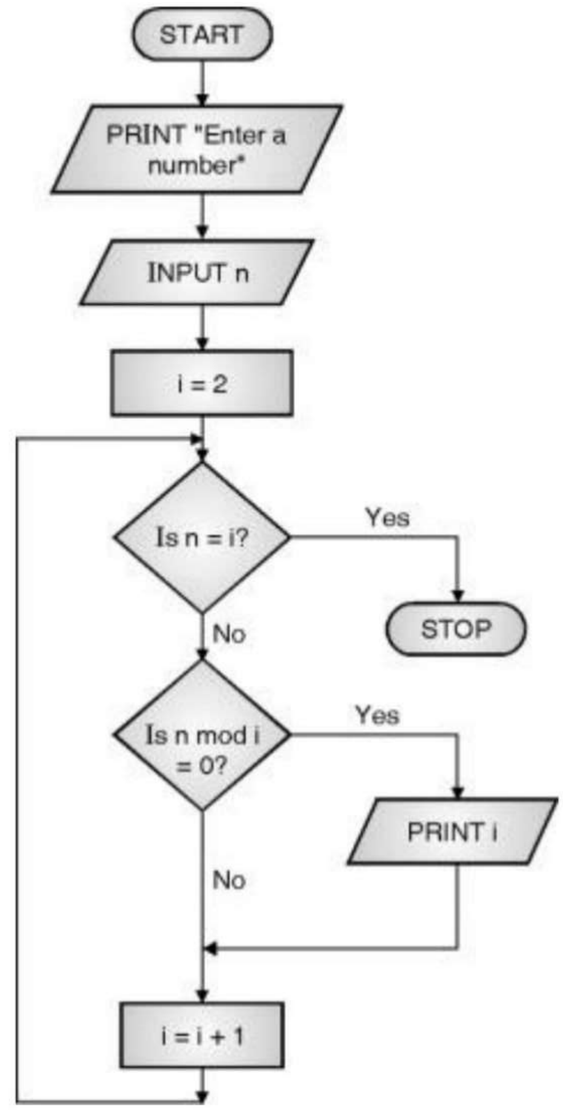
Step VI :IF n mod i = 0 THEN PRINT i.

Step VII :i = i + 1.

Step VIII :GOTO step V

Step IX :STOP.

➢Flowchart : (Refer Flowchart 8)



Flowchart 8

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int n,i;
clrscr();
   printf("Enter a number:");
scanf("%d",&n);
printf("Factors are:\n");
i=2;
while(i<n)
```

```
{
if(n%i==0)
printf("%d\n",i);
i++;
}
getch();
}
```

Output

```
Enter a number:30
Factors are:
2
3
5
6
10
15
```

Syllabus Topic : Multiway Decision

Q.Explain the need and the syntax of if-else ladder.

Ans. :

- In some cases we have to check multiple cases of a particular condition. In such cases we have to use an if-else ladder.
- A set of if-else statements as shown below is called as if-else ladder.

Syntax

```
if(condition)
{
statements;
}
else
{
if (condition)
{
statements;
}
else
{
if (condition)
```

```
{
statements;
}
else
{
if(condition)
{
statements;
}
|
|
|
else
{
statements;
}
}
}
}
}
```

- In this case the first condition is checked, if it is true the statements inside the if statement are executed. But if the condition is false it goes to the else statement. Again there is a condition with if; the statements are executed if this second condition is true. Else it again goes to the else and again checks the condition associated with this if statement. Thus if one condition satisfies no other else is checked thereafter.

➢Program 4.1.9 :Write a program to display the class according to the marks scored by a student. The marks scored is taken as input and the class is displayed according to the following range :

Marks	Class
70-100	Distinction
60-69	First Class
50-59	Second Class
40-49	Pass Class
0-39	Fail

- Algorithm

Step I :START.

Step II :PRINT "Enter marks out of 100".

Step III :INPUT marks

Step IV :IF marks > = 70, PRINT "Distinction" AND GOTO step IX

Step V :IF marks > = 60, PRINT "First Class" AND GOTO step IX

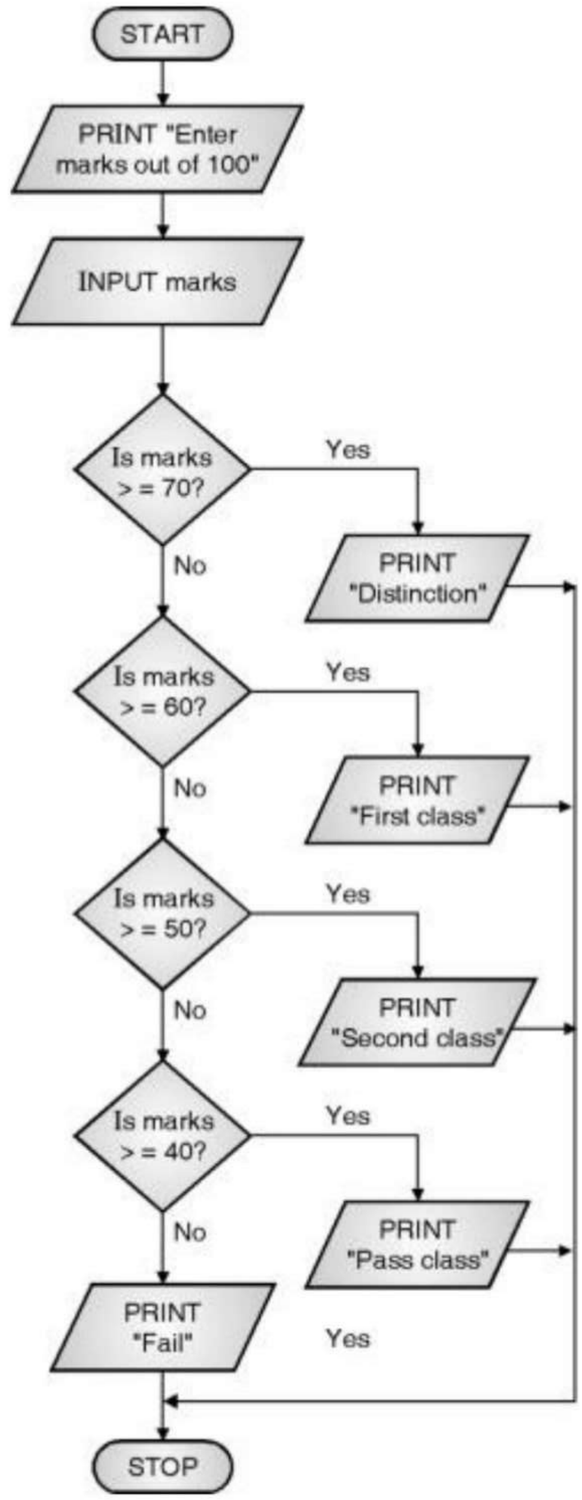
Step VI :IF marks > = 50, PRINT "Second Class" AND GOTO step IX

Step VII :IF marks > = 40, PRINT "Pass Class" AND GOTO step IX

Step VIII :PRINT "Fail"

Step IX :Stop.

➢Flowchart : (Refer Flowchart 9)



Flowchart 9

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int marks;
clrscr();
```

```
    printf("Enter marks scored (0-100):");
scanf("%d",&marks);
if(marks>=70)
{
printf("Distinction");
}
else
{
if (marks>=60)
{
printf("First Class");
}
else
{
if (marks>=50)
{
printf("Second Class");
}
else
{
if(marks>=40)
{
printf("Pass Class");
}
else
{
printf("Fail");
}
}
}
}
getch();
}
```

Output

```
Enter marks scored (0-100):89
Distinction
```

Syllabus Topic : Switch Statement

Q.Explain switch control statement with the help of example. ***(May 2014)***

OR Explain syntax of switch case? Discuss what is need of break statement in switch case? ***(May 2015)***

Ans. :

The syntax of switch-case is given below :

```
switch(expression / variable)
{
case label1: statements;
break;
case label2:statements;
break;
case label3:statements;
break;
|
|
case labeln:statements;
break;
default : statements;
}
```

Note : Break statement transfers the control outside the current loop. We will see some more cases of break statement along with the for / while /do-while statements in section 4.1.3.

- The expression or variable can be given in the brackets associated with the switch statement. The values of this expression / variable are the labels associated with the cases inside the switch statement.
- The value of the expression / variable is first compared with the label1. If they are equal, then the statements followed by the corresponding case are executed. The break statement followed by these statements, transfers the control after the switch statement.
- If the expression / variable is not equal to label1, then it is directly compared to label2 without executing the statements followed by the label1.
- Hence the statements of only that case are executed with the correct label value of the expression / variable.

Note :

1. Break statement is not compulsory. But if the break statement is nut written everything will be executed after the case statements
2. Default is not necessary. But in case if default is not written and some case occurs which is not considered then there will be no operation performed and the user will not be able to realize the problem in the program.
3. The label can only be value, it cannot be a condition or an expression.
4. The brackets associated with the switch statement can have either the variable or expression and no condition

➢Program 4.1.10 :Write a program to display the user entered single digit number in words. ***(May 2014)***

➢Algorithm

Step I :START

Step II :PRINT "Enter a single digit number".

Step III:INPUT n.

Step IV :IF n = 0, THEN PRINT "Zero" AND GOTO step XIV

Step V :IF n = 1, THEN PRINT "One" AND GOTO step XIV

Step VI :IF n = 2, THEN PRINT "Two" AND GOTO step XIV

Step VII :IF n = 3, THEN PRINT "Three" AND GOTO step XIV

Step VIII :IF n = 4, THEN PRINT "Four" AND GOTO step XIV

Step IX :IF n = 5, THEN PRINT "Five" AND GOTO step XIV

Step X :IF n = 6, THEN PRINT "Six" AND GOTO step XIV

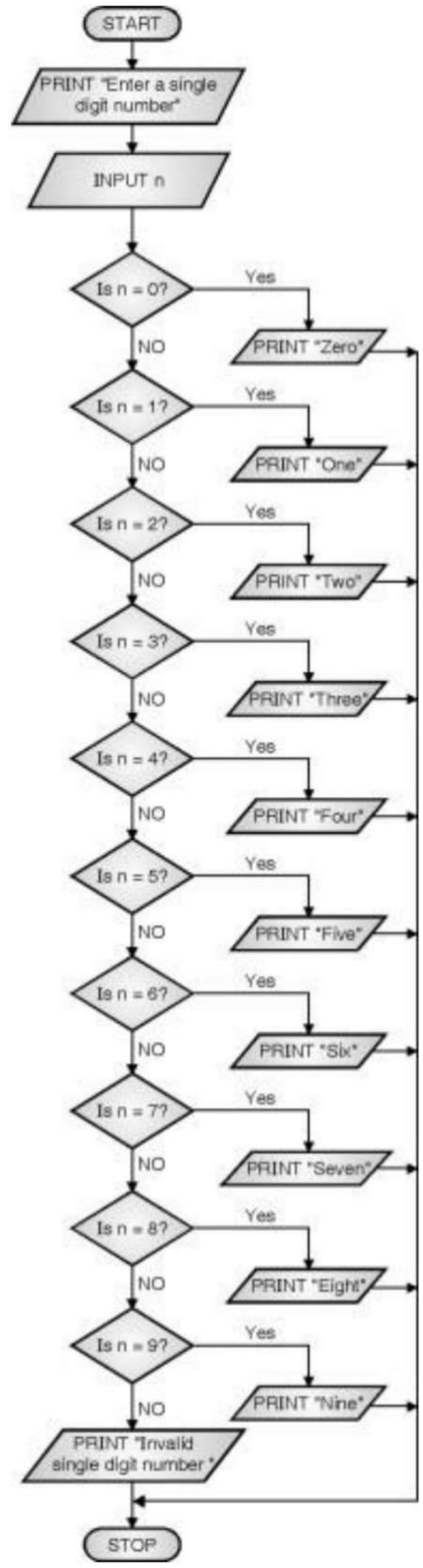
Step XI :IF n = 7, THEN PRINT "Seven" AND GOTO step XIV

Step XII :IF n = 8, THEN PRINT "Eight" AND GOTO step XIV

Step XIII :IF n = 9, THEN PRINT "Nine" AND GOTO step XIV

Step XIV:Stop.

- Flowchart : (Refer Flowchart 10)



Flowchart 10

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int n;
clrscr();
   printf("Enter a single digit number:");
scanf("%d",&n);
switch(n)
```

```
{
case 0:printf("Zero");
break;
case 1:printf("One");
break;
case 2:printf("Two");
break;
case 3:printf("Three");
break;
case 4:printf("Four");
break;
case 5:printf("Five");
break;
case 6:printf("Six");
break;
case 7:printf("Seven");
break;
case 8:printf("Eight");
break;
case 9:printf("Nine");
break;
}
getch();
}
```

Output

```
Enter a single digit number:4
Four
```

➢Program 4.1.11 :Write a program to display a user entered number in words.

➢Algorithm

Step I :START

Step II :PRINT "Enter a number".

Step III :INPUT n.

Step IV :rev = 0.

Step V :IF n = 0, THEN GOTO step X.

Step VI :digit = n mod 10.

Step VII :n = n / 10.

Step VIII:rev = rev * 10 + digit.

Step IX :GOTO step V

Step X :IF rev = 0, THEN GOTO step XXV.

Step XI :digit = rev mod 10.

Step XII :IF digit = 0, THEN PRINT "Zero" AND GOTO step XXII

Step XIII :IF digit = 1, THEN PRINT "One" AND GOTO step XXII

Step XIV :IF digit = 2, THEN PRINT "Two" AND GOTO step XIXI

Step XV :IF digit = 3, THEN PRINT "Three" AND GOTO step XXII

Step XVI :IF digit = 4, THEN PRINT "Four" AND GOTO step XXII

Step XVII :IF digit = 5, THEN PRINT "Five" AND GOTO step XXII

Step XVIII:IF digit = 6, THEN PRINT "Six" AND GOTO step XXII

Step XIX :IF digit = 7, THEN PRINT "Seven" AND GOTO step XXII

Step XX :IF digit = 8, THEN PRINT "Eight" AND GOTO step XXII

Step XXI :IF digit = 9, THEN PRINT "Nine" AND GOTO step XXII

Step XXII:rev = rev / 10.

Step XXIII:GOTO step X

Step XXIV:STOP.

- The flowchart is similar to the previous program and the program for reversing the given number as studied earlier

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int n,digit,rev=0;
clrscr();
    printf("Enter a number:");
scanf("%d",&n);
while(n!=0)
{
digit=n%10;
rev=rev*10+digit;
n=n/10;
}
while(rev!=0)
{
digit=rev%10;
rev=rev/10;
switch(digit)
```

```
{
case 0:printf("Zero ");
break;
case 1:printf("One ");
break;
case 2:printf("Two ");
break;
case 3:printf("Three ");
break;
case 4:printf("Four ");
break;
case 5:printf("Five ");
break;
case 6:printf("Six ");
break;
case 7:printf("Seven ");
break;
case 8:printf("Eight ");
break;
case 9:printf("Nine ");
break;
}
}
getch();
}
```

Output

```
Enter a number:513
Five One Three
```

➢Program 4.1.12 :Write a menu driven program to perform add / subtract / multiply / divide / modulus based on the users choice.***(Dec. 2014)***

➢Algorithm

Step I :START.

Step II :PRINT "Enter two numbers".

Step III :INPUT a, b.

Step IV :PRINT "1. Addition, 2. Subtraction, 3. Multiplication, 4. Division 5. Modulus. Enter your choice".

Step V :INPUT choice.

Step VI :IF choice = 1 THEN

sum = a + b

PRINT sum AND

GOTO step XII

Step VII :IF choice = 2 THEN

difference = a - b

PRINT difference AND

GOTO step XII

Step VIII :IF choice = 3 THEN

prod = a * b

PRINT prod AND

GOTO step XII

Step IX :IF choice = 4 THEN

quotient = a / b

PRINT quotient AND

GOTO step XII

Step X :IF choice = 5 THEN

remainder = a mod b

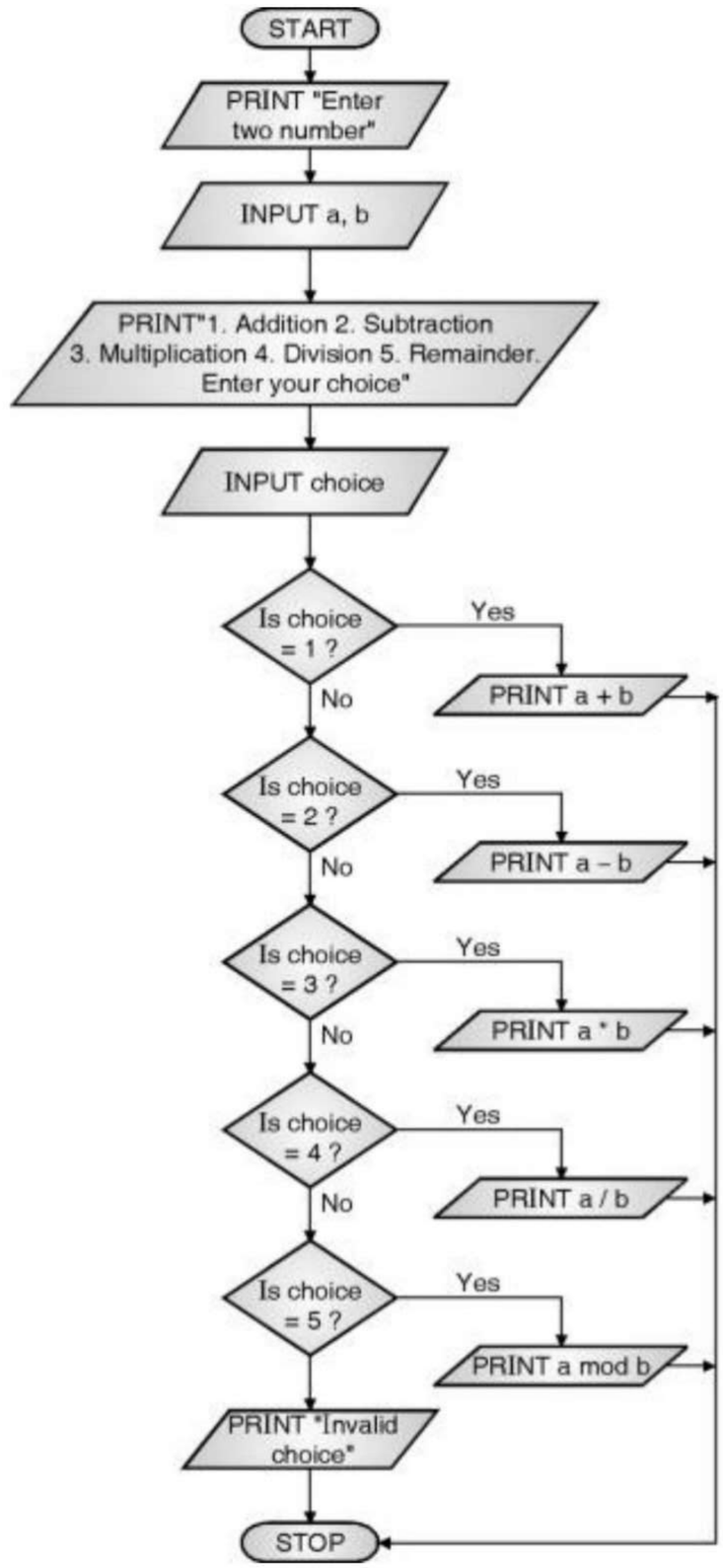
PRINT remainder AND

GOTO step XII

Step XI :PRINT "Invalid choice".

Step XII :STOP.

- Flowchart : (Refer Flowchart 11)



Flowchart 11

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int no1,no2,result,choice;
clrscr();
    printf("Enter two numbers:");
scanf("%d %d",&no1,&no2);
printf("1.Add\n2.Subtract\n3.Multiply\n4.Divide\n5.Modulus\nEnter your choice:");
scanf("%d",&choice);
switch(choice)
```

```
{
case 1:result=no1+no2;
printf("Sum= %d",result);
break;
case 2:result=no1–no2;
printf("Difference= %d",result);
break;
case 3:result=no1*no2;
printf("Product= %d",result);
break;
case 4:result=no1/no2;
printf("Quotient= %d",result);
break;
case 5:result=no1%no2;
printf("Remainder= %d",result);
break;
default:printf("Invalid choice");
}
getch();
}
```

Output

```
Enter two numbers:12
6
1.Add
2.Subtract
3.Multiply
4.Divide
5.Modulos
Enter your choice:3
Product=72
```

➢Program 4.1.13 :Write a menu driven program to perform add / subtract / multiply / divide based on the users choice. The user will indicate the operation to be performed using the signs i.e. + for addition, - for subtraction and so on.

➢Algorithm

Step I :START.

Step II :PRINT "Enter two numbers".

Step III :INPUT a, b.

Step IV :PRINT "+. Addition, –. Subtraction, *. Multiplication, /. Division %. Modulus. Enter your choice".

Step V :INPUT choice.

Step VI :IF choice = + THEN

sum = a + b

PRINT sum AND

GOTO step XII

Step VII :IF choice = – THEN

difference = a -– b

PRINT difference AND

GOTO step XII

Step VIII :IF choice = * THEN

prod = a * b

PRINT prod AND

GOTO step XII

Step IX :IF choice = / THEN

quotient = a / b

PRINT quotient AND

GOTO step XII

Step X :IF choice = % THEN

remainder = a mod b

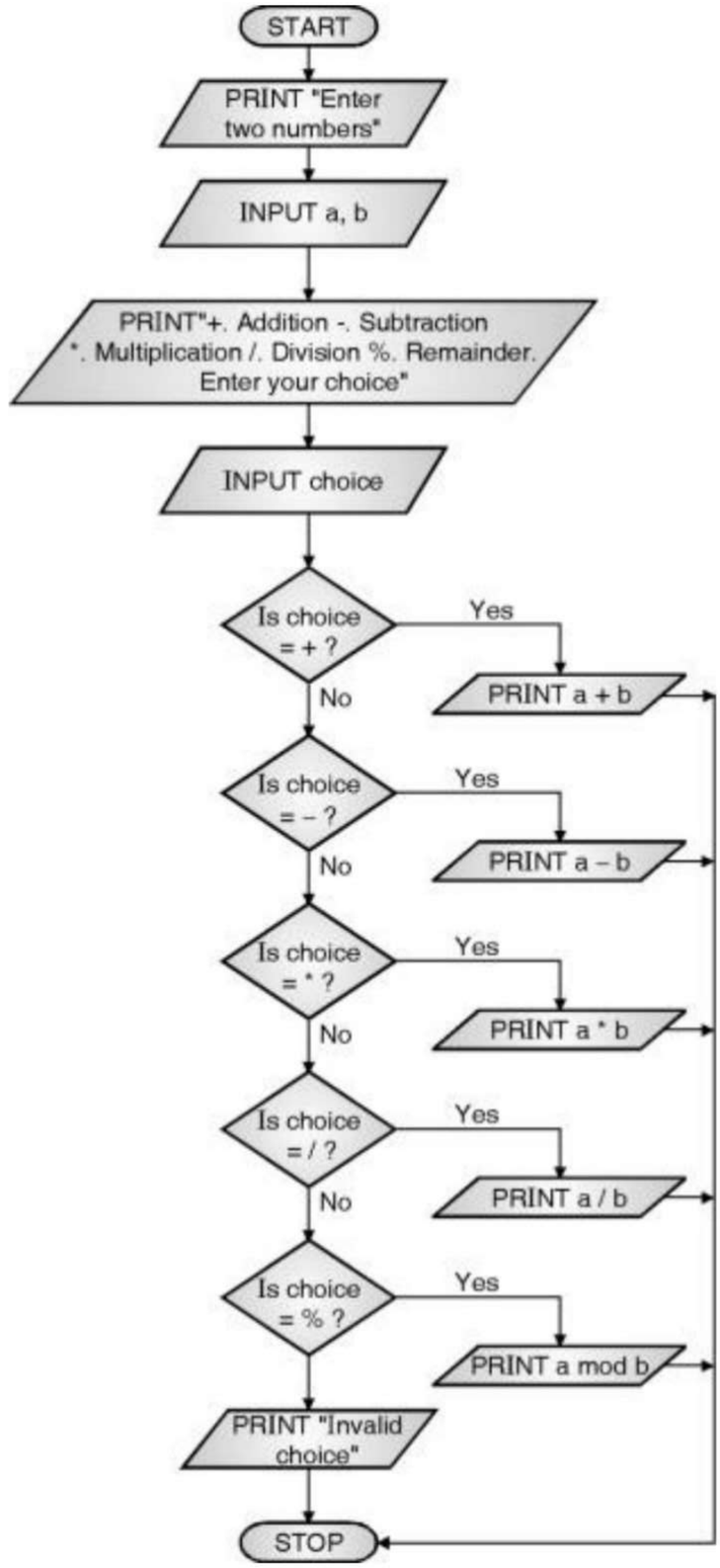
PRINT remainder AND

GOTO step XII

Step XI :PRINT "Invalid choice".

Step XII :STOP.

- Flowchart : (Refer Flowchart 12)



Flowchart 12

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int no1,no2,result;
char choice;
clrscr();
printf("Enter the numbers:");
scanf("%d %d",&no1,&no2);
printf("+.Add\n-.Subtract\n*.Multiply\n/.Divide\n%.Modulus\nEnter your choice:");
choice=getche();
```

```
printf("\n");
switch(choice)
{
case '+':result=no1+no2;
printf("Sum= %d",result);
break;
case '-':result=no1-no2;
printf("Difference= %d",result);
break;
case '*':result=no1*no2;
printf("Product= %d",result);
break;
case '/':result=no1/no2;
printf("Quotient= %d",result);
break;
case '%':result=no1%no2;
printf("Remainder= %d",result);
break;
default:printf("Invalid choice");
}
getch();
}
```

Output

```
Enter the numbers:4
5
+.Add
-.Subtract
*.Multiply
/.Divide
%.Modulus
Enter your choice:*
Product=20
```

➢Program 4.1.14 :Write a program to enter the operations in natural form for doing the basic operations like add / subtract / multiply / divide based on the users choice. For e.g. the user enters 5 + 12, and the sum is displayed.

➢Algorithm

Step I :START.

Step II :PRINT "Enter the operation".

Step III :INPUT a, choice, b.

Step IV :IF choice = + THEN

sum = a + b

PRINT sum AND

GOTO step X

Step V :IF choice = – THEN

difference = a – b

PRINT difference AND

GOTO step X

Step VI :IF choice = * THEN

prod = a * b

PRINT prod AND

GOTO step X

Step VII :IF choice = / THEN

quotient = a / b

PRINT quotient AND

GOTO step X

Step VIII :IF choice = % THEN

remainder = a mod b

PRINT remainder AND

GOTO step X

Step IX :PRINT "Invalid operation".

Step X :STOP.

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int no1,no2,result;
char choice;
clrscr();
printf("Enter the operation to be performed:");
scanf("%d",&no1);
choice=getche();
scanf("%d",&no2);
switch(choice)
{
```

```
case '+':result=no1+no2;
printf("Sum= %d",result);
break;
case '–':result=no1–no2;
printf("Difference= %d",result);
break;
case '*':result=no1*no2;
printf("Product= %d",result);
break;
case '/':result=no1/no2;
printf("Quotient= %d",result);
break;
case '%':result=no1%no2;
printf("Remainder= %d",result);
break;
default:printf("Invalid choice");
}
getch();
}
```

Output

```
Enter the operation to be performed:3
*
7
Product=21
```

➢Program 4.1.15 :Write a program using switch-case to display the class according to the marks scored by a student. The marks scored is taken as input and the class is displayed according to the following range :

Marks	Class
70-100	Distinction
60-69	First Class
50-59	Second Class
40-49	Pass Class
0-39	Fail

➢Algorithm

Step I :START.

Step II :PRINT "Enter marks out of 100".

Step III :INPUT marks

Step IV :IF marks = 100, PRINT "Distinction" AND GOTO step XII

Step V :marks = marks / 10

Step VI:IF marks = 0, 1, 2, or 3 PRINT "Fail" AND GOTO step XII

Step VII :IF marks = 4, PRINT "Pass Class" AND GOTO step XII

Step VIII :IF marks = 5, PRINT "Second Class" AND GOTO step XII

Step IX :IF marks = 6, PRINT "First" AND GOTO step XII

Step X:IF marks = 7, 8, or 9 PRINT "Distinction" AND GOTO step XII

Step XI:PRINT "Invalid marks"

Step XII :Stop.

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int marks;
clrscr();
   printf("Enter the marks out of 100:");
scanf("%d",&marks);
if(marks==100)
printf("Distinction");
else
{
switch(marks/10)
{
case 0:
case 1:
case 2:
case 3:printf("Fail");
break;
case 4:printf("Pass Class");
break;
case 5:printf("Second Class");
break;
case 6:printf("First Class");
break;
case 7:
case 8:
```

```
case 9:printf("Distinction");
break;
default:printf("Invalid marks");
}
}
getch();
}
```

Output

```
Enter the marks out of 100:91
Distinction
```

➢Program 4.1.16 :Write a program to display the month name by accepting the month number from user.

➢Algorithm

Step I :START

Step II :PRINT "Enter month number".

Step III:INPUT n.

Step IV :IF n = 1, THEN PRINT "Jan" AND GOTO step XVII

Step V :IF n = 2, THEN PRINT "Feb" AND GOTO step XVII

Step VI :IF n = 3, THEN PRINT "Mar" AND GOTO step XVII

Step VII :IF n = 4, THEN PRINT "Apr" AND GOTO step XVII

Step VIII :IF n = 5, THEN PRINT "May" AND GOTO step XVII

Step IX :IF n = 6, THEN PRINT "Jun" AND GOTO step XVII

Step X :IF n = 7, THEN PRINT "Jul" AND GOTO step XVII

Step XI :IF n = 8, THEN PRINT "Aug" AND GOTO step XVII

Step XII :IF n = 9, THEN PRINT "Sept" AND GOTO step XVII

Step XIII :IF n = 10, THEN PRINT "Oct" AND GOTO step XVII

Step XIV :IF n = 11, THEN PRINT "Nov" AND GOTO step XVII

Step XV :IF n = 12, THEN PRINT "Dec" AND GOTO step XVII

Step XVI:PRINT "Invalid month number"

Step XVII:Stop.

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int month;
clrscr();
   printf("Enter a month number:");
```

```
scanf("%d",&month);
switch(month)
{
case 1:printf("January");
break;
case 2:printf("February");
break;
case 3:printf("March");
break;
case 4:printf("April");
break;
case 5:printf("May");
break;
case 6:printf("June");
break;
case 7:printf("July");
break;
case 8:printf("August");
break;
case 9:printf("September");
break;
case 10:printf("October");
break;
case 11:printf("November");
break;
case 12: printf("December");
break;
default:printf("Invalid month number");
}
getch();
}
```

Output

```
Enter a month number:8
August
```

Syllabus Topic : Break Statement, Continue Statement, Goto Statement

Q.Explain following statement with example.

(i)goto (ii)continue ***(May 2015)***

Ans. :

- The break statement neglects the statements after it in the loop and transfers the control outside the loop as shown in Fig. 4.1.1. The Fig. 4.1.1 shows the operation of break statement in each of the loop statements i.e. for, while and do-while statements.

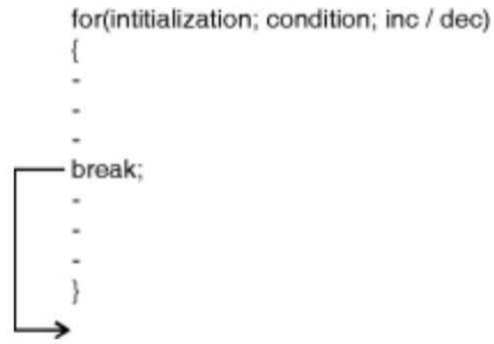


Fig. 4.1.1(a) : Operation of break statement in a for loop

(b) Operation of break statement in a while loop

(c) Operation of break statement in a do-while loop

Fig. 4.1.1

- The continue statement also neglects the statements after it in the loop and transfers the control back to the starting of the loop for next iteration. The Fig. 4.1.2 shows the operation of continue statement in each of the loop statements i.e. for, while and do-while statements.

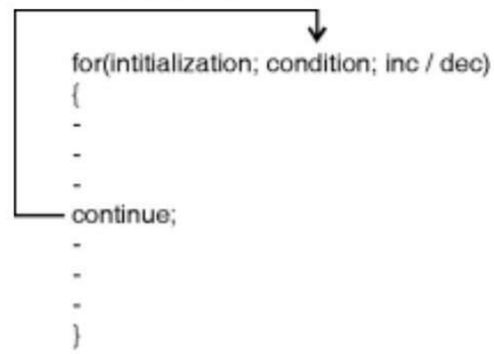
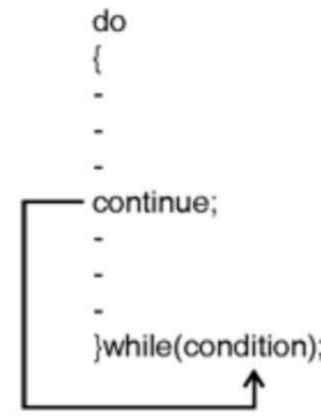


Fig. 4.1.2(a) : Operation of continue statement in a for loop

(b) Operation of continue statement in a while loop



(c) Operation of continue statement in a do-while loop

Fig 4.1.2

- The goto statement transfers the control to the label specified with the goto statement. A label is any name given to a particular part in the program. The label is to be followed with a colon (:).
- The syntax of a goto statement is as shown below :

--

goto label1label2:

--

-**<u>or</u>**-

label1:-

-goto label2

--

- Thus it shows that the label can be before or after the goto statement and hence the control can be transferred to earlier or next statements using a goto statement.

➢Program 4.1.17 :Write a program to demonstrate the use of goto statement.

or

am to accept numbers from user and add them. Stop accepting the numbers if the sum is greater than or equal to 100 and display the sum.

➢Algorithm

Step I :START

Step II:total = 0

Step III :PRINT "Enter a number"

Step IV :INPUT n.

Step V :total = total + n.

Step VI :IF total > = 100, THEN GOTO step VIII

Step VII :GOTO step III

Step VIII :PRINT total

Step IX :Stop.

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int n,total=0;
clrscr();
    again:
printf("Enter a number:");
scanf("%d",&n);
total=total+n;
if(total<100)
goto again;
printf("Total=%d",total);
getch();
}
```

Output

```
Enter a number:56
Enter a number:12
Enter a number:33
Total=101
```

➢Program 4.1.18 :Write a program to demonstrate the use of break statement

or

ram to accept 10, 2-digit numbers from user and add them. If the user enters a three digit number stop accepting the numbers and display the sum.

➢Algorithm

Step I :START

Step II:total = 0

Step III :PRINT "Enter a number"

Step IV :INPUT n.

Step V :IF n > 99, THEN GOTO step VIII

Step VI :total = total + n.

Step VII :GOTO step III

Step VIII :PRINT total

Step IX :Stop.

➢Program

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

```
void main()
{
int n,total=0,i;
clrscr();
    for(i=1;i<=10;i++)
{
printf("Enter a number:");
scanf("%d",&n);
if(n>99)
break;
total=total+n;
}
printf("Total=%d",total);
getch();
}
```

Output

```
Enter a number:45
Enter a number:54
Enter a number:22
Enter a number:59
Enter a number:121
Total=180
```

➢Program 4.1.19 :Write a program to demonstrate the use of continue statement

or

am to accept 5, 2-digit numbers from user and add them. If the user enters a three digit number, this number should be discarded and again accepted. Also an indication must be given to the user that he has entered a number greater than 99 and hence it is not accepted. The sum must be displayed at the end.

➢Algorithm

Step I :START

Step II:total = 0, i = 1

Step III:IF i > 5 THEN GOTO step IX

Step III :PRINT "Enter a number"

Step IV :INPUT n.

Step V :IF n > 99, THEN PRINT "Number is greater than 100" AND i = i – 1 AND GOTO step VII

Step VI :total = total + n.

Step VII :i = i + 1

StepVIII:GOTO step III

Step IX :PRINT total

Step X :Stop.

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int n,total=0,i;
clrscr();
    for(i=1;i<=5;i++)
{
printf("Enter a number:");
scanf("%d",&n);
if(n>99)
{
printf("Number is greater than 99\n");
i--;
continue;
}
total=total+n;
}
printf("Total=%d",total);
getch();
}
```

Output

```
Enter a number:12
Enter a number:121
Number is greater than 99
Enter a number:22
Enter a number:11
Enter a number:89
Enter a number:99
Total=233
```

➢Program 4.1.20 : Write a program to find GCD of two numbers.

➢Algorithm

Step I :START.

Step II :PRINT "Enter two numbers".

Step III :INPUT n1, n2.

Step IV :IF n1 < n2 THEN gcd = n1 ELSE gcd = n2.

Step V :IF (n1 mod gcd = 0 AND n2 mod gcd = 0), THEN GOTO step VIII.

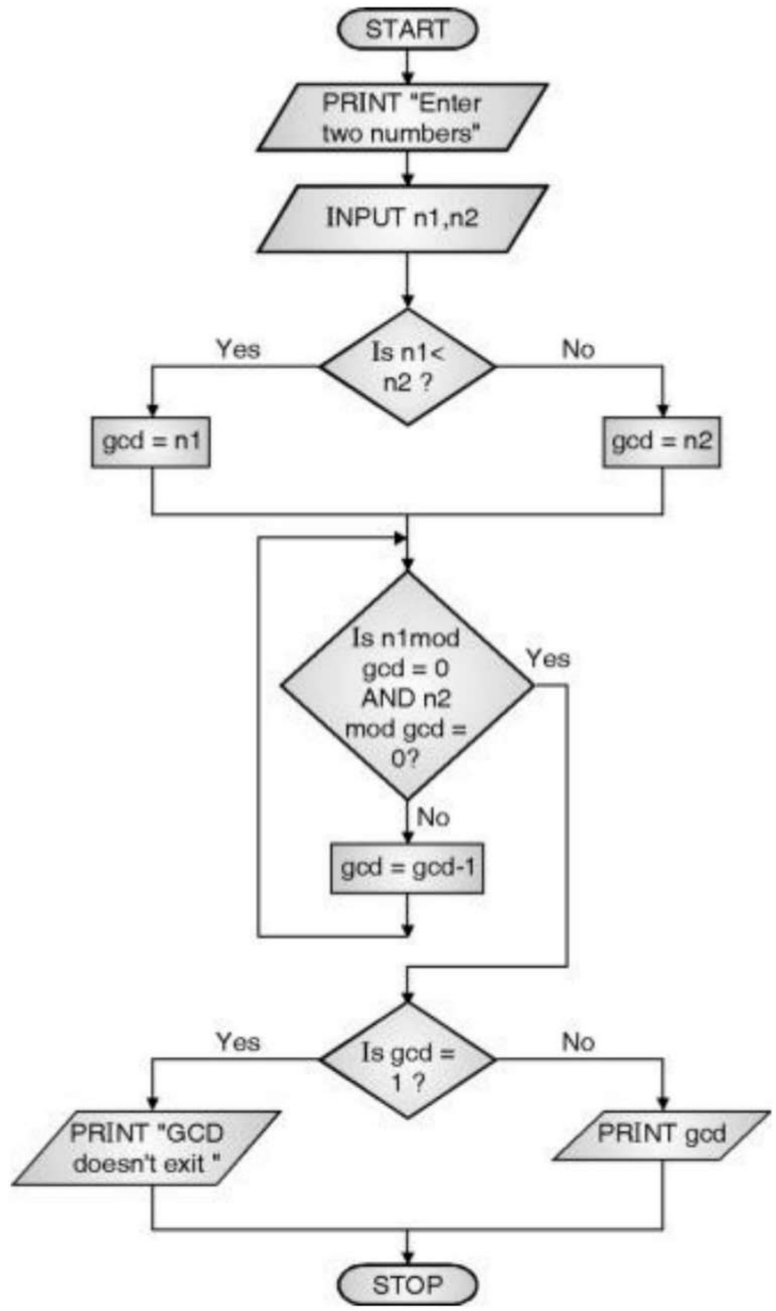
Step VI :gcd = gcd - 1.

Step VII :GOTO step V

Step VIII :IF gcd = 1 THEN PRINT "GCD doesn't exists" ELSE PRINT gcd

Step IX :STOP.

➢Flowchart : (Refer Flowchart 13)



Flowchart 13

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int n1,n2,gcd;
clrscr();
   printf("Enter two numbers:");
```

```
scanf("%d %d",&n1,&n2);
if (n1<n2)
gcd=n1;
else gcd=n2;
while(n1%gcd!=0 || n2%gcd!=0)
{
gcd--;
}
if(gcd==1)
printf("GCD doesnt exists");
else
printf("GCD=%d",gcd);
getch();
}
```

Output

```
Enter two numbers:125
20
GCD=5
```

➢Program 4.1.21 : Write a program to find LCM of two numbers.

➢Algorithm

Step I :START.

Step II :PRINT "Enter two numbers".

Step III :INPUT n1, n2.

Step IV :IF n1 > n2 THEN lcm = n1 ELSE lcm = n2.

Step V :IF (lcm mod n1 = 0 AND lcm mod n2 = 0), THEN GOTO step VIII.

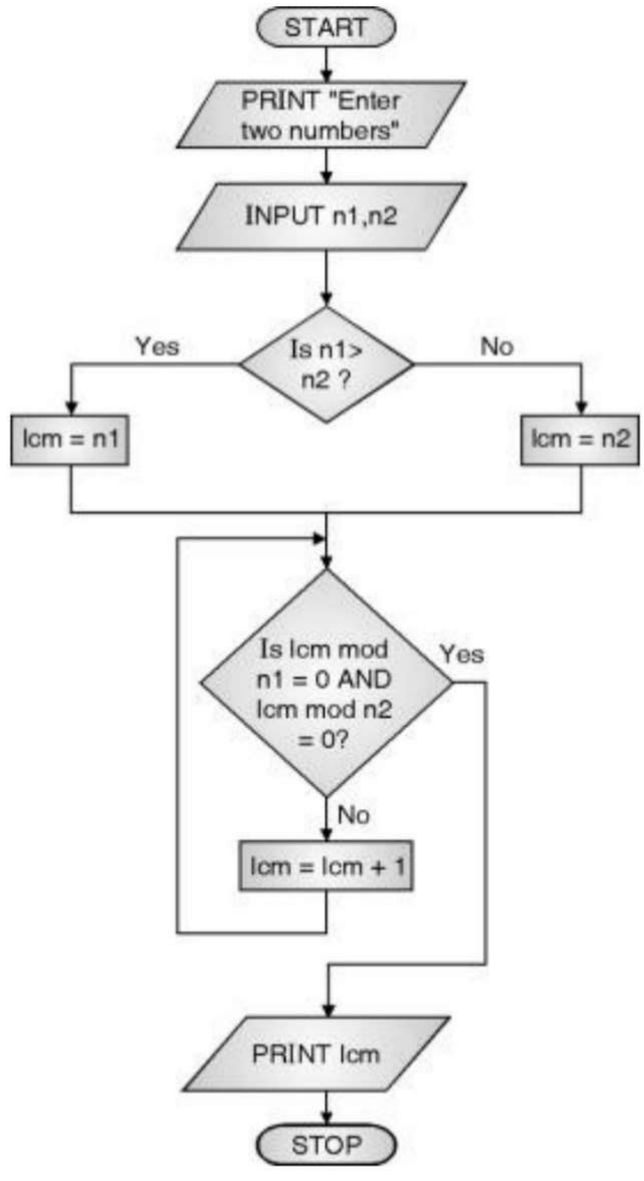
Step VI :lcm = lcm + 1.

Step VII :GOTO step V

Step VIII :PRINT lcm

Step IX :STOP.

➢Flowchart (Refer Flowchart 14)



Flowchart 14

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int n1,n2,lcm;
clrscr();
    printf("Enter two numbers:");
scanf("%d %d",&n1,&n2);
if (n1>n2)
lcm=n1;
else lcm=n2;
while(lcm%n1!=0 || lcm%n2!=0)
{
lcm++;
}
printf("LCM= %d",lcm);
getch();
```

```
}
```

Output

```
Enter two numbers:25
10
```

➢Program 4.1.22 :Write a program to find GCD and LCM of two numbers.***(Dec. 2014)***

➢Algorithm

Step I :START.

Step II :PRINT "Enter two numbers".

Step III :INPUT n1, n2.

Step IV :IF n1 < n2 THEN gcd = n1 ELSE gcd = n2.

Step V:IF n1 > n2 THEN lcm = n1 ELSE lcm = n2.

Step VI :IF (n1 mod gcd = 0 AND n2 mod gcd = 0), THEN GOTO step IX.

Step VII :gcd = gcd – 1.

Step VIII :GOTO step VI

Step IX :IF (lcm mod n1 = 0 AND lcm mod n2 = 0), THEN GOTO step XII.

Step X :lcm = lcm + 1.

Step XI :GOTO step IX

Step XII :IF gcd = 1 THEN PRINT "GCD doesn't exists" ELSE PRINT gcd

Step XIII :PRINT lcm

Step XIV :STOP

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int n1,n2,lcm,gcd;
clrscr();
    printf("Enter two numbers:");
scanf("%d %d",&n1,&n2);
if (n1>n2)
{
lcm=n1;
gcd=n2;
}
else
{
```

```
lcm=n2;
gcd=n1;
}
while(lcm%n1!=0 || lcm%n2!=0)
{
lcm++;
}
while(n1%gcd!=0 || n2%gcd!=0)
{
gcd--;
}
printf("LCM= %d \nGCD=%d",lcm,gcd);
getch();
}
```

Output

```
Enter two numbers:25
10
LCM=50
GCD=5
```

➢Program 4.1.23 : Find the output of the following program.

```
#include<stdio.h>
void main()
{
int i=1,j=1;
for(;;)
{
if(i>3) break;
else j+=i;
printf("%d\n",j);
i+=j;
}
}
```

Output

```
2
5
```

➢Program 4.1.24 : Find the output of the following program

```
#include<stdio.h>
#include<conio.h>
int main()
{
int i;
for(i=0;i<8;i++)
{
if(i%2==0)
printf("%d\n",i+1);
else if(i%3==0)
continue;
else if (i%5==0)
break;
printf("\n End of Program \n");
}
printf("\nEnd of program \n");
}
```

Output

```
1
End of Program
End of Program
3
End of Program
5
End of Program
End of Program
```

➢Program 4.1.25 :Write a program to display Armstrong numbers between 1 to 1000. ***(May 2013)***

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int sum,digit,x=1,copy;
clrscr();
printf("The list of Armstrong number is given below\n");
```

```
while(x<=1000)
{
x++;
copy=x;
sum=0;
while(x!=0)
{
digit=x%10;
sum=sum+digit*digit*digit;
x=x/10;
}
if(sum==copy)
{
printf("%d\n",copy);
n--;
}
x=copy;
}
getch();
}
```

➢Program 4.1.26 Write a program to generate prime nos between 1 to 100.*(Dec. 2013)*

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int i,x=1;
clrscr();
while (x<=100)
{
x++;
i=2;
while(x%i!=0)
{
i++;
}
if(x==i)
printf("%d\n",x);
```

```
}
getch();
}
```

Output

```
2
3
5
7
11
13
17
19
23
29
31
37
41
43
47
53
59
61
67
71
73
79
83
89
97
```

Q.Explain difference between switch statement, ladder of if else and nested if else.***(May 2013)***

Ans. :

Sr. No.	if else statement	switch statement	ladder if else statement	nested if else statement
1.	In this we can test only one condition	In this case we can test multiple conditions	In this case we can test multiple conditions	In this case we can test multiple conditions
2.	In this we cannot have different conditions	In this case we can have only one expression but various value of the same expression	In this case we can have multiple conditions independent of each other.	In this case we can have multiple conditions independent of each other.
3.	This is used when there is only one condition and one value of the same to be tested	This is used when there is only one condition but multiple values of the same to be tested	This is typically used when there are multiple conditions to be tested.	This is typically used when there are multiple conditions to be tested.
4.	It can have values based on constraints	It can have values based on user choice	It can have values based on constraints	It can have values based on constraints
5.	In this first the condition is tested and then it comes to else if the condition is not true	In this case first the case is checked and then it switches to that case	In this first the condition is tested and then it comes to else if the condition is not true and then the other conditions are tested	In this first the condition is tested and then it comes to else if the condition is not true and then the other conditions are tested
6.	if else statement is used to evaluate a condition to be true or false	A switch case statement is used to test for multiple values of the same variable or expression like 1, 2 3 etc	ladder if else is used to test multiple conditions to be true of false and accordingly perform various tasks according to the correct condition	In a nested if else, we can have multiple conditions evaluated based on the previous condition to be true to decide whether to perform a task or not

Q.Difference between break and continue with example. ***(May 2016)***

Ans. :

Sr. No.	Break	Continue
1.	A break can appear in both switch and loop (for, while, do) statements.	A continue can appear only in loop (for, while, do) statements.
2.	A break causes the switch or loop statements to terminate the moment it is executed. Loop or switch ends abruptly when break is encountered.	A continue doesn't terminate the loop, it causes the loop to go to the next iteration. All iterations of the loop are executed even if continue is encountered.
3.	The break statement can be used in both switch and loop statements.	The continue statement can appear only in loops. You will get an error if this appears in switch statement.
4.	When a break statement is encountered, it terminates the block and gets the control out of the switch or loop.	When a continue statement is encountered, it gets the control to the next iteration of the loop.

5.	#include<stdio.h> (1)
	void main()
	{
	int i,n;
	clrscr();
	for(i=0;i<10;i++)
	{
	if(i==5)
	break;
	printf("%d ",i);
	}
	}
	Output:
	1 2 3 4
		#include<stdio.h> (1)
		void main()
		{
		int i,n;
		clrscr();
		for(i=0;i<10;i++)
		{
		if(i==5)
		continue;
		printf("%d ",i);
		}
		}
		Output:
		1 2 3 4 6 7 8 9

➢Program 4.1.27 : An electric power distribution company charges its domestic consumer as follows :

Consumption Units	Rate of Charge
0–200	0.50 per unit
201-400	Rs. 100 plus Rs. 0.65 per unit excess of 200
401-600	Rs. 230 plus Rs. 0.85 per unit excess of 400
601-above	Rs. 390 plus Rs. 1.00 per unit excess of 600

Program should read the units consumed for a customer and calculate the total bill. *(Dec. 2014)*

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int units;
float charge;
```

```
clrscr();
   printf("Enter the units consumed:");
scanf("%d",&units);
switch(units/200)
{
case 0: charge=units *0.5;
  break;
case 1:charge=(units-200) *0.65 +100;
 break;
case 2: charge=(units-400) *0.85 +230;
 break;
default:charge=(units-600) +390;
 break;
}
printf("Charges=%f",charge);
getch();
}
```

CHAPTER

5

Functions and Parameter

Syllabus Topic : Introduction of a function, function main, Defining a Function, accessing a function, function prototype, passing arguments to function

Q.Explain the library functions in math.h along with its uses ***(May 2013, Dec. 2013, Dec. 2015)***

Ans. :

1.pow(x,y)

- This function is available in the header file "math.h". This function calculates and returns the value of x^y. The parameters x and y are to be passed to it.
- The parameters as well the answer returned are of double data type. But it can also accept parameters of different data type. The prototype of the function is as given below :

double pow (double x, double y)

2.sqrt(x)

- This function finds the square root of the parameter passed to it and the result is returned to the caller function. It is available in math.h.
- This function also accepts a double as input parameter and returns the result also of double data type. The prototype of the function is as given below :

double sqrt (double x)

3.ceil(x)

- This function returns the smallest integral value of the parameter passed to the function. It rounds the value of x upward and returns the value.
- This function is also in the header file math.h. The parameter accepted and returned of the data type double. The prototype of the function is given below :

double ceil (double x)

4.floor(x)

- This function returns the largest integral value of the parameter passed to the function. It rounds the value of x downward and returns the value.
- This function is also in the header file math.h. The parameter accepted and returned is of double data type. The prototy[e of the function is given below:

double floor (double x)

5.abs()

abs( ) function in C returns the absolute value of an integer. The absolute value of a number is always positive. Only integer values are supported in C.

"stdlib.h" header file supports abs( ) function in C language. Syntax for abs( ) function in C is :

int abs (int n);

6.insalnum()

Insalnum() checks if the ASCII value passed in has a character equivalent to a number of letter. It returns non-zero for true, and zero for false.

Syntax for insalnum () function : int Insalnum(int ch);

ctype.h header file supports insalnum () function.

➢Program 5.2.1 : Write a program to add two numbers using function.

➢Algorithm

main() function

Step I :START

Step II :PRINT "Enter two numbers".

Step III :INPUT a,b

Step IV:sum = CALL add (arguments: a,b).

Step V :PRINT sum

Step VI :Stop.

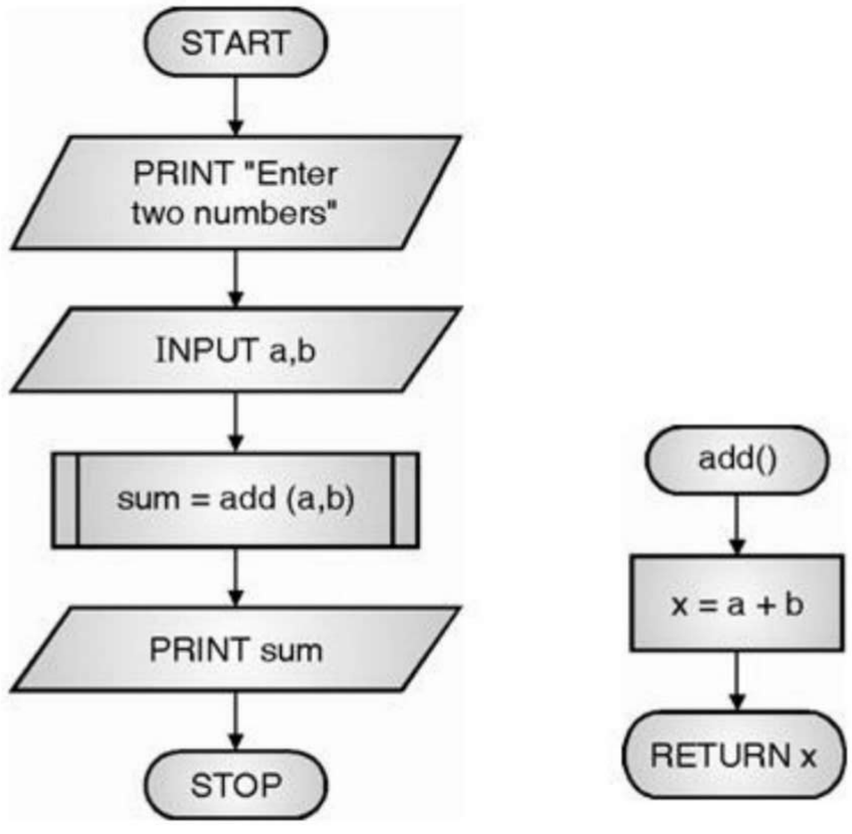
add(parameters: a, b)

Step I :START.

Step II :x = a + b.

Step III :RETURN (x).

➢Flowchart : (Refer Flowchart 1)



Flowchart 1

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int n1,n2,sum;
int add (int a, int b);
clrscr();
   printf("Enter two numbers:");
scanf("%d %d",&n1,&n2);
sum=add(n1,n2);
printf("Sum=%d",sum);
getch();
}
int add (int a, int b)
{
int c;
c=a+b;
return c;
}
```

Output

```
Enter two numbers:4
6
Sum=10
```

➢Program 5.2.2 : Write a program to add two numbers using a void function.

➢Algorithm

main() function

Step I :START

Step II :PRINT "Enter two numbers".

Step III :INPUT a,b

Step IV:CALL add (arguments: a,b).

Step V :Stop.

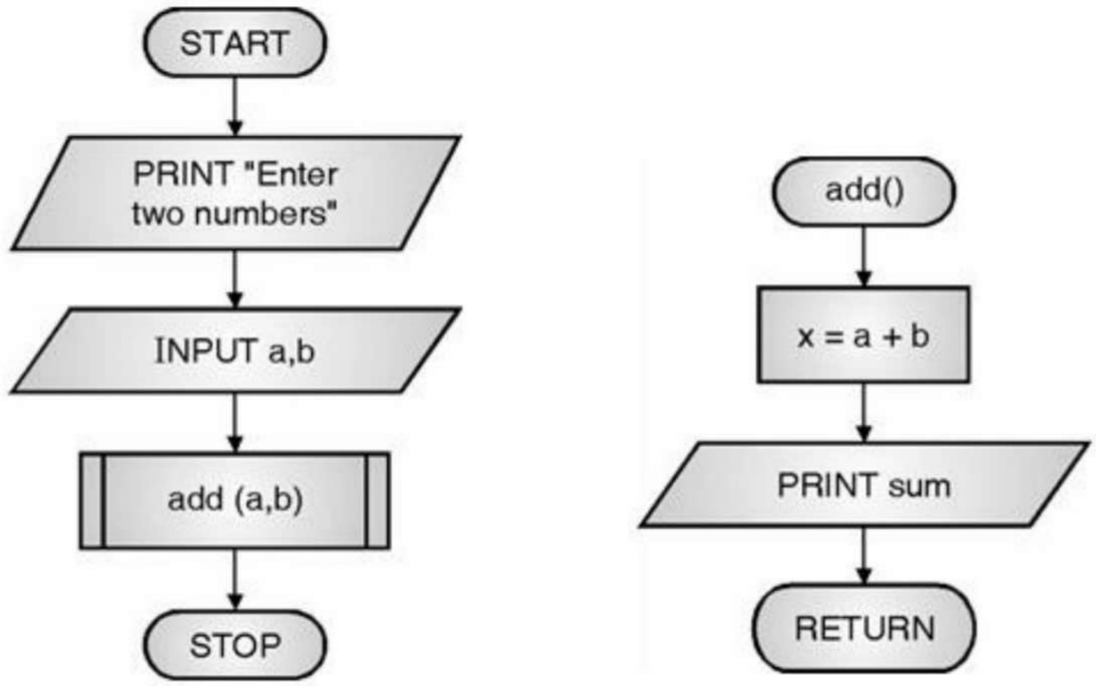
add(parameters: a, b)

Step I :START.

Step II :x = a + b.

Step III :PRINT x.

➢Flowchart : (Refer Flowchart 2)



Flowchart 2

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int n1,n2;
void add (int a, int b);
clrscr();
printf("Enter two numbers:");
scanf("%d %d",&n1,&n2);
add(n1,n2);
getch();
}
void add (int a, int b)
{
int c;
c=a+b;
printf("sum=%d",c);
}
```

Output

```
Enter two numbers:4
5
sum=9
```

➢Program 5.2.3 :Write a program to find the average of three numbers using a function.

➢Algorithm

main() function

Step I :START

Step II :PRINT "Enter three numbers".

Step III :INPUT a,b, c

Step IV:CALL average (arguments: a,b, c).

Step V :STOP.

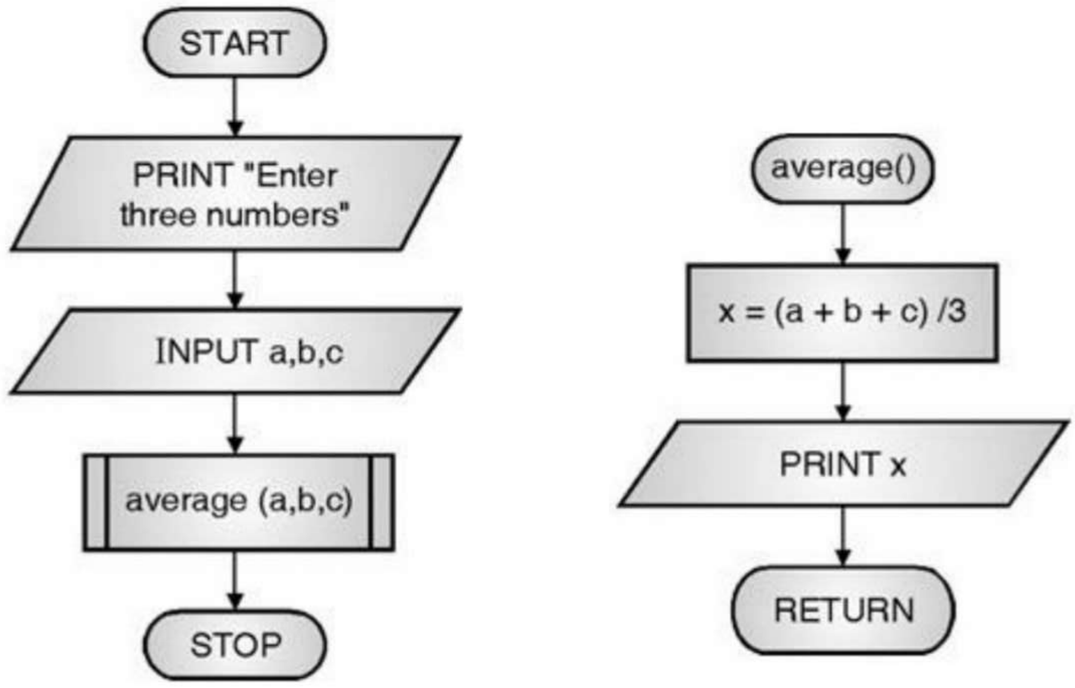
average(parameters: a, b, c)

Step I :START.

Step II :x = (a + b + c) / 3.

Step III :PRINT x.

➢Flowchart : (Refer Flowchart 3)



Flowchart 3

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int n1,n2,n3;
void avg (int a, int b, int c);
clrscr();
printf("Enter three numbers:");
scanf("%d %d %d",&n1,&n2,&n3);
avg(n1,n2,n3);
getch();
}
void avg (int a, int b, int c)
{
```

```
float average;
average=(a+b+c)/3.0;
printf("Average=%f",average);
}
```

Output

```
Enter three numbers:4
6
3
Average=4.333333
```

➢Program 5.2.4 : Write a program to find the factorial of a number using a function. ***(May 2013)***

➢Algorithm

main() function

Step I :START

Step II :PRINT "Enter a number".

Step III :INPUT a

Step IV:fact = CALL factorial (arguments: a).

Step V :PRINT fact

Step VI:STOP.

factorial (parameters: a)

Step I :START.

Step II :i = 1, fact = 1

Step III :IF i > a, THEN GOTO step VII

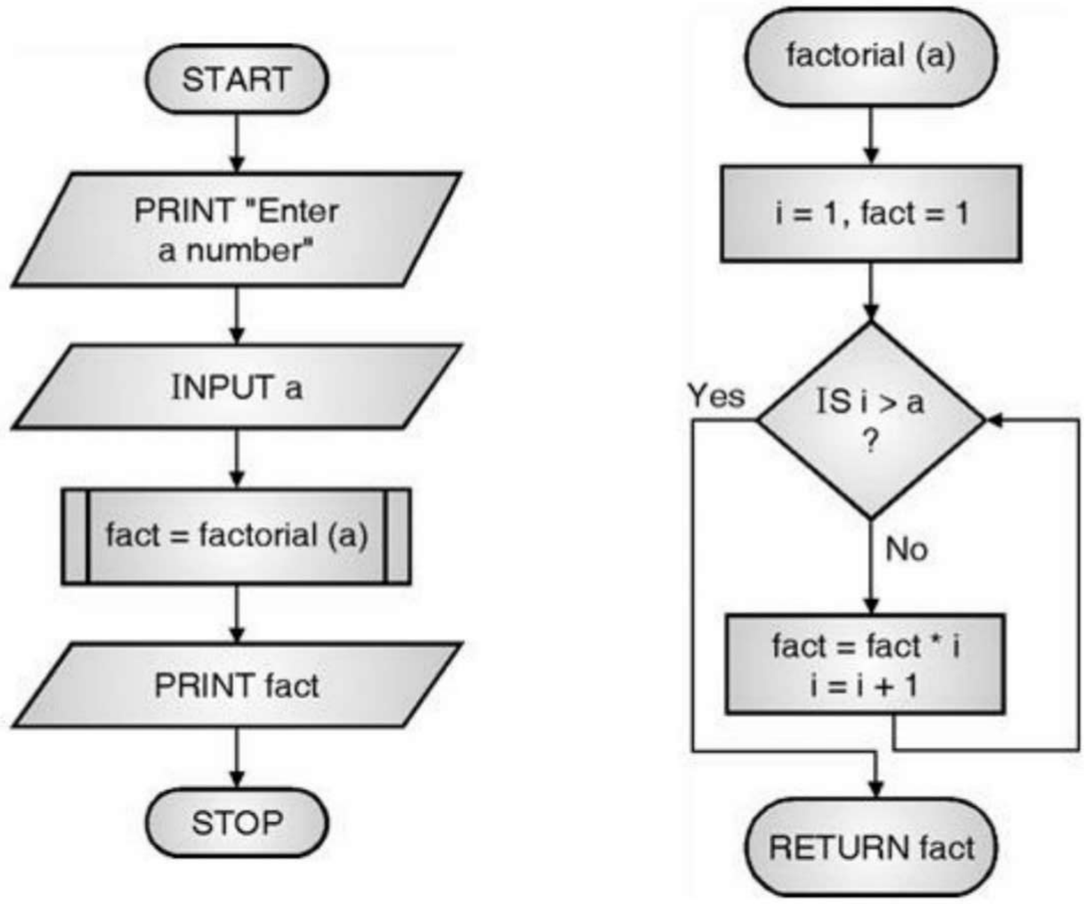
Step IV:fact = fact * i

Step V:i = i + 1

Step VI:GOTO step III

Step VII:RETURN (fact)

➢Flowchart : (Refer Flowchart 4)



Flowchart 4

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int no,factorial;
int fact (int no);
clrscr();
printf("Enter a number:");
scanf("%d",&no);
factorial=fact(no);
printf("Factorial=%d",factorial);
getch();
}
int fact (int no)
{
int i,ans;
for(i=1,ans=1;i<=no;i++)
{
ans=ans*i;
}
return ans;
}
```

Output

```
Enter a number:4
Factorial=24
```

➢Program 5.2.5 :Write a program to find the value of ${}^{n}C_{r}$ (Combination), using a function.

main() function

Step I :Declare the required variables and the prototype of the function add().

Step II :Indicate the user to enter the values of n and r, by displaying suitable sentence using printf() function.

main() function

Step I :START

Step II :PRINT "Enter the values of n and r".

Step III :INPUT n, r

$$\textbf{Step IV} \quad : \quad nCr = \frac{\text{CALL factorial (arguments : n)}}{(\text{CALL factorial (arguments : r)} * \text{CALL factorial (arguments; n} - \text{r))}}$$

Step V :PRINT fact

Step VI:STOP.

factorial (parameters: a)

Step I :START.

Step II :i = 1, fact = 1

Step III :IF i > a, THEN GOTO step VII

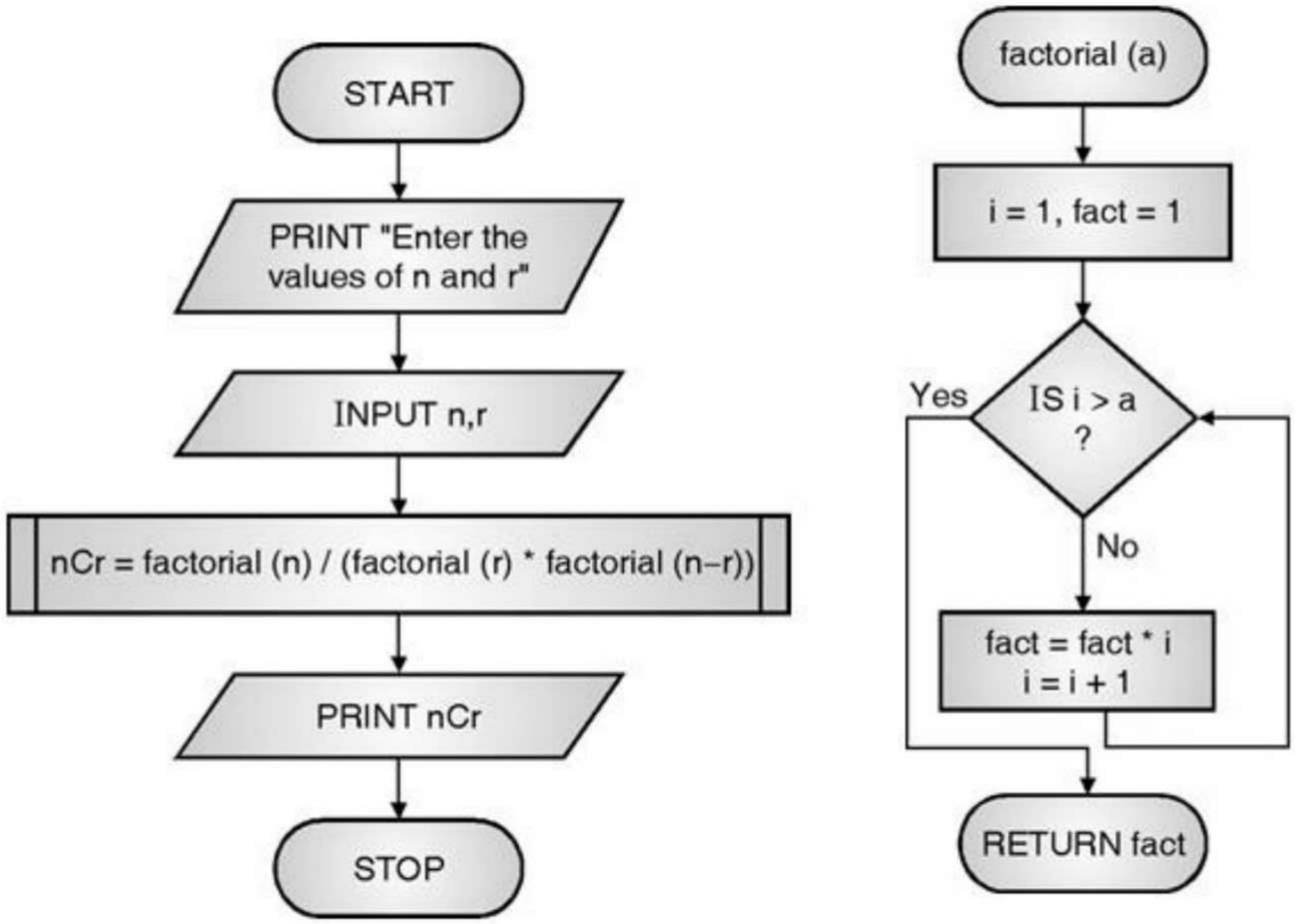
Step IV:fact = fact * i

Step V:i = i + 1

Step VI:GOTO step III

Step VII:RETURN (fact)

➢Flowchart : (Refer Flowchart 5)



Flowchart 5

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int n,r,ncr;
int fact (int no);
clrscr();
printf("Enter the values of n and r:");
scanf("%d %d",&n,&r);
ncr=fact(n)/(fact(r)*fact(n–r));
printf("nCr=%d",ncr);
getch();
}
int fact (int no)
{
int i,ans;
for(i=1,ans=1;i<=no;i++)
{
ans=ans*i;
}
return ans;
}
```

Output

```
Enter the values of n and r:5
3
nCr=10
```

➢Program 5.2.6 :Write a program to find the value of ${}^{n}P_{r}$ (Permutation), using a function.

main() function

Step I :START

Step II :PRINT "Enter the values of n and r".

Step III :INPUT n, r

$$\textbf{Step IV} \quad : \quad nCr = \frac{\text{CALL factorial (arguments:n)}}{\text{CALL factorial (arguments; n} - \text{r)}}$$

Step V :PRINT fact

Step VI:STOP.

factorial (parameters: a)

Step I :START.

Step II :i = 1, fact = 1

Step III :IF i > a, THEN GOTO step VII

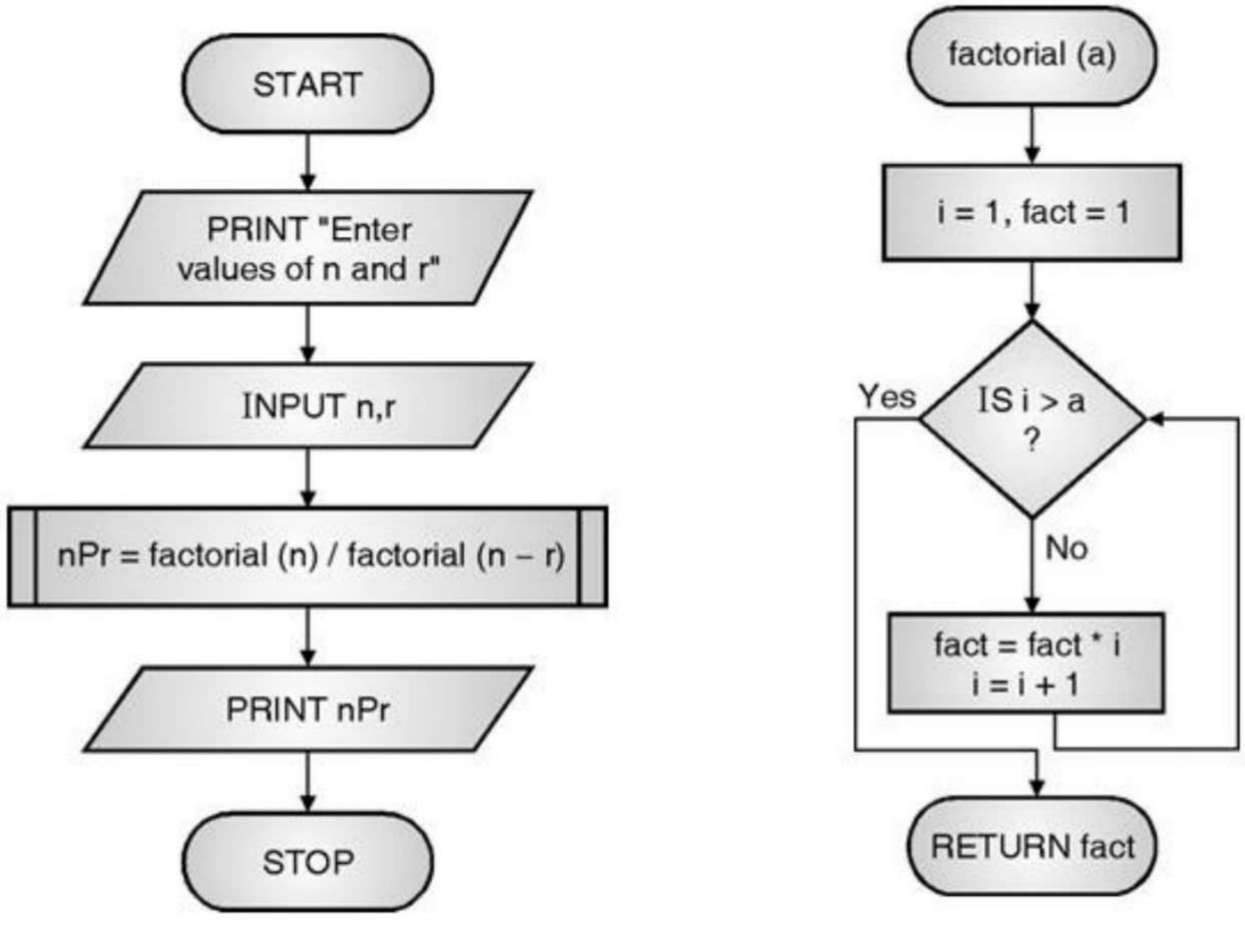
Step IV:fact = fact * i

Step V:i = i + 1

Step VI:GOTO step III

Step VII:RETURN (fact)

➢Flowchart : (Refer Flowchart 6)



Flowchart 6

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int n,r,npr;
int fact (int no);
clrscr();
printf("Enter the values of n and r:");
scanf("%d %d",&n,&r);
npr=fact(n)/fact(n–r);
printf("nPr=%d",npr);
getch();
}
int fact (int no)
{
int i,ans;
for(i=1,ans=1;i<=no;i++)
{
ans=ans*i;
}
return ans;
}
```

Output :

```
Enter the values of n and r:5
3
nPr=60
```

➢Program 5.2.7 :Write a program to find the GCD of two numbers, using a function.

main() function

Step I :START

Step II :PRINT "Enter two numbers"

Step III :INPUT x, y

Step IV :gcd = CALL GCD(arguments: x, y)

Step V :IF gcd = 1 THEN PRINT "GCD doesn't exists" ELSE PRINT gcd

Step VI :STOP.

GCD(paramters: x, y) function

Step I :START.

Step II :IF n1 < n2 THEN gcd = n1 ELSE gcd = n2.

Step III :IF (n1 mod gcd = 0 AND n2 mod gcd = 0), THEN GOTO step VIII.

Step IV :gcd = gcd – 1.

Step V :GOTO step III

Step VI :RETURN (gcd)

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int no1,no2,gcd;
int GCD (int no1, int no2);
clrscr();
printf("Enter two numbers:");
scanf("%d %d",&no1,&no2);
gcd=GCD(no1,no2);
if(gcd==1)
printf("GCD doesnt exist");
else
printf("GCD=%d",gcd);
getch();
}
int GCD (int no1, int no2)
{
int gcd;
if(no1<no2)
gcd=no1;
else
gcd=no2;
while(no1%gcd!=0 || no2%gcd!=0)
{
gcd--;
}
return gcd;
}
```

Output

```
Enter two numbers:10
25
GCD=5
```

➢Program 5.2.8 :Write a program to find the LCM of two numbers, using a function.

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int no1,no2,lcm;
int LCM (int no1, int no2);
clrscr();
printf("Enter two numbers:");
scanf("%d %d",&no1,&no2);
lcm=LCM(no1,no2);
printf("LCM=%d",lcm);
getch();
}
int LCM (int no1, int no2)
{
int lcm;
if(no1>no2)
lcm=no1;
else
lcm=no2;
while(lcm%no1!=0 || lcm%no2!=0)
{
lcm++;
}
return lcm;
}
```

Output

```
Enter two numbers:14
2
LCM=14
```

Syllabus Topic : Recursion

Q.What is recursion ? ***(May 2013, Dec. 2013, May 2014, May 2015, Dec. 2015, May 2016)***

Ans. :

- A function that calls itself is called as a recursive function and the implementation of a program that uses recursive function is called as recursion.

- A recursive function must definitely have a condition that exits from calling the function again.
- Hence there must be a condition that calls the function itself if that condition is true.
- If the condition is false then it will exit from the loop of calling itself again.
- The condition could also be implemented vice versa i.e. if the condition is false then the function calls itself else if the condition is true, it will exit from the loop calling itself again.

➢Program 5.3.1 :Write a program to find the factorial of a number, using a recursive function.

➢Algorithm

main() function

Step I :START

Step II :PRINT "Enter a number".

Step III :INPUT a

Step IV:fact = CALL factorial (arguments: a).

Step V :PRINT fact

Step VI:STOP.

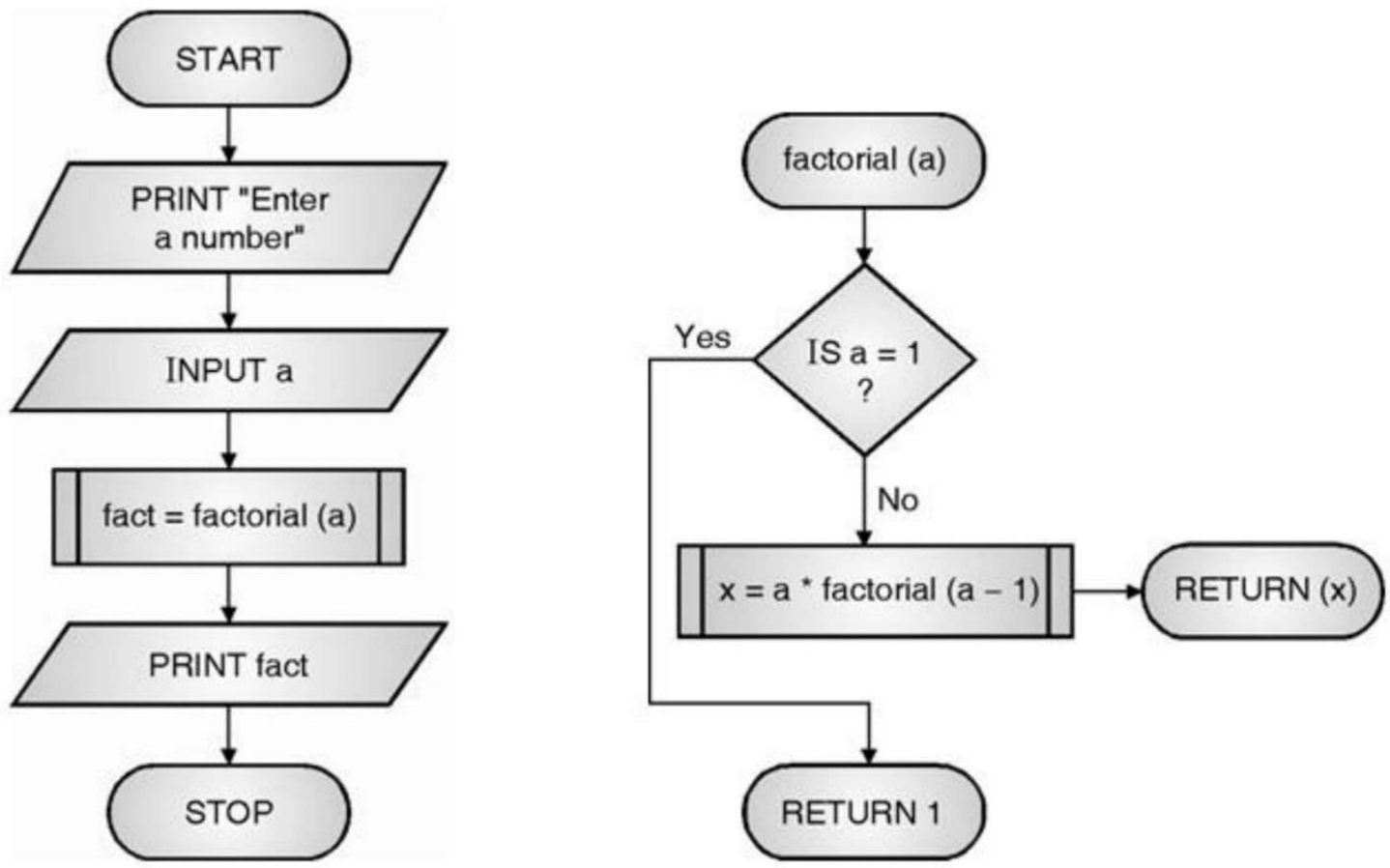
factorial (parameters: a)

Step I :START.

Step II :IF a = 1, THEN RETURN (1)

ELSE RETURN (a * CALL factorial(arguments: a – 1))

➢Flowchart : (Refer Flowchart 7)



Flowchart 7

➢Program

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

```
#include<conio.h>
void main()
{
int no,factorial;
int fact (int no);
clrscr();
printf("Enter a number:");
scanf("%d",&no);
factorial=fact(no);
printf("Factorial=%d",factorial);
getch();
}
int fact (int no)
{
if(no==1)
return 1;
else
return (no * fact (no-1));
}
```

Output

```
Enter a number:5
Factorial=120
```

➢Program 5.3.2 :Write a program to find n Fibonacci elements, using a recursive function. ***(May 2013, Dec. 2014)***

➢Algorithm

main() function

Step I :START

Step II :PRINT "Enter a number".

Step III :INPUT n

Step IV:PRINT "0"

Step V:i = 1

Step VI:IF i > n – 1, THEN GOTO step

Step VII:x = CALL fibo (arguments: 0, 1, i).

Step VIII:PRINT x

Step IX:i = i + 1

Step X:GOTO step VI

Step XI:STOP.

fibo (parameters: a, b, i)

Step I :START.

Step II :IF i = 1, THEN RETURN (b)

ELSE RETURN (CALL (fibo(arguments: b, a + b, i – 1))

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int n,i;
int fibo (int ,int ,int);
clrscr();
printf("Enter the number of elements:");
scanf("%d",&n);
printf("0\n");
for(i=1;i<=n–1;i++)
{
printf("%d\n",fibo(0,1,i));
}
getch();
}
int fibo (int a, int b, int i)
{
if(i==1)
return b;
else
return (fibo(b,a+b,– –i));
}
```

Output

```
Enter the number of elements:10
1
1
2
3
5
8
13
21
```

➢Program 5.3.3 : Write a program to find value of y using recursive function, where $y = x^n$.*(Dec. 2013, Dec. 2014)*

➢Algorithm

main() function

Step I :START

Step II :PRINT "Enter values of x and n".

Step III :INPUT x, n

Step IV:y = CALL exponential (arguments: x, n).

Step V :PRINT y

Step VI:STOP.

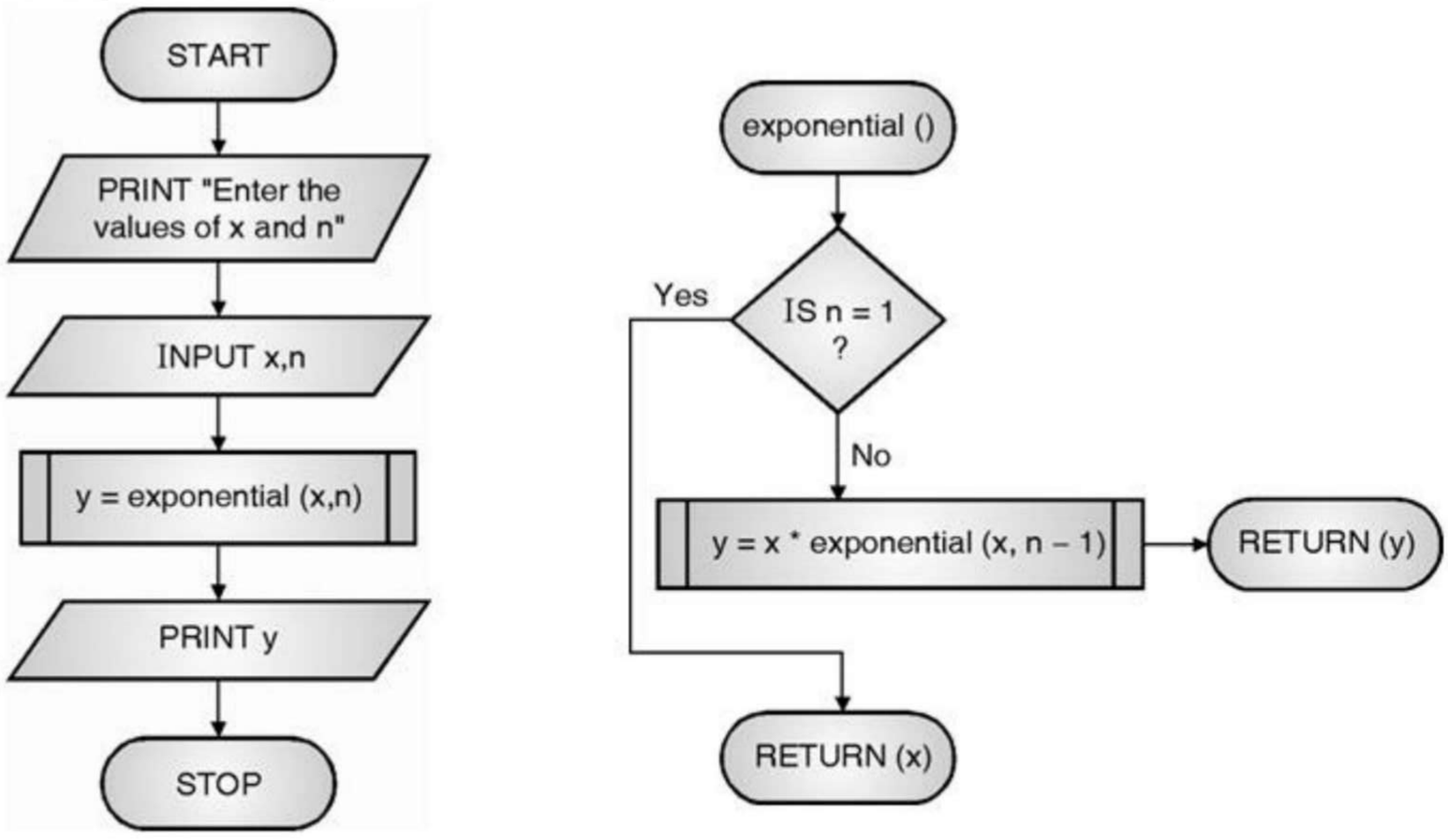
exponential (parameters: x, n)

Step I :START.

Step II :IF n = 1, THEN RETURN (x)

ELSE RETURN (x * CALL exponential (arguments: x, n – 1))

➢Flowchart : (Refer Flowchart 8)



Flowchart 8

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int n,x,y;
```

```
int exponential (int x, int n);
clrscr();
printf("Enter the values of x and n:");
scanf("%d %d",&x,&n);
y=exponential(x,n);
printf("The value of x raise to n is %d",y);
getch();
}
int exponential (int x, int n)
{
if(n==1)
return x;
else
return (x*exponential(x,n–1));
}
```

Output

```
Enter the values of x and n:2
6
The value of x raise to n is 64
```

➢Program 5.3.4 :Write a recursion function to compute the sum of first 'n' number beginning with 1. ***(May 2015, Dec. 2015)***

➢Algorithm

main() function

Step I :START

Step II :PRINT "Enter a number".

Step III :INPUT a

Step IV:sum = CALL recursion (arguments: a).

Step V :PRINT sum

Step VI:STOP.

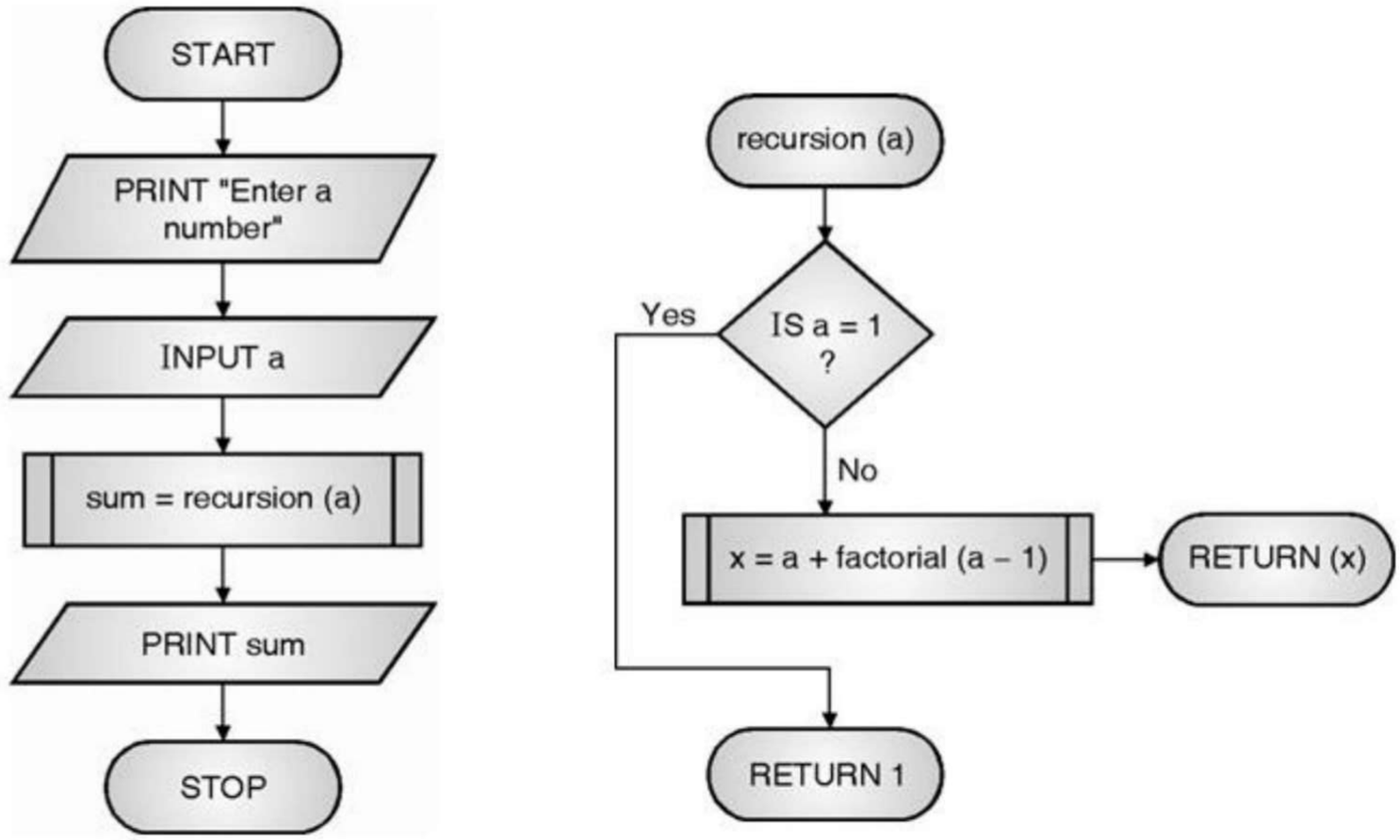
recursion (parameters: a)

Step I :START.

Step II :IF a = 1, THEN RETURN (1)

ELSE RETURN (a + CALL recursion(arguments: a–1))

➢Flowchart : (Refer Flowchart 9)



Flowchart 9

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int no,sum;
int recursion (int no);
clrscr();
printf("Enter a number:");
scanf("%d",&no);
sum=recursion(no);
printf("Sum of numbers from 1 to n is %d",sum);
getch();
}
int recursion (int no)
{
if(no==1)
return 1;
else
return (no + recursion (no–1));
}
```

Output

```
Enter a number:10
Sum of numbers from 1 to n is 55
```

➢Program 5.3.5 :What do you mean by recursion? Write a program which will accept two numbers, n and r and calculate value of nCr = n! /r!(n- r)! . Program should make use of recursion.***(May 2014)***

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int n,r,ncr;
int fact (int no);
clrscr();
printf("Enter the values of n and r:");
scanf("%d %d",&n,&r);
ncr=fact(n)/(fact(r)*fact(n–r));
printf("nCr=%d",ncr);
getch();
}
int fact (int no)
{
int i,ans;
for(i=1,ans=1;i<=no;i++)
{
ans=ans*i;
}
return ans;
}
```

Output

```
Enter the values of n and r:5
3
nCr=10
```

➢Program 5.4.1 : Write a program using function, to display the first n natural numbers, where the value of n is taken from user

➢Program

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

```
void main()
{
int n;
void display (int n);
clrscr();
printf("Enter a number: ");
scanf("%d",&n);
display(n);
getch();
}
void display (int n)
{
int i;
for(i=1;i<=n;i++)
{
printf("%d\n",i);
}
}
```

Output

```
Enter a number: 7
1
2
3
4
5
6
7
```

➢Program 5.4.2 : Write a program using function, to display the multiplication table of a user entered number. The table must be upto 10.

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int i,n;
void mul(int n);
clrscr();
printf("Enter a number: ");
```

```
scanf("%d",&n);
mul(n);
getch();
}
void mul(int n)
{
int i;
for(i=1;i<=10;i++)
{
printf("%d X %d = %d\n",n,i,(n*i));
}
}
```

Output

```
Enter a number: 5
5 X 1 = 5
5 X 2 = 10
5 X 3 = 15
5 X 4 = 20
5 X 5 = 25
5 X 6 = 30
5 X 7 = 35
5 X 8 = 40
5 X 9 = 45
5 X 10 = 50
```

➢Program 5.4.3 : Write a program using function, to display first n odd numbers.

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int n;
void odd(int n);
clrscr();
printf("Enter a number: ");
scanf("%d",&n);
odd(n);
getch();
}
```

```
void odd(int n)
{
int i;
for(i=0;i<=n-1;i++)
{
printf("%d\n",2*i+1);
}
}
```

Output

```
Enter a number: 6
1
3
5
7
9
11
```

➢Program 5.4.4 : Write a program using function, to display odd numbers upto n.

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int i,n;
void odd(int n);
clrscr();
printf("Enter a number: ");
scanf("%d",&n);
odd(n);
getch();
}
void odd(int n)
{
int i;
for(i=1;i<=n;i+=2)
{
printf("%d\n",i);
}
}
```

Output

```
Enter a number: 10
1
3
5
7
9
```

➢Program 5.4.5 : Write a program using function, to display first n elements of Fibonacci series.

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int n;
void fibo(int n);
clrscr();
printf("Enter a number: ");
scanf("%d",&n);
printf("Fibonacci Series\n0\n1\n");
fibo(n);
getch();
}
void fibo(int n)
{
int a=0,b=1,c,i;
for(i=1;i<=n-2;i++)
{
c=a+b;
printf("%d\n",c);
a=b;
b=c;
}
}
```

Output

```
Enter a number: 10
Fibonacci Series
1
1
```

```
2
3
5
8
13
21
34
```

Note : Fibonacci series is a series wherein the first two elements are 0 and 1. Thereafter the next values are the sum of previous two elements. Hence it can be said that the series is 0,1,1,2,3,5,8,13,21,34,55 . . .

➢Program 5.4.6 :Write a program using function, to display the following for the user specified number of lines

```
*
**
***
****
*****
|
|
n lines
```

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int n;
void pattern(int n);
clrscr();
printf("Enter the number of lines:");
scanf("%d",&n);
pattern(n);
getch();
}
void pattern(int n)
{
int i,j;
for(i=1;i<=n;i++)
{
for(j=1;j<=i;j++)
{
printf("*");
```

```
}
printf("\n");
}
}
```

Output

```
Enter the number of lines:7
*
**
***
****
*****
******
*******
```

➢Program 5.4.7 :Write a program using function, to display the following asking the user for the number of lines.

```
1
12
123
1234
12345
123456
```

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int n;
void pattern(int n);
clrscr();
printf("Enter the number of lines:");
scanf("%d",&n);
pattern(n);
getch();
}
void pattern(int n)
{
int i,j;
for(i=1;i<=n;i++)
{
```

```
for(j=1;j<=i;j++)
{
printf("%d",j);
}
printf("\n");
}
}
```

Output

```
Enter the number of lines:6
1
12
123
1234
12345
123456
```

➢Program 5.4.8 :Write a program using function, to display the following asking the user for the number of lines.

A

ABA

ABCBA

ABCDCBA

ABCDEDCBA

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int n;
void pattern(int n);
clrscr();
printf("Enter the number of lines:");
scanf("%d",&n);
pattern(n);
getch();
}
void pattern(int n)
{
int i,j;
for(i=1;i<=n;i++)
```

```
{
for(j=1;j<=n-i;j++)
{
printf(" ");
}
for(j=1;j<=i;j++)
{
printf("%c",(char)(j+64));
}
for(j=i-1;j>=1;j--)
{
printf("%c",(char)(j+64));
}
printf("\n");
}
}
```

Output

```
Enter the number of lines:6
A
ABA
ABCBA
ABCDCBA
ABCDEDCBA
ABCDEFEDCBA
```

➢Program 5.4.9 :Write a program using function, to check if the entered number is prime number or not.

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int n;
void prime(int n);
clrscr();
printf("Enter a number:");
scanf("%d",&n);
prime(n);
getch();
}
```

```
void prime(int n)
{
int i=2;
while(n%i!=0)
{
i++;
}
if(n==i)
{
printf("Prime Number");
}
else
{
printf("Not a prime number");
}
}
```

Output

```
Enter a number:17
Prime Number
```

➢Program 5.4.10 :Write a program using function, to display a user entered number in words.

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int n;
void digits(int n);
clrscr();
    printf("Enter a number:");
scanf("%d",&n);
digits(n);
getch();
}
void digits(int n)
{
int digit,rev=0;
while(n!=0)
{
```

```
digit=n%10;
rev=rev*10+digit;
n=n/10;
}
while(rev!=0)
{
digit=rev%10;
rev=rev/10;
switch(digit)
{
case 0:printf("Zero ");
break;
case 1:printf("One ");
break;
case 2:printf("Two ");
break;
case 3:printf("Three ");
break;
case 4:printf("Four ");
break;
case 5:printf("Five ");
break;
case 6:printf("Six ");
break;
case 7:printf("Seven ");
break;
case 8:printf("Eight ");
break;
case 9:printf("Nine ");
break;
}
}
}
```

Output

```
Enter a number:513
Five One Three
```

➢Program 5.4.11 :Write a program to reverse a number using recursion.***(May 2016)***

➢Program

```
#include<stdio.h>
intmain()
{
intnum,reverse;
printf("Enter any number: ");
scanf("%d",&num);
reverse=rev(num);
printf("Reverse of number: %d",reverse);
return0;
}
intrev(intnum)
 {
staticsum,r;
if(num)
    {
r=num%10;
sum=sum*10+r;
rev(num/10);
   }
else
return0;
returnsum;
  }
```

Sample output

```
Enter any number: 456
Reverse of number: 654
```

➢Program 5.4.12 :Write a program to calculate compound interest and amount.

Using formula A=P(1+R/100)^n, where P=Principal Amt., R is Rate of interest , n= number of years. Your program should make use of user defined function to calculate power. Program should accept P, R and N, Display interest earned for each year. ***(May 2016)***

➢Program

```
#include<stdio.h>
#include<math.h>
void main()
{
float R,P,CI;
int N;
```

```
float comp_int_calc(float,float,int);
clrscr();
```

```
printf("ENTER THE PRINCIPAL AMOUNT : ");
```

```
scanf("%f",&P);
```

```
printf("ENTER THE NUMBER OF YEAR(S) : ");
```

```
scanf("%d",&N);
```

```
printf("ENTER THE RATE OF INTEREST(%) : ");
```

```
scanf("%f",&R);
R =R/100;
CI =comp_int_calc(P,R,N);
```

```
printf("THE CALCULATED SIMPLE INTEREST IS RUPEES : %f",CI);
```

```
getch();
}
float comp_int_calc(float AMT,float RATE,int YEARS)
{
  float COMP_INT=0;
  COMP_INT =pow(1+RATE,YEARS);
  COMP_INT =AMT*COMP_INT;
  return COMP_INT;
}
```

Syllabus Topic : Storage Classes- Auto, Extern, Static, Register

Q. Explain the different storage classes like auto, static, register and extern. ***(May 2015, Dec. 2015, May 2016)***

Ans. :

- The different locations in the computer where we can store data and their accessibility, initial values etc. vary based on the way they are declared. These different ways are termed as different storage classes.
- In C, we have four storage classes, namely

1. Automatic
2. Register
3. Static
4. External or Global

- Let us see these storage classes one by one

1.Automatic storage class

- In this case data is stored in memory

- The initial value of such a variable is garbage.
- The scope of the variable is local i.e. limited to the function in which it is defined.
- The life of such variables is till the control remains in the particular function where it is defined.
- For e.g.:

int i; or auto int i;

- In all our programs till now we have been using the automatic storage class for our variables.

2. Register storage class

- In this case data is stored in CPU register
- The initial value of such a variable is garbage.
- The scope of the variable is local i.e. limited to the function in which it is defined.
- The life of such variables is till the control remains in the particular function where it is defined.

For e.g.:

register int i;

- In this case the data is stored in a small memory inside the processor called as its registers.
- The advantage of such storage class is that since the data is in the processor itself, its access and operations on such data is faster.
- There is a limitation on the size of the data that can be declared to be register storage class. The data should be such that it doesn't require more than 4 bytes. Hence double and long double data types cannot be declared as register.
- Also there is a limitation on the maximum number of variables in a function that can be of register class. The limitation is that a maximum of 3 register class variable can be declared in a function.

3. Static storage class

- In this case data is stored in memory
- The initial value of such a variable is zero.
- The scope of the variable is local i.e. limited to the function in which it is defined.
- The life of such variables is till the program is alive.

For e.g. :

static int i;

- If a variable is declared static, its value remains unchanged even if the function execution is completed.
- When the execution to that function returns, the previous value is retained.
- Thus it says the initialization is only once. If you have an initialization statement of a static member, it will be executed only once i.e. for the first time when this function is called.
- Example of register storage class: Addition of Two numbers

```
#include<stdio.h>
int main()
{
int num1,num2;
register int sum;
printf("\nEnter the Number 1 : ");
scanf("%d",&num1);
printf("\nEnter the Number 2 : ");
scanf("%d",&num2);
sum = num1 + num2;
printf("\nSum of Numbers : %d",sum);
return(0);
}
```

➢Program 5.5.1 : Write a program to initialize an automatic variable and increment it in the function. Call this function thrice and print the value of the variable every time after incrementing.

➢Program

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

Output

```
1
1
1
```

➢Program 5.5.2 : Write a program to initialize a static variable and increment it in the function. Call this function thrice and print the value of the variable every time after incrementing.

➢Program

```
#include<stdio.h>
#include<conio.h>
void incr()
{
static int i=0;
i++;
printf("%d\n",i);
}
int main()
{
incr();
incr();
incr();
getch();
}
```

Output

```
1
2
3
```

4.External or Global storage class

- In this case data is stored in memory
- The initial value of such a variable is zero.
- The scope of the variable is global i.e. it is accessible from anywhere in the program.
- The life of such variables is till the program is alive.
- Let us see some program examples to understand the global variables.

➢Program 5.5.3 : Write a program to demonstrate the access of global variable.

➢Program

```
#include<stdio.h>
#include<conio.h>
int a=5;
void main()
{
int a=10;
printf("%d\n",a);
printf("%d\n",::a);
a=::a+a;
```

```
printf("%d\n",a);
printf("%d\n",::a);
::a=a;
printf("%d\n",a);
printf("%d\n",::a);
getch();
}
```

Output

```
10
5
15
5
15
15
```

➢Program 5.5.4 : Find the output of the following program.

```
#include<stdio.h>
#include<conio.h>
int x;
void f1()
{
++x;
}
void main()
{
int x = 10;
f1();
x = ::x + 10;
printf("%d %d\n",x,::x);
getch();
}
```

Output

```
11 1
```

➢Program 5.5.5 : Find the output of the following program.

```
#include<stdio.h>
#include<conio.h>
void f1()
```

```
{
extern int n3;
static int n1;
int n2 = 20;
n1 = n1 + 10;
n2 = n1 + n2;
n3 = n1 + n2;
printf("%d %d %d\n",n1,n2,n3);
}
int n3;
void main()
{
register int i;
for (i = 1; i <=3; i++) f1();
getch();
}
```

Output

```
10 30 40
20 40 60
30 50 80
```

CHAPTER

6

Arrays, String, Structure, Union, Pointers and Files

Syllabus Topic : Array - Concept, Declaration, Definition, Accessing Array Element for One-Dimensional Array

➢Program 6.1.1 :Write a program to accept 'n' integers from user into an array and display them one in each line.

➢Algorithm

Step I :START

Step II :PRINT "Enter number of elements".

Step III :INPUT n

Step IV:i = 0

Step V :IF i > n – 1 THEN GOTO step X

Step VI :PRINT "Enter a number"

Step VII:INPUT a[i]

Step VIII:i = i + 1

Step IX:GOTO step V

Step X:PRINT "The numbers entered are"

Step XI:i = 0

Step XII :IF i > n – 1 THEN GOTO step XVI

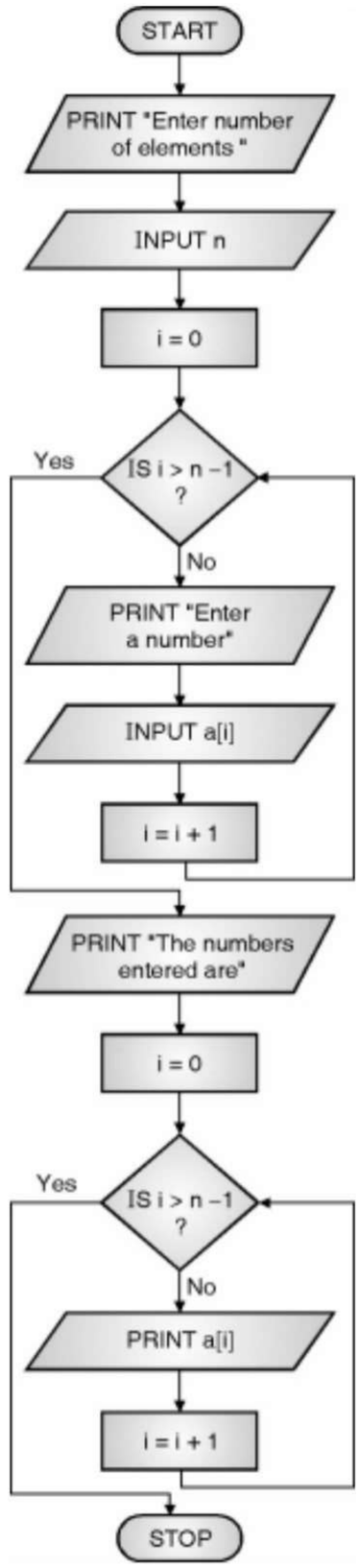
Step XIII :PRINT a[i]

Step XIV:i = i + 1

Step XV:GOTO step XII

Step XVI:STOP

➢Flowchart : (Refer Flowchart 1)



Flowchart 1

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int n,i,a[100];
clrscr();
   printf("Enter the number of elements:");
scanf("%d",&n);
for(i=0;i<=n–1;i++)
{
```

```
printf("Enter a value:");
scanf("%d",&a[i]);
}
printf("The numbers entered are\n");
for(i=0;i<=n-1;i++)
{
printf("%d\n",a[i]);
}
getch();
}
```

Output

```
Enter the number of elements:5
Enter a value:1
Enter a value:2
Enter a value:3
Enter a value:4
Enter a value:5
The numbers entered are
1
2
3
4
5
```

➢Program 6.1.2 :Write a program to accept 'n' integers from user into an array and display the average of these numbers.

➢Algorithm

Step I :START

Step II :PRINT "Enter number of elements".

Step III :INPUT n

Step IV:i = 0

Step V :IF i > n – 1 THEN GOTO step X

Step VI :PRINT "Enter a number"

Step VII:INPUT a[i]

Step VIII:i = i + 1

Step IX:GOTO step V

Step X:i = 0, sum = 0

Step XI :IF i > n – 1 THEN GOTO step XV

Step XII :sum = sum + a[i]

Step XIII:i = i + 1

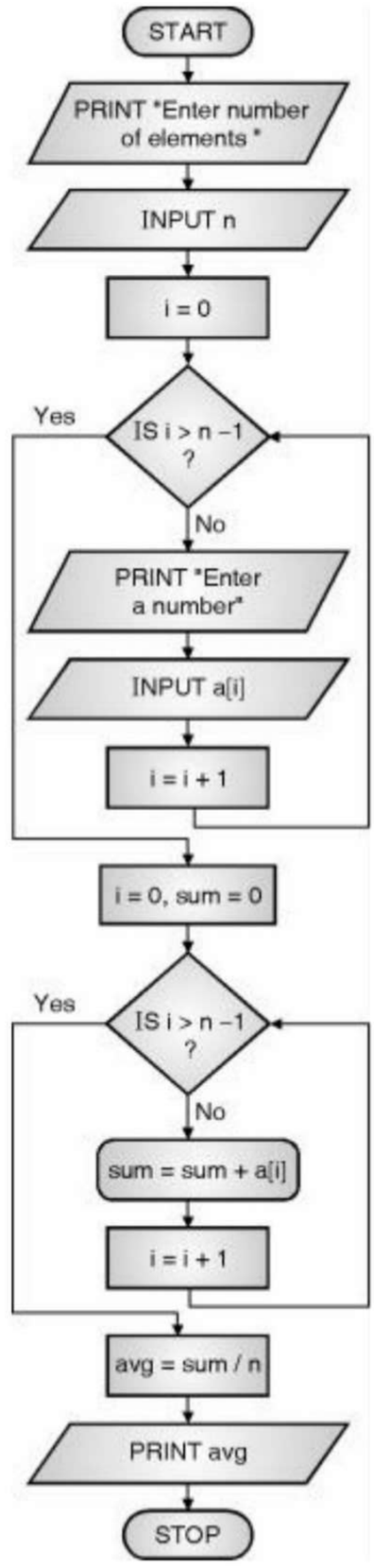
Step XIV:GOTO step XI

Step XV:avg = sum / n

Step XVI:PRINT avg

Step XVII:STOP

➢Flowchart : (Refer Flowchart 2)



Flowchart 2

➢Program

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

```
{
int n,i,a[100],sum=0;
  float avg;
clrscr();
printf("Enter the number of elements:");
scanf("%d",&n);
for(i=0;i<=n–1;i++)
{
printf("Enter a value:");
scanf("%d",&a[i]);
}
for(i=0;i<=n–1;i++)
{
sum=sum+a[i];
}
avg=sum;
avg=avg/n;
printf("The average of the numbers entered is %f",avg);
getch();
}
```

Output

```
Enter the number of elements:4
Enter a value:7
Enter a value:6
Enter a value:5
Enter a value:9
The average of the numbers entered is 6.75
```

➢Program 6.1.3 :Write a program to accept 'n' integers from user into an array and display the count of even and odd numbers of these.

➢Algorithm

Step I :START

Step II :PRINT "Enter number of elements".

Step III :INPUT n

Step IV:i = 0

Step V :IF i > n – 1 THEN GOTO step X

Step VI :PRINT "Enter a number"

Step VII:INPUT a[i]

Step VIII:i = i + 1

Step IX:GOTO step V

Step X:i = 0, even = 0

Step XI :IF i > n – 1 THEN GOTO step XV

Step XII :IF a[i] mod 2 = 0 THEN even = even + 1

Step XIII:i = i + 1

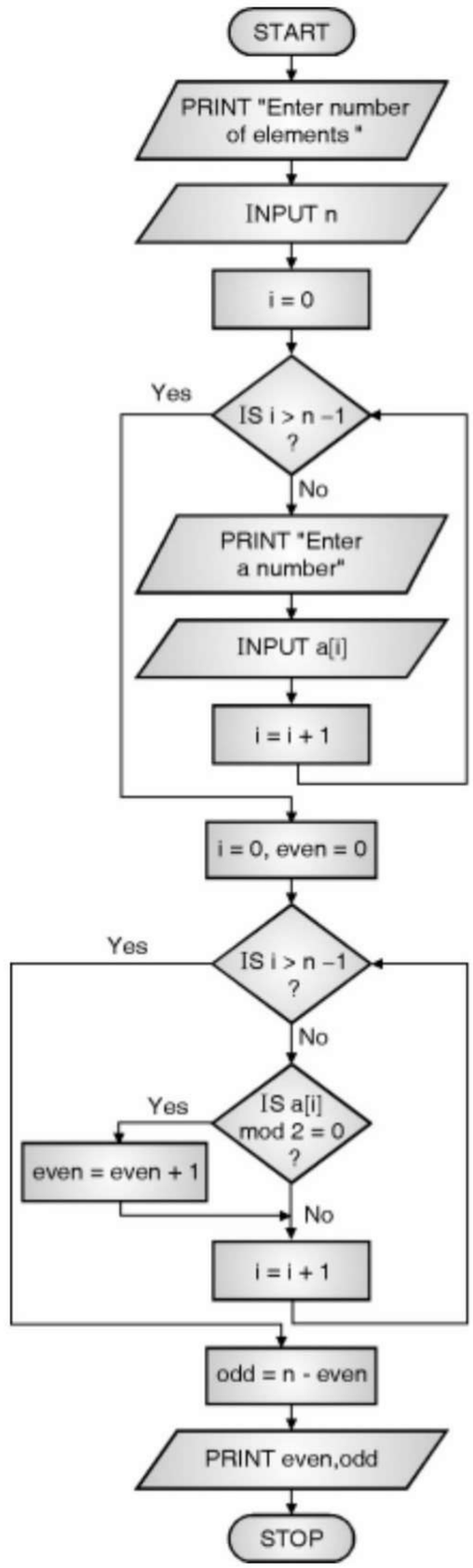
Step XIV:GOTO step XI

Step XV:odd = n – even

Step XVI:PRINT even, odd

Step XVII:STOP

➢Flowchart : (Refer Flowchart 3)



Flowchart 3

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int n,i,a[100],even=0;
clrscr();
   printf("Enter the number of elements:");
scanf("%d",&n);
for(i=0;i<=n–1;i++)
{
printf("Enter a value:");
scanf("%d",&a[i]);
}
for(i=0;i<=n–1;i++)
{
if (a[i]%2==0)
even++;
}
printf("The count of even numbers is %d and that of odd numbers is
%d",even,(n–even));
getch();
}
```

Output

```
Enter the number of elements:6
Enter a value:2
Enter a value:3
Enter a value:4
Enter a value:5
Enter a value:1
Enter a value:7
The count of even numbers is 2 and that of odd numbers is 4
```

➢Program 6.1.4 :Write a program to evaluate the value of the following series and display the result.

$$\sum_{i=1}^{n} x_i^2 - \left[\sum_{i=1}^{n} x_i\right]^2$$

➢Algorithm

Step I :START

Step II :PRINT "Enter value of n".

Step III :INPUT n

Step IV:i = 0

Step V :IF i > n – 1 THEN GOTO step X

Step VI :PRINT "Enter a value"

Step VII:INPUT a[i]

Step VIII:i = i + 1

Step IX:GOTO step V

Step X:i = 0, sum = 0, sum1 = 0

Step XI :IF i > n – 1 THEN GOTO step XV

Step XII :sum = sum + a[i], sum1 = sum1 + a[i] * a[i]

Step XIII:i = i + 1

Step XIV:GOTO step XI

Step XV:sum=sum 1 – (sum * sum)

Step XVI:PRINT sum

Step XVII:STOP

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int n,i,a[100],sum1=0,sum2=0,sum;
clrscr();
   printf("Enter the number of elements:");
scanf("%d",&n);
for(i=0;i<=n–1;i++)
{
printf("Enter a value:");
scanf("%d",&a[i]);
}
for(i=0;i<=n–1;i++)
{
sum1 = sum1 + a[i] * a[i];
sum2 = sum2 +a[i];
}
sum = sum1 – sum2 * sum2;
printf("The value of the series is %d",sum);
```

```
getch();
}
```

Output

```
Enter the number of elements:5
Enter a value:1
Enter a value:2
Enter a value:3
Enter a value:4
Enter a value:5
The value of the series is –170
```

➢Program 6.1.5 :Write a program to evaluate the value of the standard deviation (s.d.) and display the result.

$$\text{s.d.} = \sqrt{\frac{\sum_{i=i}^{n} (x_i - \bar{x})^2}{n}}$$

where, $\bar{x}$ is the average of all the numbers. *(May 2015)*

➢Algorithm

Step I :START

Step II :PRINT "Enter value of n".

Step III :INPUT n

Step IV:i = 0

Step V :IF i > n – 1 THEN GOTO step X

Step VI :PRINT "Enter a value"

Step VII:INPUT a[i]

Step VIII:i = i + 1

Step IX:GOTO step V

Step X:i = 0, sum = 0

Step XI :IF i > n – 1 THEN GOTO step XV

Step XII :sum = sum + a[i]

Step XIII:i = i + 1

Step XIV:GOTO step XI

Step XV:avg = sum / n

Step XVI:i = 0, sum = 0

Step XVII :IF i > n – 1 THEN GOTO step XXI

Step XVIII :sum = sum + (a[i] – avg) * (a[i] – avg)

Step XIX:i = i + 1

Step XX:GOTO step XVII

Step XXI:sum = sum / n

Step XXII:sum = square root of sum

Step XXIII:PRINT sum

Step XIV:STOP

➢Program

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
void main()
{
int n,i,a[100],sum=0;
   float avg,sum1=0,sd;
clrscr();
printf("Enter the number of elements:");
scanf("%d",&n);
for(i=0;i<=n-1;i++)
{
printf("Enter a value:");
scanf("%d",&a[i]);
sum=sum+a[i];
}
avg=sum;
avg=avg/n;
for(i=0;i<=n-1;i++)
{
sum1 = sum1 + pow(a[i]-avg,2);
}
sd = sum1/n;
sd = pow(sd,0.5);
printf("The standard deviation is %f",sd);
getch();
}
```

Output

```
Enter the number of elements:5
Enter a value:3
Enter a value:4
Enter a value:4
Enter a value:5
Enter a value:4
```

```
The standard deviation is 0.632456
```

➢Program 6.1.6 :Write a program to find the largest of 'n' numbers taken from user. ***(Dec. 2013)***

➢Algorithm

Step I :START

Step II :PRINT "Enter value of n".

Step III :INPUT n

Step IV:i = 0

Step V :IF i > n – 1 THEN GOTO step X

Step VI :PRINT "Enter a value"

Step VII:INPUT a[i]

Step VIII:i = i + 1

Step IX:GOTO step V

Step X:i = 1, large = a[0]

Step XI :IF i > n – 1 THEN GOTO step XV

Step XII :IF large < a[i], THEN large = a[i]

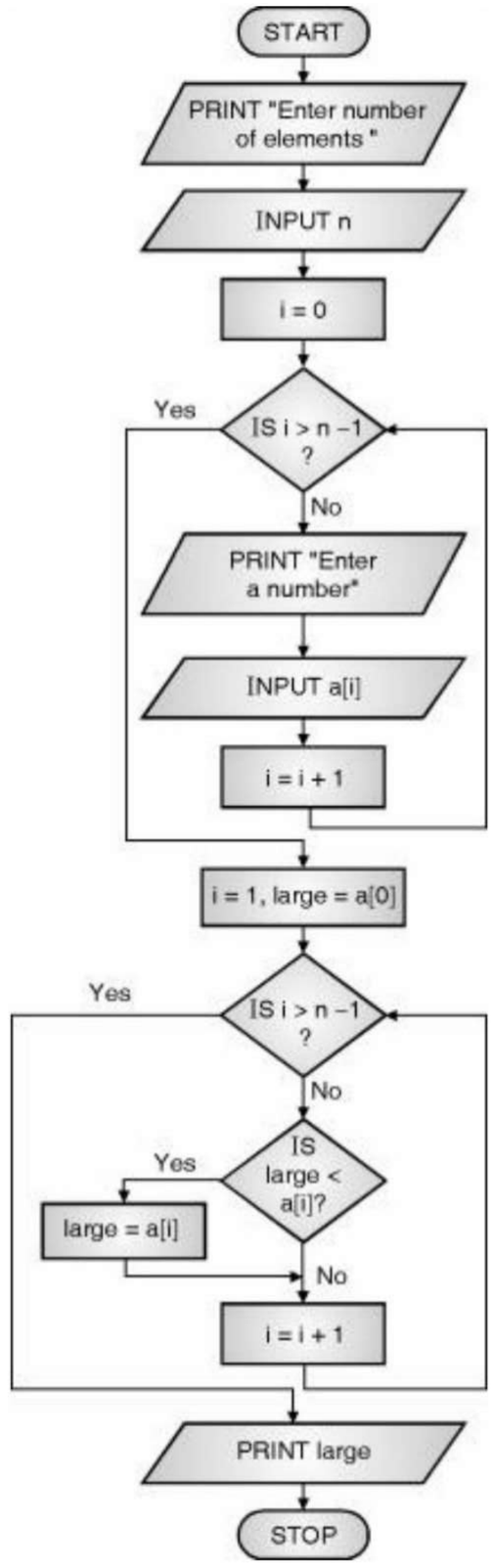
Step XIII:i = i + 1

Step XIV:GOTO step XI

Step XV:PRINT large

Step XVI:STOP

➢Flowchart : (Refer Flowchart 4)



Flowchart 4

➢Program

```
#include<stdio.h>
#include<conio.h>
void main( )
{
int n,i,a[100],large;
clrscr();
   printf("Enter the number of elements:");
scanf("%d",&n);
for(i=0;i<=n–1;i++)
{
printf("Enter a value:");
```

```
scanf("%d",&a[i]);
}
large=a[0];
for(i=1;i<=n-1;i++)
{
if(large<a[i])
large=a[i];
}
printf("The largest number is %d",large);
getch();
}
```

Output

```
Enter the number of elements:5
Enter a value:32
Enter a value:12
Enter a value:44
Enter a value:28
Enter a value:9
The largest number is 44
```

Syllabus Topic : Passing Arrays to Function

➢Program 6.1.7 :Write a program to find the largest of 'n' numbers taken from user, using a function. The array of numbers must be passed to the function. The largest number must be found in the function and returned to the main function.

➢Algorithm

main() function

Step I :START

Step II :PRINT "Enter value of n".

Step III :INPUT n

Step IV:i = 0

Step V :IF i > n – 1 THEN GOTO step X

Step VI :PRINT "Enter a value"

Step VII:INPUT a[i]

Step VIII:i = i + 1

Step IX:GOTO step V

Step X:large = CALL largest(arguments: a, n)

Step XI:PRINT large

Step XII:STOP

largest(parameters: a[], n)

Step I:i = 1, large = a[0]

Step II :IF i > n – 1 THEN GOTO step VI

Step III :IF large < a[i], THEN large = a[i]

Step IV:i = i + 1

Step V:GOTO step II

Step VI:RETURN (large)

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int largest (int a[], int n);
int n,i,a[100],large;
clrscr();
   printf("Enter the number of elements:");
scanf("%d",&n);
for(i=0;i<=n–1;i++)
{
printf("Enter a value:");
scanf("%d",&a[i]);
}
large=largest(a,n);
printf("The largest number is %d",large);
getch();
}
int largest (int a[], int n)
{
int i,large;
large=a[0];
for(i=1;i<=n–1;i++)
{
if(large<a[i])
large=a[i];
}
return large;
```

```
}
```

Output

```
Enter the number of elements:5
Enter a value:2
Enter a value:33
Enter a value:12
Enter a value:24
Enter a value:11
The largest number is 33
```

➢Program 6.1.8 :Write a program to find the smallest of 'n' numbers taken from user, using a function. The array of numbers must be passed to the array. The smallest number must be found in the function and returned to the main function.

(Dec. 2015)

➢Algorithm

main() function

Step I :START

Step II :PRINT "Enter value of n".

Step III :INPUT n

Step IV:i = 0

Step V :IF i > n – 1 THEN GOTO step X

Step VI :PRINT "Enter a value"

Step VII:INPUT a[i]

Step VIII:i = i + 1

Step IX:GOTO step V

Step X:small = CALL smallest(arguments: a, n)

Step XI:PRINT small

Step XII:STOP

smallest(parameters: a[], n)

Step I:i = 1, small = a[0]

Step II :IF i > n – 1 THEN GOTO step VI

Step III :IF small > a[i], THEN small = a[i]

Step IV:i = i + 1

Step V:GOTO step II

Step VI:RETURN (small)

➢Program

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

```
#include<conio.h>
void main()
{
int n,i,a[100],small;
int smallest (int a[], int n);
clrscr();
   printf("Enter the number of elements:");
scanf("%d",&n);
for(i=0;i<=n-1;i++)
{
printf("Enter a value:");
scanf("%d",&a[i]);
}
small=smallest(a,n);
printf("The smallest number is %d",small);
getch();
}
int smallest (int a[], int n)
{
int i,small;
small=a[0];
for(i=1;i<=n-1;i++)
{
if(small>a[i])
small=a[i];
}
return small;
}
```

Output

```
Enter the number of elements:5
Enter a value:12
Enter a value:32
Enter a value:7
Enter a value:123
Enter a value:76
The smallest number is 7
```

➢Program 6.1.9 :Write a program to find and display the reverse of an array using a function.

➢Algorithm

main() function

Step I :START

Step II :PRINT "Enter value of n".

Step III :INPUT n

Step IV:i = 0

Step V :IF i > n – 1 THEN GOTO step X

Step VI :PRINT "Enter a value"

Step VII:INPUT a[i]

Step VIII:i = i + 1

Step IX:GOTO step V

Step X:CALL reverse(arguments: a, n)

Step XI:STOP

reverse(parameters: a[], n)

Step I:i = 0

Step II :IF i > n – 1 THEN GOTO step VI

Step III :rev[n – 1 – i] = a[i]

Step IV:i = i + 1

Step V:GOTO step II

Step VI:PRINT "The reverse of array is"

Step VII:i = 0

Step VIII :IF i > n – 1 THEN GOTO step XII

Step IX :PRINT rev[i]

Step X:i = i + 1

Step XI:GOTO step VIII

Step XII:STOP

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int n,i,a[100];
void reverse (int a[], int n);
clrscr();
   printf("Enter the number of elements:");
scanf("%d",&n);
for(i=0;i<=n–1;i++)
{
```

```
printf("Enter a value:");
scanf("%d",&a[i]);
}
reverse(a,n);
getch();
}
void reverse (int a[], int n)
{
int i,rev[100];
for(i=0;i<=n–1;i++)
{
rev[n–i–1]=a[i];
}
printf("The reverse of this array is:\n");
for(i=0;i<=n–1;i++)
{
printf("%d\n",rev[i]);
}
}
```

Output

```
Enter the number of elements:5
Enter a value:1
Enter a value:23
Enter a value:43
Enter a value:12
Enter a value:22
The reverse of this array is:
22
12
43
23
1
```

➢Program 6.1.10 :Write a program to find and display the reverse of an array into the same array using a function.

➢Algorithm

main() function

Step I :START

Step II :PRINT "Enter value of n".

Step III :INPUT n

Step IV:i = 0

Step V :IF i > n – 1 THEN GOTO step X

Step VI :PRINT "Enter a value"

Step VII:INPUT a[i]

Step VIII:i = i + 1

Step IX:GOTO step V

Step X:CALL reverse(arguments: a, n)

Step XI:STOP

reverse(parameters: a[], n)

Step I:i = 0

Step II :IF i > (n – 1) / 2 THEN GOTO step VI

Step III :temp = a[n – 1 – i], a[n – 1 – i] = a[i], a[i] = temp

Step IV:i = i + 1

Step V:GOTO step II

Step VI:PRINT "The reverse of array is"

Step VII:i = 0

Step VIII :IF i > n – 1 THEN GOTO step XII

Step IX :PRINT a[i]

Step X:i = i + 1

Step XI:GOTO step VIII

Step XII:STOP

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int n,i,a[100];
void reverse (int a[], int n);
clrscr();
   printf("Enter the number of elements:");
scanf("%d",&n);
for(i=0;i<=n–1;i++)
{
printf("Enter a value:");
scanf("%d",&a[i]);
}
reverse(a,n);
```

```
getch();
}
void reverse (int a[], int n)
{
int i,temp;
for(i=0;i<=(n-1)/2;i++)
{
temp=a[n-i-1];
   a[n-i-1]=a[i];
   a[i]=temp;
}
printf("The reverse of this array is:\n");
for(i=0;i<=n-1;i++)
{
printf("%d\n",a[i]);
}
}
```

Output

```
Enter the number of elements:6
Enter a value:1
Enter a value:2
Enter a value:3
Enter a value:4
Enter a value:5
Enter a value:6
The reverse of this array is:
6
5
4
3
2
1
```

➢Program 6.1.11 :Write a program to find an element in an array and display the index of the element using a function. **OR**Write a program to implement sequential search algorithm.

➢Algorithm

main() function

Step I :START

Step II :PRINT "Enter value of n".

Step III :INPUT n

Step IV:i = 0

Step V :IF i > n – 1 THEN GOTO step X

Step VI :PRINT "Enter a value"

Step VII:INPUT a[i]

Step VIII:i = i + 1

Step IX:GOTO step V

Step X:PRINT "Enter the element to be searched"

Step XI:INPUT x

Step XII:index = CALL search(arguments: a, n, x)

Step XIII:IF (index = n) THEN PRINT "Not Found" ELSE PRINT index

Step XIV:STOP

search(parameters: a[], n, x)

Step I:i = 0

Step II :IF i > n – 1 THEN GOTO step VI

Step III :IF x = a[i] THEN GOTO step VI

Step IV:i = i + 1

Step V:GOTO step II

Step VI:RETURN (i)

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int n,i,a[100],x,index;
int search (int a[], int n, int x);
clrscr();
  printf("Enter the number of elements:");
scanf("%d",&n);
for(i=0;i<=n–1;i++)
{
printf("Enter a value:");
scanf("%d",&a[i]);
}
```

```
printf("Enter the element to be searched:");
scanf("%d",&x);
```

```
index=search(a,n,x);
if(index==n)
printf("Not Found");
else
printf("The element %d is found in the index %d",x,index);
getch();
}
int search (int a[], int n, int x)
{
int i;
for(i=0;i<=n-1;i++)
{
if(x==a[i])
break;
}
return i;
}
```

Output

```
Enter the number of elements:5
Enter a value:23
Enter a value:56
Enter a value:774
Enter a value:22
Enter a value:90
Enter the element to be searched:22
The element 22 is found in the index 3
```

➢Program 6.1.12 :Write a program to search a number within the array.***(Dec. 2015)***

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int n,i,a[100],x,index;
int search (int a[], int n, int x);
clrscr();
  printf("Enter the number of elements:");
```

```
scanf("%d",&n);
for(i=0;i<=n-1;i++)
{
printf("Enter a value:");
scanf("%d",&a[i]);
}
printf("Enter the element to be searched:");
scanf("%d",&x);
index=search(a,n,x);
if(index==n)
printf("Not Found");
else
printf("The element %d is found in the index %d",x,index);
getch();
}
int search (int a[], int n, int x)
{
int i;
for(i=0;i<=n-1;i++)
{
if(x==a[i])
break;
}
return i;
}
```

➢Program 6.1.13 :Write a program to sort numbers in ascending order. **OR** Write a program to implement bubble sorting algorithm for sorting numbers in ascending order. ***(May 2013, May 2014)***

➢Algorithm

main() function

Step I :START

Step II :PRINT "Enter value of n".

Step III :INPUT n

Step IV:i = 0

Step V :IF i > n – 1 THEN GOTO step X

Step VI :PRINT "Enter a value"

Step VII:INPUT a[i]

Step VIII:i = i + 1

Step IX:GOTO step V

Step X:CALL ascend(arguments: a, n)

Step XI:STOP

ascend(parameters: a[], n)

Step I:i = 0

Step II :IF i > n – 2 THEN GOTO step X

Step III :j = 0

Step IV:IF j > n – 2, THEN GOTO step VIII

Step V:IF a[j] > a[j + 1] THEN temp = a[j], a[j] = a[j +1], a[j + 1] = temp

Step VI:j = j + 1

Step VII:GOTO step IV

Step VIII:i = i + 1

Step IX:GOTO step II

Step X:PRINT "Sorted numbers"

Step XI:i =0

Step XII:IF i > n – 1 THEN GOTO step XVI

Step XIII:PRINT a[i]

Step XIV:i = i + 1

Step XV:GOTO step XII

Step XVI:STOP

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int n,i,a[100];
void ascend (int a[], int n);
clrscr();
   printf("Enter the number of elements:");
scanf("%d",&n);
for(i=0;i<=n–1;i++)
{
printf("Enter a value:");
scanf("%d",&a[i]);
}
ascend(a,n);
getch();
}
```

```
void ascend (int a[], int n)
{
int i,j,temp;
for(i=0;i<=n-2;i++)
{
for(j=0;j<=n-2;j++)
{
if(a[j]>a[j+1])
{
temp=a[j];
a[j]=a[j+1];
a[j+1]=temp;
}
}
}
printf("After sorting\n");
for(i=0;i<=n-1;i++)
{
printf("%d\n",a[i]);
}
}
```

Output

```
Enter the number of elements:5
Enter a value:4
Enter a value:6
Enter a value:1
Enter a value:85
Enter a value:9
After sorting
1
4
6
9
85
```

➢Program 6.1.14 :Write a program to sort numbers in descending order. OR Write a program to implement bubble sorting algorithm for sorting numbers in descending order. ***(May 2013, Dec. 2013)***

➢Algorithm

main() function

Step I :START

Step II :PRINT "Enter value of n".

Step III :INPUT n

Step IV:i = 0

Step V :IF i > n – 1 THEN GOTO step X

Step VI :PRINT "Enter a value"

Step VII:INPUT a[i]

Step VIII:i = i + 1

Step IX:GOTO step V

Step X:CALL descend(arguments: a, n)

Step XI:STOP

descend(parameters: a[], n)

Step I:i = 0

Step II :IF i > n – 2 THEN GOTO step X

Step III :j = 0

Step IV:IF j > n – 2, THEN GOTO step VIII

Step V:IF a[j] < a[j + 1] THEN temp = a[j], a[j] = a[j +1], a[j + 1] = temp

Step VI:j = j + 1

Step VII:GOTO step IV

Step VIII:i = i + 1

Step IX:GOTO step II

Step X:PRINT "Sorted numbers"

Step XI:i =0

Step XII:IF i > n – 1 THEN GOTO step XVI

Step XIII:PRINT a[i]

Step XIV:i = i + 1

Step XV:GOTO step XII

Step XVI:STOP

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int n,i,a[100];
void descend (int a[], int n);
clrscr();
```

```
   printf("Enter the number of elements:");
scanf("%d",&n);
for(i=0;i<=n-1;i++)
{
printf("Enter a value:");
scanf("%d",&a[i]);
}
descend(a,n);
getch();
}
void descend (int a[], int n)
{
int i,j,temp;
for(i=0;i<=n-2;i++)
{
for(j=0;j<=n-2;j++)
{
if(a[j]<a[j+1])
{
temp=a[j];
a[j]=a[j+1];
a[j+1]=temp;
}
}
}
printf("After sorting\n");
for(i=0;i<=n-1;i++)
{
printf("%d\n",a[i]);
}
}
```

Output

```
Enter the number of elements:5
Enter a value:1
Enter a value:3
Enter a value:5
Enter a value:4
```

```
Enter a value:2
After sorting
5
4
3
2
1
```

➢Program 6.1.15 :Write a program to sort float numbers in descending order.

[**Note :** Algorithm is same as program 6.1.14]

Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int n,i;
   float a[100];
void descend (float a[], int n);
clrscr();
printf("Enter the number of elements:");
scanf("%d",&n);
for(i=0;i<=n-1;i++)
{
printf("Enter a value:");
scanf("%f",&a[i]);
}
descend(a,n);
getch();
}
void descend (float a[], int n)
{
int i,j;
float temp;
for(i=0;i<=n-2;i++)
{
for(j=0;j<=n-2;j++)
{
if(a[j]<a[j+1])
```

```
{
temp=a[j];
a[j]=a[j+1];
a[j+1]=temp;
}
}
}
printf("After sorting\n");
for(i=0;i<=n-1;i++)
{
printf("%f\n",a[i]);
}
}
```

Output

```
Enter the number of elements:4
Enter a value:1.1
Enter a value:3.4
Enter a value:0.3
Enter a value:2.7
After sorting
3.4
2.7
1.1
0.3
```

➢Program 6.1.16 :Write a program in C to delete all the occurrences of given number from an ARRAY. For example, suppose ARRAY A = {2,5,3,9,2,2,3} and given number is 2 then after deletion. ARRAY should have A = {5,3,9,3}. ***(May 2014)***

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int n,i,a[100],x,index,j;
clrscr();
printf("Enter the number of elements:");
scanf("%d",&n);
for(i=0;i<=n-1;i++)
```

```
{
printf("Enter a value:");
scanf("%d",&a[i]);
}
printf("Enter the element to be deleted:");
scanf("%d",&x);
for(i=0;i<=n–1;i++)
{
if(x==a[i])
{
for(j=i;j<=n-2;j++)
{
a[j]=a[j+1];
}
n--;
}
}
printf("New array is:\n");
for(i=0;i<=n-1;i++)
{
printf("%d\n",a[i]);
}
getch();
}
```

Syllabus Topic : Concept, Declaration, Defination, Accessing Array Element for Multidimensional Arrays

➢Program 6.2.1 :Write a program to accept an m × n matrix and display it in natural form.

```
#include<stdio.h>
#include<conio.h>
void main()
{
int m,n,i,j,a[10][10];
clrscr();
   printf("Enter the number of rows and columns:");
scanf("%d %d",&m,&n);
for(i=0;i<=m–1;i++)
{
```

```
for(j=0;j<=n-1;j++)
{
printf("Enter a value:");
scanf("%d",&a[i][j]);
}
}
printf("The entered matrix is:\n");
for(i=0;i<=m-1;i++)
{
for(j=0;j<=n-1;j++)
{
printf("%d\t",a[i][j]);
}
printf("\n");
}
getch();
}
```

Output

```
Enter the number of rows and columns:3
3
Enter a value:1
Enter a value:2
Enter a value:3
Enter a value:4
Enter a value:5
Enter a value:6
Enter a value:7
Enter a value:8
Enter a value:9
The entered matrix is:
1	2	3
4	5	6
7	8	9
```

➢Program 6.2.2 :Write a program to calculate and display the average of all the elements of a m × n matrix.

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

```
{
int m,n,i,j,a[10][10],sum=0;
float avg;
clrscr();
   printf("Enter the number of rows and columns:");
scanf("%d %d",&m,&n);
for(i=0;i<=m-1;i++)
{
for(j=0;j<=n-1;j++)
{
printf("Enter a value:");
scanf("%d",&a[i][j]);
}
}
for(i=0;i<=m-1;i++)
{
for(j=0;j<=n-1;j++)
{
sum=sum+a[i][j];
}
}
avg=1.0 * sum/(m*n);
printf("The average is equal to %f",avg);
getch();
}
```

Output

```
Enter the number of rows and columns:3
2
Enter a value:1
Enter a value:2
Enter a value:3
Enter a value:4
Enter a value:5
Enter a value:6
The average is equal to 3
```

➢Program 6.2.3 :Write a program to find and display the largest of all the elements of a m × n matrix.

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

```
#include<conio.h>
void main()
{
int m,n,i,j,a[10][10],large;
clrscr();
   printf("Enter the number of rows and columns:");
scanf("%d %d",&m,&n);
for(i=0;i<=m-1;i++)
{
for(j=0;j<=n-1;j++)
{
printf("Enter a value:");
scanf("%d",&a[i][j]);
}
}
large=a[0][0];
for(i=0;i<=m-1;i++)
{
for(j=0;j<=n-1;j++)
{
if(large<a[i][j])
large=a[i][j];
}
}
printf("The largest element in the matrix is %d",large);
getch();
}
```

Output

```
Enter the number of rows and columns:3
2
Enter a value:1
Enter a value:2
Enter a value:65
Enter a value:33
Enter a value:3
Enter a value:34
The largest element in the matrix is 65
```

➢Program 6.2.4 : Write a program to find and display the sum of the diagonal elements of a square matrix.

```
#include<stdio.h>
#include<conio.h>
void main()
{
int m,n,i,j,a[10][10],sum=0;
clrscr();
printf("Enter the number of rows / columns:");
scanf("%d",&m);
n=m;
for(i=0;i<=m-1;i++)
{
for(j=0;j<=n-1;j++)
{
printf("Enter a value:");
scanf("%d",&a[i][j]);
}
}
for(i=0;i<=m-1;i++)
{
for(j=0;j<=n-1;j++)
{
if(i==j)
sum+=a[i][j];
}
}
printf("The sum of diagonal elements is %d",sum);
getch();
}
```

Output

```
Enter the number of rows / columns:3
Enter a value:1
Enter a value:2
Enter a value:3
Enter a value:4
Enter a value:5
Enter a value:6
```

```
Enter a value:7
Enter a value:8
Enter a value:9
The sum of diagonal elements is 15
```

➢Program 6.2.5 : Write a program to add two matrices of size m × n.

```
#include<stdio.h>
#include<conio.h>
void main()
{
int m,n,i,j,a[10][10],b[10][10],c[10][10];
clrscr();
printf("Enter the number of rows and columns:");
scanf("%d %d",&m,&n);
printf("Enter the elements of Matrix 1\n");
for(i=0;i<=m-1;i++)
{
for(j=0;j<=n-1;j++)
{
printf("Enter a value:");
scanf("%d",&a[i][j]);
}
}
printf("Enter the elements of Matrix 2\n");
for(i=0;i<=m-1;i++)
{
for(j=0;j<=n-1;j++)
{
printf("Enter a value:");
scanf("%d",&b[i][j]);
}
}
for(i=0;i<=m-1;i++)
{
for(j=0;j<=n-1;j++)
{
c[i][j]=a[i][j]+b[i][j];
}
```

```
}
printf("The sum of two matrices is:\n");
for(i=0;i<=m-1;i++)
{
for(j=0;j<=n-1;j++)
{
printf("%d\t",c[i][j]);
}
printf("\n");
}
getch();
}
```

Output

```
Enter the number of rows and columns:2
3
Enter the elements of Matrix 1
Enter a value:1
Enter a value:2
Enter a value:3
Enter a value:4
Enter a value:5
Enter a value:6
Enter the elements of Matrix 2
Enter a value:1
Enter a value:2
Enter a value:3
Enter a value:4
Enter a value:5
Enter a value:6
The sum of two matrices is:
2     4     6
8     10    12
```

➢Program 6.2.6 :Write a program to find the transpose of a matrix of size m × n. Use a function transpose to do this operation.***(May 2013)***

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

```
{
int m,n,i,j,a[10][10];
void transpose (int a[10][10],int m, int n);
clrscr();
  printf("Enter the number of rows and columns:");
scanf("%d %d",&m,&n);
for(i=0;i<=m-1;i++)
{
for(j=0;j<=n-1;j++)
{
printf("Enter a value:");
scanf("%d",&a[i][j]);
}
}
printf("The original Matrix is:\n");
for(i=0;i<=m-1;i++)
{
for(j=0;j<=n-1;j++)
{
printf("%d \t",a[i][j]);
}
printf("\n");
}
transpose(a,m,n);
getch();
}
void transpose (int a[10][10], int m, int n)
{
int b[10][10],i,j;
for(i=0;i<=m-1;i++)
{
for(j=0;j<=n-1;j++)
{
b[j][i]=a[i][j];
}
}
printf("The transpose of this matrix is:\n");
for(i=0;i<=n-1;i++)
```

```
{
for(j=0;j<=m-1;j++)
{
printf("%d\t",b[i][j]);
}
printf("\n");
}
}
```

Output

```
Enter the number of rows and columns:3
2
Enter a value:1
Enter a value:2
Enter a value:3
Enter a value:4
Enter a value:5
Enter a value:6
The original Matrix is:
1	2
3	4
5	6
The transpose of this matrix is:
1	3	5
2	4	6
```

➢Program 6.2.7 :Write a program to find the transpose of a square matrix. Use a function transpose to do this operation.***(Dec. 2013, May 2014, Dec. 2015)***

```
#include<stdio.h>
#include<conio.h>
void main()
{
int m,n,i,j,a[10][10];
void transpose (int a[10][10],int m, int n);
clrscr();
  printf("Enter the number of rows / columns:");
scanf("%d",&m);
n=m;
for(i=0;i<=m-1;i++)
```

```
{
for(j=0;j<=n-1;j++)
{
printf("Enter a value:");
scanf("%d",&a[i][j]);
}
}
printf("The original Matrix is:\n");
for(i=0;i<=m-1;i++)
{
for(j=0;j<=n-1;j++)
{
printf("%d\t",a[i][j]);
}
printf("\n");
}
transpose(a,m,n);
getch();
}
void transpose (int a[10][10], int m, int n)
{
int i,j,temp;
for(i=0;i<=m-1;i++)
{
for(j=i;j<=n-1;j++)
{
    temp=a[j][i];
    a[j][i]=a[i][j];
    a[i][j]=temp;
}
}
printf("The transpose of this matrix is:\n");
for(i=0;i<=m-1;i++)
{
for(j=0;j<=n-1;j++)
{
printf("%d\t",a[i][j]);
}
```

```
printf("\n");
}
}
```

Output

```
Enter the number of rows / columns:3
Enter a value:1
Enter a value:2
Enter a value:3
Enter a value:4
Enter a value:5
Enter a value:6
Enter a value:7
Enter a value:8
Enter a value:9
The original Matrix is:
1    2    3
4    5    6
7    8    9
The transpose of this matrix is:
1    4    7
2    5    8
3    6    9
```

➢Program 6.2.8 :Write a program to multiply two matrices using a function. ***(May 2013, May 2014)***

```
#include<stdio.h>
#include<conio.h>
void main()
{
int m,n,p,i,j,a[10][10],b[10][10];
void MatMul (int a[10][10],int b[10][10], int m, int n, int p);
clrscr();
   printf("Enter the number of rows and columns of matrix 1:");
scanf("%d %d",&m,&n);
printf("Enter the values of matrix 1\n");
for(i=0;i<=m-1;i++)
{
for(j=0;j<=n-1;j++)
{
```

```
printf("Enter a value:");
scanf("%d",&a[i][j]);
}
}
printf("The number of rows for matrix 2 will be %d\n",n);
printf("Enter the columns of matrix 2:");
scanf("%d",&p);
printf("Enter the elements of matrix 2:\n");
for(i=0;i<=n-1;i++)
{
for(j=0;j<=p-1;j++)
{
printf("Enter a value:");
scanf("%d",&b[i][j]);
}
}
MatMul(a,b,m,n,p);
getch();
}
void MatMul (int a[10][10], int b[10][10], int m, int n, int p)
{
int i,j,k,c[10][10];
for(i=0;i<=m-1;i++)
{
for(j=0;j<=p-1;j++)
{
    c[i][j]=0;
    for(k=0;k<=n-1;k++)
    {
c[i][j]=c[i][j]+a[i][k]*b[k][j];
    }
}
}
printf("The resultant matrix is:\n");
for(i=0;i<=m-1;i++)
{
for(j=0;j<=p-1;j++)
{
```

```
printf("%d\t",c[i][j]);
}
printf("\n");
}
}
```

Output

```
Enter the number of rows and columns of matrix 1:3
2
Enter the values of matrix 1
Enter a value:1
Enter a value:2
Enter a value:3
Enter a value:4
Enter a value:5
Enter a value:6
The number of rows for matrix 2 will be 2
Enter the columns of matrix 2:3
Enter the elements of matrix 2:
Enter a value:1
Enter a value:2
Enter a value:3
Enter a value:4
Enter a value:5
Enter a value:6
The resultant matrix is:
9     12     15
19     26     33
29     40     51
```

Syllabus Topic : Basic of String

➢Program 6.3.1 : Write a program to accept and display a string.

```
#include<stdio.h>
#include<conio.h>
void main()
{
char a[100];
clrscr();
```

```
  printf("Enter a string\n");
gets(a);
printf("The entered string is: %s",a);
getch();
}
```

Output

```
Enter a string
How are you?
The entered string is: How are you?
```

Syllabus Topic : Functions in String.h

Q.Explain any three string standard library functions. ***(May 2013)***

ORState library functions in string.h along with their uses. ***(May 2014, May 2015, Dec. 2015)***

Ans. :

This header file has many functions that perform certain operations on strings.

1.strlen() function

- This function returns an integer value that is the length of the string passed to the function.
- When returning the length of the string it does not consider the space required for null character. Hence it returns the exact length of the string neglecting the space required for null character.

➢Program 6.4.1 : Write a program to accept a string and display its length.

```
#include<conio.h>
#include<stdio.h>
#include<string.h>
void main()
{
int l;
char a[100];
clrscr();
printf("Enter a string\n");
gets(a);
l=strlen(a);
printf("The length of the entered string is: %d",l);
getch();
}
```

Output

```
Enter a string
```

```
Hello
The length of the entered string is: 5
```

2.strcpy() function

- This function copies the second string into the first string passed to the function.
- The second string remains unchanged. Only the first string is changed and gets a copy of the second string. For e.g. strcpy(str1,str2), will copy the string "str2" into the string "str1".

➢Program 6.4.2 :Write a program to accept a string, copy it into another string and display this new string.

```
#include<conio.h>
#include<stdio.h>
#include<string.h>
void main()
{
char a[100],b[100];
clrscr();
printf("Enter a string\n");
gets(a);
strcpy(b,a);
printf("The new string is %s",b);
getch();
}
```

Output

```
Enter a string
Hello, how are you?
The new string is Hello, how are you?
```

3.strcmp() function

- This function compares the two string variables passed to it. It returns an integer value equal to
  - 0 (zero), if the two strings are equal.
  - Negative value, if the first string is smaller than the second string.
  - Positive value, if the first string is greater than the second string.
- Both the strings remain unchanged.
- The string is smaller means its alphabetical sequence is smaller. For example, "Hello" is lesser than "Hi"; because the first character "H" is same, but the second character "e" is smaller than "i". "e" is smaller than "i" because "e" comes before "i" in the alphabets i.e. A, B, C, ...Z. The function compares the ASCII value of the characters.

➢Program 6.4.3 :Write a program to accept two strings, compare them and display if they are equal or not. If they are not equal display the one which is greater.

```
#include<conio.h>
#include<stdio.h>
#include<string.h>
void main()
{
char a[100],b[100];
clrscr();
  printf("Enter two strings:\n");
gets(a);
gets(b);
if(strcmp(a,b)==0)
printf("The strings are equal ");
else if(strcmp(a,b)>0)
printf("%s string is greater",a);
else
printf("%s string is greater",b);
getch();
}
```

Output

```
Enter two strings:
Hello
Hi
Hi string is greater
```

4.strcat() function

- This function concatenates (joins) the two string variables passed to it. It returns a string of te combination of the two in the first string variable

➢Program 6.4.4 :Write a program to accept two strings, join them and display theresult.

```
#include<conio.h>
#include<stdio.h>
#include<string.h>
void main()
{
char a[100],b[100];
clrscr();
printf("Enter two strings:\n");
gets(a);
```

```
gets(b);
strcat(a,b);
printf("The concatenated string is %s",a);
getch();
}
```

Output

```
Enter two strings:
Harish
Narula
The concatenated string is HarishNarula
```

Syllabus Topic : Array of String

➢Program 6.4.5 :Write a C program to accept a month number and display the month name(Use 2 Dimensional char array).

```
#include<stdio.h>
#include<conio.h>
void main()
{
char
a[12][10]={"January","February","March","April","May","June","July","August",  "September","October","
November", "December"};
int m;
clrscr();
printf("Enter the month number:");
scanf("%d",&m);
printf("The month name is %s",a[m-1]);
getch();
}
```

Output

```
Enter the month number:6
The month name is June
```

➢Program 6.5.1 :Write a program to accept a string and find its length without using the string header file.

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

```
char a[100];
   int len=0;
clrscr();
printf("Enter a string:\n");
gets(a);
while(a[len]!='\0')
{
len++;
}
printf("The length of this string is %d characters.",len);
getch();
}
```

Output

```
Enter a string:
Hello
The length of this string is 5 characters.
```

➢Program 6.5.2 :Write a program to accept a string and find the number of vowels in it (Do not use the string header file).

```
#include<conio.h>
#include<stdio.h>
void main()
{
char a[100];
   int i,len=0,count=0;
clrscr();
printf("Enter a string:\n");
gets(a);
while(a[len]!=0)
{
len++;
}
for(i=0;i<=len-1;i++)
{
if(a[i]=='a' ||a[i]=='e' || a[i]=='i' || a[i]=='o' || a[i]=='u' || a[i]=='A' || a[i]=='E' || a[i]=='I' || a[i]=='O' || a[i]=='U')
count++;
}
printf("The total number of vowels are: %d",count);
```

```
getch();
}
```

Output

```
Enter a string:
Hello
The total number of vowels are: 2
```

➢Program 6.5.3 :Write a program to count vowel using switch case.***(May 2015)***

```
#include<stdio.h>
#include< conio.h>
Void main ( )
{
char a [100];
int i, len = 0, count = 0;
clrscr ( );
printf ("Enter a string");
gets (a);
while (a [count]! = '\0')
{
Count ++ ;
}
for (i = 0; i < = count – 1; i ++)
{
switch (a [i] )
{
case 'a'; len ++;
break;
case 'e' =len ++;
break;
case 'i' =len ++;
break;
case 'o' =len ++;
break;
case 'u' =len ++;
break;
case 'A' =len ++;
break;
case 'E' =len ++;
```

```
break;
case 'I' =len ++;
break;
case 'O' =len ++;
break;
case 'u' = len ++;
break;
}
}
printf("No of values = %d\n", len);
getch ( );
}
```

➢Program 6.5.4 :Write a program to reverse a user entered string.

```
#include<conio.h>
#include<stdio.h>
#include<string.h>
void main()
{
int l;
char a[100];
clrscr();
printf("Enter a string\n");
gets(a);
strrev(a);
printf("The reversed string is: %s",a);
getch();
}
```

Output

```
Enter a string
Hello
The reversed string is: olleH
```

➢Program 6.5.5 :Write a program to reverse a user entered string (Do not use the string header file). Write a function to do this work of reversing and displaying the string.

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

```
{
int n=0;
char a[100];
void reverse (char a[100], int n);
clrscr();
printf("Enter a string:");
gets(a);
while(a[n]!='\0')
{
n++;
}
reverse(a,n);
getch();
}
void reverse (char a[100], int n)
{
int i;
char temp;
for(i=0;i<=(n–1)/2;i++)
{
temp=a[n–i–1];
a[n–i–1]=a[i];
a[i]=temp;
}
printf("The reverse of this string is: %s",a);
}
```

Output

```
Enter a string:Hello
The reverse of this string is: olleH
```

➢Program 6.5.6 :Write a program to reverse a user entered string (Do not use the string header file). Reverse the string in the same function i.e. the main function.

```
#include<stdio.h>
#include<conio.h>
void main()
{
int n=0,i;
char a[100],temp;
```

```
clrscr();
printf("Enter a string:");
gets(a);
while(a[n]!='\0')
{
n++;
}
for(i=0;i<=(n-1)/2;i++)
{
temp=a[n-i-1];
a[n-i-1]=a[i];
a[i]=temp;
}
printf("The reverse of this string is: %s",a);
getch();
}
```

Output

```
Enter a string:Hello
The reverse of this string is: olleH
```

➢Program 6.5.7 :Write a program to check whether the entered string is palindrome or not (Do not use the string header file).***(May 2013, Dec. 2013, Dec. 2014)***

Note : Palindrome is a string which is same when read from either of the sides. For e.g. nitin, madam, malayalam etc.

```
#include<stdio.h>
#include<conio.h>
void main()
{
int n=0,i;
char a[100],rev[100];
clrscr();
printf("Enter a string:");
gets(a);
while(a[n]!='\0')
{
n++;
}
for(i=0;i<=(n-1);i++)
```

```
{
rev[n-i-1]=a[i];
}
for(i=0;i<=n-1;i++)
{
if(a[i]!=rev[i])
break;
}
if(i==n)
printf("The string is palindrome.");
else
printf("The string is not palindrome.");
getch();
}
```

Output

```
Enter a string:nitin
The string is palindrome.
```

➢Program 6.5.8 :Write a program to reverse a sentence (do not reverse the words).

```
#include<conio.h>
#include<stdio.h>
void main()
{
int n=0,i,j,k;
char a[100],rev[100];
clrscr();
printf("Enter a string:");
gets(a);
while(a[n]!='\0')
{
n++;
}
for(i=0,j=0;i<=n;i++)
{
if(a[i]==' '|| a[i]=='\0')
{
for(k=0;j<=i;k++,j++)
{
```

```
rev[n–i+k]=a[j];
}
}
}
rev[k–1]=' ';
rev[n]='\0';
printf("The reversed sentence is: %s",rev);
getch();
}
```

Output

```
Enter a string:This is a bench
The reversed sentence is: bench a is This
```

➢Program 6.6.1 :Write a program to shift the elements of a single dimensional array in the right direction by one position. If the given array is [76 35 43 22] then after execution of the program it should be [22 76 35 43].

```
#include<conio.h>
#include<stdio.h>
void main()
{
int n,i,a[100],temp;
clrscr();
printf("Enter the number of elements:");
scanf("%d",&n);
for(i=0;i<=n–1;i++)
{
printf("Enter a value");
scanf("%d",&a[i]);
}
temp=a[n–1];
for(i=n–1;i>0;i––)
{
a[i]=a[i–1];
}
a[0]=temp;
for(i=0;i<=n–1;i++)
{
printf("%d\n",a[i]);
```

```
}
getch();
}
```

Output

```
Enter the number of elements:5
Enter a value1
Enter a value2
Enter a value3
Enter a value4
Enter a value5
5
1
2
3
4
```

➢Program 6.6.2 :Write a function named SUMFUN(),with argument x and N which return the sum of the following series $x - x^3 / 3! + x^5 / 5! - x^7 / 7! + x^9 / 9! + \ldots\ldots$

Write main program to access this function.

```
#include<stdio.h>
#include<math.h>
#include<conio.h>
void main()
{
int i,fact=1,N,sign=-1;
float x,numerator,sum,term;
float SUMFUN(float x, int N);
clrscr();
printf("Enter the value of x and N: ");
scanf("%f %d",&x,&N);
sum=SUMFUN(x,N);
printf("The value of the series is %f",sum);
getch();
}
float SUMFUN(float x, int N)
{
int i,fact=1,sign=-1;
float numerator,sum,term;
```

```
term=x;
sum=term;
for(i=3;i<=N;i=i+2)
{
fact=fact*i*(i-1);
numerator=pow(x,i);
term=numerator/fact;
sum=sum + sign * term;
sign=sign*-1;
}
return(sum);
}
```

Output

```
Enter the value of x and N: 2
5
The value of the series is 0.933333
```

➢Program 6.6.3 :Write a C program to find the sum of column elements of a 2 dimensional M × N array A.

```
#include<stdio.h>
#include<conio.h>
void main()
{
int m,n,i,j,a[10][10],b[10];
clrscr();
printf("Enter the number of rows and columns of matrix:");
scanf("%d %d",&m,&n);
printf("Enter the values of matrix\n");
for(i=0;i<=m-1;i++)
{
for(j=0;j<=n-1;j++)
{
printf("Enter a value:");
scanf("%d",&a[i][j]);
}
}
for(i=0;i<=n-1;i++)
{
b[i]=0;
```

```
for(j=0;j<=m-1;j++)
{
b[i]=b[i]+a[j][i];
}
}
printf("The entered matrix is:\n");
for(i=0;i<=m-1;i++)
{
for(j=0;j<=n-1;j++)
{
printf("%d\t",a[i][j]);
}
printf("\n");
}
printf("The sum of the columns are:\n");
for(i=0;i<=n-1;i++)
{
printf("%d\t",b[i]);
}
getch();
}
```

Output

```
Enter the number of rows and columns of matrix:3
3
Enter the values of matrix
Enter a value:1
Enter a value:2
Enter a value:3
Enter a value:4
Enter a value:5
Enter a value:6
Enter a value:7
Enter a value:8
Enter a value:9
The entered matrix is:
1	2	3
4	5	6
```

```
7     8     9
The sum of the columns are:
12    15    18
```

➢Program 6.6.4 :Write a program to count blank spaces, digits, vowels and consonants in the string.

```
#include<conio.h>
#include<stdio.h>
void main()
{
char a[100];
   int i,len=0,vowels=0,spaces=0,digits=0,consonants=0;
clrscr();
printf("Enter a string:\n");
gets(a);
while(a[len]!=0)
{
len++;
}
for(i=0;i<=len-1;i++)
{
if(a[i]=='a' ||a[i]=='e' || a[i]=='i' || a[i]=='o' || a[i]=='u' || a[i]=='A' || a[i]=='E' || a[i]=='I' || a[i]=='O' || a[i]=='U')
vowels++;
else
{
if((a[i]>='a' && a[i]<='z') || (a[i]>='A' && a[i]<='Z'))
consonants++;
else
{
if(a[i]>='0' &&a[i]<='9')
digits++;
else
{
if(a[i]==' ')
spaces++;
}
}
}
}
```

```
printf("The total number of vowels are: %d\nThe total number of spaces are:%d\nThe total number of digits are: %d\nThe total number of consonants are: %d", vowels, spaces, digits, consonants);
getch();
}
```

Output

```
Enter a string:
There are 20 benches
The total number of vowels are: 6
The total number of spaces are 3
The total number of digits are 2
The total number of consonants are 9
```

➢Program 6.6.5 :Develop your own functions for performing following operations on strings :

(i)Copying one string to another.

(ii)Adding one string to the end of another. ***(Dec. 2014)***

```
#include<stdio.h>
#include<conio.h>
void main()
{
int n=0;
char a[100],b[100];
void copy (char a[100], char *p, int n);
void join (char a[100], char *p, int n, int m);
clrscr();
printf("Enter a string:");
gets(a);
while(a[n]!='\0')
{
n++;
}
copy(a,&b[0],n);
printf("New string after copy is %s\n",b);
join(a,&b[0],n,n);
printf("New string after concatenation is %s\n",b);
getch();
}
void copy (char a[100], char *p, int n)
{
```

```
int i;
for(i=0;i<=n;i++)
{
*(p+i)=a[i];
}
}
void join (char a[100], char *p, int n, int m)
{
int i;
for(i=n;i<=m+n;i++)
{
*(p+i)=a[i-n];
}
}
```

Output

```
Enter a string:HGN
New string after copy is HGN
New string after concatenation is HGNHGN
```

➢Program 6.6.6 :Write a program to calculate sum of list by passing array to function. ***(May 2013)***

```
#include<stdio.h>
#include<conio.h>
void main()
{
int addition (int a[], int n);
int n,i,a[100],sum;
clrscr();
   printf("Enter the number of elements:");
scanf("%d",&n);
for(i=0;i<=n-1;i++)
{
printf("Enter a value:");
scanf("%d",&a[i]);
}
sum=addition(a,n);
printf("The sum is %d",sum);
getch();
}
```

```
int addition (int a[], int n)
{
int i,sum =0;
for(i=1;i<=n-1;i++)
{
sum = sum + a[i];
}
return sum;
}
```

Output

```
Enter the number of elements:5
Enter a value:32
Enter a value:12
Enter a value:44
Enter a value:28
Enter a value:9
The largest number is 125
```

➢Program 6.6.7 :Write a program in C to cyclically rotate the elements in array. Program should accept a choice in which direction to rotate i.e. left or right. Depending on choice it shouldperform cyclic rotation. Suppose array A contains elements {1, 2, 3, 4, 5} then if choice is rotate right o/p should be {5, 1, 2, 3, 4} and if choice is rotate left then o/p should be {2, 3, 4, 5, 1}. ***(Dec. 2013)***

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int n,i,a[100], new[100], choice, temp;
clrscr();
   printf("Enter the number of elements:");
scanf("%d",&n);
for(i=0;i<=n-1;i++)
{
printf("Enter a value:");
scanf("%d",&a[i]);
}
printf("1. Rotate right\n2. Rotate left\nEnter your choice:");
```

```
scanf("%d",&choice);
switch(choice)
{
case 1: temp = a[n-1];
for(i=n-1;i>=1;i--)
{
a[i] = a[i-1];
}
a[0] = temp;
printf("The new array is:\n");
for(i=0;i<=n-1;i++)
{
printf("%d\n",a[i]);
}
case 2: temp = a[0];
for(i=0;i<=n-2;i++)
{
a[i] = a[i+1];
}
a[n-1] = temp;
printf("The new array is:\n");
for(i=0;i<=n-1;i++)
{
printf("%d\n",a[i]);
}
default : printf("Invalid Choice");
}
getch();
}
```

➢Program 6.6.8 :Write a program in C to read and display elements of a square (2D) matrix and check whether the entered matrix is symmetric or not. ***(Dec. 2014, May 2015)***

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int m,n,i,j,a[10][10];
```

```
void transpose (int a[10][10],int m, int n);
clrscr();
  printf("Enter the number of rows and columns:");
scanf("%d %d",&m,&n);
for(i=0;i<=m-1;i++)
{
for(j=0;j<=n-1;j++)
{
printf("Enter a value:");
scanf("%d",&a[i][j]);
}
}
printf("The original Matrix is:\n");
for(i=0;i<=m-1;i++)
{
for(j=0;j<=n-1;j++)
{
printf("%d \t",a[i][j]);
}
printf("\n");
}
transpose(a,m,n);
getch();
}
void transpose (int a[10][10], int m, int n)
{
int b[10][10],i,j,flag=0;
for(i=0;i<=m-1;i++)
{
for(j=0;j<=n-1;j++)
{
b[j][i]=a[i][j];
}
}
printf("The transpose of this matrix is:\n");
for(i=0;i<=n-1;i++)
{
for(j=0;j<=m-1;j++)
```

```
{
printf("%d\t",b[i][j]);
}
printf("\n");
}
for(i=0;i<=n-1;i++)
{
for(j=0;j<=m-1;j++)
{
if(a[i][j]!=b[i][j]) flag++;
}
}
if(flag==0) printf("Symmetric Matrix");
else printf("Not a symmetric matrix");
}
```

➢Program 6.6.9 :Four experiments are performed, each experiment consisting of six test results, The result for each experiment follows. WAP to compute and display the average of the test results for each experiment.*(Dec. 2014)*

1st Experiment results	**23.2**	**31.5**	**16.9**	**28.0**	**26.3**	**28.2**
2nd Experiment results	**34.8**	**45.2**	**20.8**	**39.4**	**33.4**	**36.8**
3rd Experiment results	**19.4**	**50.6**	**45.1**	**20.8**	**50.6**	**13.4**
4th Experiments results	**36.9**	**42.7**	**20.8**	**10.2**	**16.8**	**42.7**

➢Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int i,j,a[100],sum;
  float avg;
clrscr();
for(j=1;j<=4;j++)
{
sum=0;
for(i=0;i<=5;i++)
{
printf("Enter a value:");
scanf("%d",&a[i]);
sum=sum+a[i];
```

```
}
avg=sum;
avg=avg/n;
printf("The average of the readings of experiment %d is %f",j,avg);
}
getch();
}
```

➢Program 6.6.10 :Write a program in C to accept an ARRAY A with n elements and Separate it into two different arrays B and C in such a way that B contains Odd numbers and C contains Even numbers. i.e, if ARRAY A contains A={3,2,4,2,5,7,8} then B={3,5,7} and C={2,4,2,8}. ***(May 2016)***

```
#include <stdio.h>
void main()
{
int A[10], B[10], C[10];
   int i,=0,=0, n;
   printf("Enter the size of array A:\n");
   scanf("%d", &n);
   printf("Enter the elements of the array \n");
for (=0; i < n; i++)
{
   scanf("%d", &A[i]);
}
for (=0; i < n; i++)
{
   if (A[i] % 2 == 0)
   {
      C[j]=[i];
      j++;
   }
   else
   {
      B[k]=[i];
      k++;
   }
}
printf("The elements of odd array are \n");
for (=0; i < j; i++)
```

```
  {
     printf("%d\n", B[i]);
  }
  printf("The elements of Even array are \n");
  for (=0; i < k; i++)
  {
     printf("%d\n", C[i]);
  }
getch();
}
```

➢Program 6.6.11 :Write a program to generate Pascal Triangle upto n rows.***(May 2016)***

```
#include<stdio.h>
longfact(int);
intmain(){
intline,i,j;
printf("Enter the no. of lines: ");
scanf("%d",&line);
for(i=0;i<line;i++){
for(j=0;j<line-i-1;j++)
printf(" ");
for(j=0;j<=i;j++)
printf("%ld ",fact(i)/(fact(j)*fact(i-j)));
printf("\n");
}
return0;
}
longfact(intnum){
longf=1;
inti=1;
while(i<=num){
f=f*i;
i++;
}
returnf;
}
```

Output

```
Enter the no. of lines: 8
```

```
1
1 1
1 2 1
1 3 3 1
1 4 6 4 1
1 5 10 10 5 1
1 6 15 20 15 6 1
1 7 21 35 35 21 7 1
```

➢Program 6.6.12 :Write a program to perform matrix multiplication using user defined functions. Assume first matrix is of size M × N, second matrix of size N × P and third matrix (Result matrix) is of size M × P.

Program should include following user defined functions.

(i)read_matrix

(ii)Display_matrix

(iii)Multiply_matrix ***(May 2016)***

```
#include<stdio.h>
#include<conio.h>
void read_matrix(int a[10][10],int row,int col)
{
int i,j;
for(i=1;i<=row;i++)
{
for(j=1;j<=col;j++) { printf("Enter Element %d %d : ",i,j);
   scanf("%d",&a[i][j]);
      }
}
}
void multiply_matrix(int m1[10][10],int m2[10][10],int m3[10][10],int row,int col)
{
int i,j,k;
for(i=1;i<=row;i++)
{
for(j=1;j<=col;j++)
{
for (k=1;k<=row;k++)
{
m3[i][j] = m3[i][j] + (m1[i][k] * m2[k][j]);
}
```

```
}
}
}
void display_matrix(int m[10][10],int row,int col)
{
int i,j;
for(i=1;i<=row;i++)
   {
   for(j=1;j<=col;j++)
   {
      printf("%d ",m[i][j]);
    }
   printf("\n");
   }
}
void main()
{
int m1[10][10],m2[10][10],m3[10][10],m,n;
clrscr();
printf("Enter number of rows :");
scanf("%d",&m);
printf("Enter number of colomns :");
scanf("%d",&n);
read_matrix(m1,m,n);
read_matrix(m2,m,n);
multiply_matrix(m1,m2,m3,m,n);
display_matrix(m3,m,n);
getch();
}
```

➢Program 6.6.13 :Write user defined functions to implement following string operations

(i)strcat(ii)Strlen ***(May 2016)***

(i)strcat

```
#include<stdio.h>
void main()
{
void strcat(char *,char * );
clrscr();
```

```
char a[100], b[100];
printf("Enter the first string\n");
gets(a);
printf("Enter the second string\n");
gets(b);
strcat(a,b);
getch();
}
void strcat (char *p, char *q)
{
char c[100],*z;
z=c;
while( *p != '/0')
{
*z=*p;
p++;
z++;
}
while( *\q != '/0')
{
*z=*q;
q++;
z++;
}
*z= '/0';
printf("The concatenated string is=%s", c);
}
```

(ii)strlen

```
#include<stdio.h>
void main()
{
void strlen(char * );
clrscr();
char a[100];
printf("Enter the string\n");
gets(a);
strlen(a);
```

```
getch();
}
void strlen (char *p)
{
int c=0;
while( *p != '/0')
{
c++;
p++;
}
printf("The string length is=%d", c);
}
```

Syllabus Topic : Pointer - Introduction, Definition and uses of Pointers , Pointer Variables, Void Pointer, Array of Pointers

Q. Write short note on pointers.

Ans. :

Pointers

- Pointers are variables that are used to store the address of another variable.
- Address of a variable is the memory location number which is allotted to the variable. The memory addresses are 0, 1,2, 3... and so on up to the capacity of the memory. The address is normally displayed in hexadecimal form. Hexadecimal form is a representation of number somewhat similar to binary number. Here four binary digits are combined together to form a hexadecimal number.
- Pointers unlike other variables do not store values. As stated they store the address of other variables.
- It is already mentioned in the first statement that pointers are also variables. Hence, we can also have a pointer that is pointing to another pointer.
- We will discuss about these pointers to pointer in the later sections. To begin with let us see the syntax of declaring pointers, operators required for pointers and some basic programs.
- Syntax of pointer declaration :

Data_type *ptr_name;

Wherein "Data_type" is the data type of the variable to which the pointer is supposed to point. If we want a pointer to point to an integer than, we need to have the data type of the pointer as "int", for a float type data pointer should also be of the "float" type and so on.

The "ptr_name" is an identifier i.e. the name of the pointer. The same rules of identifiers apply to the pointer name as to any other variable declaration.

The most important difference in the declaration of a pointer is the "*" sign given before the pointer name.

- Hence, according to the syntax seen above, if we want to declare a pointer for "int" type data then we can declare it as given in the example below:

int *p;

Here, the pointer name is "p". Hence, "p" can be used as a pointer to point to any of the variable of type "int".

Syllabus Topic : Address Operator and Dereferencing Pointer

Q.Explain reference and de-reference operators with example. ***(May 2013)***

Ans. :

- There are two operators required in the pointer based programs viz. Address of operator ("&") and Value of operator ("*"). Let us see the use of these two operators in detail.

1.Address of operator ("&")

2.Value of operator ("*")

1.Address of operator ("&")

- It is also called as the referencing operator.
- This operator returns the address of the variable associated with the operator.
- For e.g., if we write "&x", it will return the address of the variable "x".
- Hence, if we have a pointer "p", which we want to point to a variable x, then we need to copy the address of the variable "x" in the pointer variable "p". This is implemented by the statement:

p=&x;

i.e. the address of the variable "x" is copied into the pointer variable "p", hence the pointer "p" pointing to the variable "x".

2.Value of operator ("*")

- It is also called as the de-referencing operator.
- This operator returns the value stored in the variable pointed by the specified pointer.
- For e.g., if we write "*p", it will return the value of the variable pointed by the pointer "p".
- Hence, if we want the value of the variable pointed by the pointer "p" to be stored in a variable "y", then the expression can be written as:

y=*p;

i.e. the value of the variable pointed by the pointer "p" is stored in the variable "y".

➢Program 6.8.1 : Write the output of the following program.

```
#include<stdio.h>
#include<conio.h>
void main()
{
int a,*p;
clrscr();
  a=125;
p=&a;
printf("%d\n",a);
printf("%x\n",p);
printf("%d\n",*p);
```

```
getch();
}
```

Output

```
125
Address of variable a
125
```

Syllabus Topic : Pointers to Pointers

➢Program 6.8.2 : Write the output of the following program.

```
#include<stdio.h>
#include<conio.h>
void main()
{
int a,*p,**p1;
clrscr();
a=125;
p=&a;
p1=&p;
printf("%d\n",a);
printf("%x\n",p);
printf("%x\n",p1);
printf("%d\n",*p);
printf("%x\n",*p1);
printf("%d\n",**p1);
getch();
}
```

Output

```
125
Address of variable a
Address of pointer variable p
125
Address of variable a
125
```

Syllabus Topic : Pointer - Pointer Arithmetic

➢Program 6.8.3 : Write the output of the following program.

```
#include<stdio.h>
#include<conio.h>
void main()
{
int a,*a1;
float b,*b1;
clrscr();
a1=&a;
b1=&b;
printf("%x \n%x\n",a1,b1);
a1++;
b1++;
printf("%x \n%x\n",a1,b1);
getch();
}
```

Output

```
Address of variable a
Address of variable b
(Address of variable a)+2
(Address of variable b)+4
```

➢Program 6.8.4 : Find output of the following statenal. ***(May 2015)***

```
int x=20, y, *ip;
ip= & x;
y=(*ip)++;
printf("%d \n", y);
printf("%d \n", *ip);
y=++x (*ip);
printf("%d \n", y);
printf("%d \n", ip);
```

Output

```
20
21
22
22
```

Syllabus Topic : Pointers and Array and Two Dimensional Array

Q.What is a relation between arrays and pointers? Explain with example. ***(May 2016)***

Ans. :

Relation between Arrays and Pointers

Consider and array:

int arr[4];

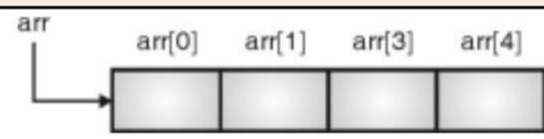


Fig. 6.8.1

- In arrays of C programming, name of the array always points to the first element of an array. Here, address of first element of an array is &arr[0]. Also, arr represents the address of the pointer where it is pointing. Hence,&arr[0] is equivalent to arr.
- Also, value inside the address &arr[0] and address arr are equal. Value in address&arr[0] is arr[0] and value in address arr is *arr. Hence, arr[0] is equivalent to *arr.

➢Program 6.8.5 : Write the output of the following program.

```
#include<stdio.h>
#include<conio.h>
void main()
{
int i,a[2]={10,20};
clrscr();
   for(i=0;i<=1;i++)
{
printf("%d\n",a[i]);
printf("%d\n",*(a+i));
printf("%d\n",*(i+a));
}
getch();
}
```

Output

```
10
10
10
20
20
20
```

➢Program 6.8.6 :Write a program to add two numbers using function. Pass the pointers to the variables as reference to the functions.

```
#include<stdio.h>
#include<conio.h>
void main()
{
int a,b;
void sum (int *p1, int *p2);
clrscr();
printf("Enter two numbers");
scanf("%d %d",&a,&b);
sum(&a,&b);
getch();
}
void sum (int *p1, int *p2)
{
int c;
c=*p1+*p2;
printf("The sum is equal to %d",c);
}
```

Output

```
Enter two numbers4
7
The sum is equal to 11
```

Q.Explain the difference in using call by reference and call by value methods.***(May 2013, Dec. 2014)***

ORWhat are the different ways for parameter passing to a function ? Explain along with example.***(Dec. 2013, May 2015, Dec. 2015)***

Ans. :

Difference between call by value and call by reference

Sr. No.	Call by Value	Call by reference
1.	In this case the value of the parameters is passed to the called function	In this case the reference of the variable is passed to the function by passing the address of the parameters
2.	In this case the actual parameters are not accessible by the called function	In this case, since the address of the variables are available, the called function can access the actual parameters
3.	This is implemented by using simple variable names	This is implemented by the use of pointer variables
4.	Hence the actual parameters remain unchanged in case of the call by value	Hence the actual parameters can be altered if required in case of the call by reference method

➢Program 6.8.7 :Write a program to swap two numbers using a function. Pass the values to be swapped to this function using call-by-value method. ***(May 2013)***

```
#include<stdio.h>
#include<conio.h>
void main()
{
int a,b;
void swap (int a, int b);
clrscr();
printf("Enter two numbers:");
scanf("%d %d",&a,&b);
printf("The values of a and b in the main function before calling the swap function are %d and %d\n",a,b);
swap(a,b);
printf("The values of a and b in main function after calling the swap function are %d and %d\n",a,b);
getch();
}
void swap (int a, int b)
{
int temp;
temp=a;
a=b;
b=temp;
printf("The values of a and b in the swap function after swapping are %d and %d\n",a,b);
}
```

Output

```
Enter two numbers:4
5
```

```
The values of a and b in the main function before calling the swap function are 4 and 5
The values of a and b in the swap function after swapping are 5 and 4
The values of a and b in main function after calling the swap function are 4 and 5
```

➢Program 6.8.8 :Write a program to swap two numbers using a function. Pass the values to be swapped to this function using call-by-reference method.

```
#include<stdio.h>
#include<conio.h>
void main()
{
int a,b;
void swap (int *p1, int *p2);
clrscr();
printf("Enter two numbers:");
scanf("%d %d",&a,&b);
printf("The values of a and b in the main function before calling the swap function are %d and %d\n",a,b);
swap(&a,&b);
printf("The values of a and b in main function after calling the swap function are %d and %d\n",a,b);
getch();
}
void swap (int *p1, int *p2)
{
int temp;
temp=*p1;
*p1=*p2;
*p2=temp;
printf("The values of a and b in the swap function after swapping are %d and %d\n",*p1,*p2);
}
```

Output

```
Enter two numbers:4
5
The values of a and b in the main function before calling the swap function are 4 and 5
The values of a and b in the swap function after swapping are 5 and 4
The values of a and b in main function after calling the swap function are 5 and 4
```

Syllabus Topic : Dynamic Memory Allocation

➢Program 6.8.9 :Write a program to accept a set of 10 numbers and print the numbers using pointers.

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

```
#include<conio.h>
void main ()
{
int *p,i,a[10];
clrscr();
    p=&a;
printf("Enter 10 nos.\n");
for(i=0;i<=9;i++)
scanf("%d",(p+i));
for(i=0;i<=9;i++)
printf("%d\n",*(p+i));
getch();
}
```

Output

```
Enter 10 nos.
1
2
3
4
5
6
7
8
9
10
1
2
3
4
5
6
7
8
9
10
```

➢Program 6.8.10 :Write a program to accept a set of 10 numbers and print the numbers using pointers. Find average of these integers.

```
#include<stdio.h>
#include<conio.h>
void main ()
{
int *p,i,sum=0,a[10];
float avg;
clrscr();
p=a;
printf("Enter 10 nos.\n");
for(i=0;i<=9;i++)
{
scanf("%d",(p+i));
sum=sum+*(p+i);
}
avg=sum/10.0;
for(i=0;i<=9;i++)
printf("%d\n",*(p+i));
printf("Average=%f",avg);
getch();
}
```

Output

```
Enter 10 nos.
1
2
3
4
5
6
7
8
9
10
1
2
3
4
5
```

```
6
7
8
9
10
Average=5.5
```

Syllabus Topic : Structure-Declaration, Initialization, Operation of Structure

Q.What do you mean by struct ? ***(Dec. 2013)***

ORWhat is the use of structure ? Explain with an example. ***(Dec. 2015, May 2016)***

Ans. :

- Structure is a collection of multiple data elements that can be of different data types.
- Array is a collection of data items of same type, while structure can have data items of different type.
- The memory space required to store one variable of structure is equal to the memory space required by all the elements independently to be stored in memory.
- Let us see the syntax of a structure declaration and declaration of a variable of the structure type.

Syntax of structure declaration

```
struct structure_name
{
data_type variable_name;
data_type variable name;
-
-
};
```

For example a structure to store the some information like name, roll number and fees of a student in college can be declared as shown below.

```
struct student
{
char name[20];
int roll_no;
float fees;
};
```

Syntax of declaration of a variable of the structure

struct structure_name structure_variable_name;

For example, to declare a variable of the above defined structure student, we can write it as :

struct student s1;

- The variables declared inside the structure are called as the member elements of the structure. For example name, roll number and fees are member elements of the structure student.
- Structure is normally used when a set of data items corresponding to a common thing are to be stored together. For example, when information of a student is to be stored, we need to store his name, roll number, fees etc. In this case we can make a structure element and store the corresponding data of every student in a separate structure element.
- To access a member element of a structure variable we use the dot operator (“.”), also called as the period operator. The syntax of accessing a structure element is given below:

structure_variable_name.varaible_name;

For example, in the above case to access the fees of the student “s1”, the period operator will be used as shown below:

s1.fees;

➢Program 6.9.1 :Write a program to store and display the name, roll number and fees of a student using structure.

```
#include<conio.h>
#include<stdio.h>
struct student
{
char name[20];
int roll_no;
float fees;
};
void main ()
{
struct student s1;
clrscr();
printf("Enter the student's name, roll number and fees paid:");
gets(s1.name);
scanf("%d %f",&s1.roll_no,&s1.fees);
printf("The student details are as follows:\nName:%s\nRoll number:%d\nFees:%f\n", s1.name, s1.roll_no,
s1.fees);
getch();
}
```

Output

```
Enter the student's name, roll number and fees paid:John Patrick
24
106000
The student details are as follows:
Name:John Patrick
```

```
Roll number:24
Fees:106000
```

➢Program 6.9.2 :Write a program to store and display the name, runs scored and wickets taken of a cricket player using structure.

```
#include<conio.h>
#include<stdio.h>
struct cricketer
{
char name[20];
int runs, wickets;
};
void main ()
{
struct cricketer c1;
clrscr();
printf("Enter the cricketer's name, runs scored and wickets taken:");
gets(c1.name);
scanf("%d %d",&c1.runs,&c1.wickets);
printf("The cricketer details are as follows:\nName:%s\nRuns scored:%d\nWickets
taken:%d\n",c1.name,c1.runs,c1.wickets);
getch();
}
```

Output

```
Enter the cricketer's name, runs scored and wickets taken:Sachin Tendulkar
20000
150
The cricketer details are as follows:
Name:Sachin Tendulkar
Runs scored:20000
Wickets taken:150
```

➢Program 6.9.3 :Write a program to store and display the name, matches played and goals scored by a hockey player using structure.

```
#include<stdio.h>
#include<conio.h>
#include<stdio.h>
struct hockey
```

```
{
char name[20];
int matches, goals;
};
void main ()
{
struct hockey c1;
clrscr();
printf("Enter the hockey player's name, matches played and goals scored:");
gets(c1.name);
scanf("%d %d",&c1.matches,c1.goals);
printf("The hockey player's details are as follows:\nNmae:%s\nMatches played:%d\nGoals
scored:%d\n",c1.name,c1.matches,c1.goals);
getch();
}
```

Output

```
Enter the hockey player's name, matches played and goals scored:V. Raghunath
67
41
The hockey player's details are as follows:
Name:V. Raghunath
Matches played:67
Goals scored:41
```

Syllabus Topic : Array of Structure

➢Program 6.10.1 :Write a program to store and display the name, marks in three subjects and roll number of 'n' students using structure. Display the output in tabular form.

```
#include<conio.h>
#include<stdio.h>
struct student
{
char name[20];
int roll_no;
int physics,chem,maths;
};
void main ()
{
```

```
struct student st[100];
int n,i,j;
clrscr();
printf("Enter the number of students:");
scanf("%d",&n);
for(i=0;i<=n–1;i++)
{
printf("Enter the student's name, roll number and marks in three subjects:");
scanf("%s %d %d %d %d" , st[i].name,&st[i].roll_no , &st[i].physics , &st[i].chem , &st[i].maths);
}
printf("Name\tRoll No\tPhysics\tChem\tMaths\n");
printf("---------------------------------------------\n");
for(i=0;i<=n–1;i++)
{
printf("%s\t%d\t%d\t%d\t%d\n",st[i].name, st[i].roll_no, st[i].physics, st[i].chem,
st[i].maths);
}
getch();
}
```

Output

```
Enter the number of students4
Enter the student's name, roll number and marks in three subjects:Jay
1
40
40
80
Enter the student's name, roll number and marks in three subjects:Ajay
12
49
49
99
Enter the student's name, roll number and marks in three subjects:Vijay
24
45
45
90
Enter the student's name, roll number and marks in three subjects:Sujoy
```

```
5
50
50
100
Name    Roll No Physics Chem    Maths
----------------------------------------
Jay    1      40     40     80
Ajay   12     49     49     99
Vijay  24     45     45     90
Sujoy  5      50     50     100
```

➢Program 6.10.2 :Write a program to store the name, runs scored and wickets taken of 'n' cricketers using structure. Display the output in tabular form.

```
#include<conio.h>
#include<stdio.h>
struct cricketer
{
char name[20];
int runs, wickets;
};
void main ()
{
struct cricketer c[100];
int n,i;
clrscr();
printf("Enter the number of cricketers");
scanf("%d",&n);
for(i=0;i<=n–1;i++)
{
printf("Enter the cricketer's name, runs scored and wickets taken:");
scanf("%s %d %d",c[i].name,&c[i].runs,&c[i].wickets);
}
printf("Name\tRuns\tWickets\n");
printf("----------------------------------\n");
for(i=0;i<=n–1;i++)
{
printf("%s\t%d\t%d\n",c[i].name,c[i].runs,c[i].wickets);
}
getch();
```

```
}
```

Output

```
Enter the number of cricketers3
Enter the cricketer's name, runs scored and wickets taken:Sachin
20000
150
Enter the cricketer's name, runs scored and wickets taken:Zaheer
500
500
Enter the cricketer's name, runs scored and wickets taken:Dhoni
10000
Name   Runs   Wickets
----------------------------------
Sachin  20000   150
Zaheer  500    500
Dhoni   10000   0
```

➢Program 6.10.3 :Write a program to store the name, roll number and marks in three subjects of 'n' students using structure. Generate a merit list with respect to the total marks scored i.e. display the output in tabular form in order of maximum total marks to minimum total marks in three subjects.

```
#include<conio.h>
#include<stdio.h>
struct student
{
char name[20];
int roll_no;
int physics,chem,maths,total;
};
void main ()
{
struct student s[100],temp;
int n,i,j;
clrscr();
printf("Enter the number of students");
scanf("%d",&n);
for(i=0;i<=n-1;i++)
{
```

```
printf("Enter the student's name, roll number and marks in three subjects:");
scanf("%s %d %d %d %d", s[i].name, &s[i].roll_no, &s[i].physics,&s [i].chem, &s[i].maths);
s[i].total=s[i].physics+s[i].chem+s[i].maths;
}
for(i=0;i<=n–1;i++)
{
for(j=0;j<=n–2;j++)
{
if(s[j].total<s[j+1].total)
{
temp=s[j];
s[j]=s[j+1];
s[j+1]=temp;
}
}
}
printf("Name\tRoll No\tPhysics\tChem\tMaths\tTotal\n");
printf("----------------------------------------------\n");
for(i=0;i<=n–1;i++)
{
printf("%s\t%d\t%d\t%d\t%d\t%d\n", s[i].name, s[i].roll_no, s[i].physics, s[i].chem, s[i].maths, s[i].total);
}
getch();
}
```

Output

```
Enter the number of students : 4
Enter the student's name, roll number and marks in three subjects:Jay
1
40
40
80
Enter the student's name, roll number and marks in three subjects:Ajay
12
49
49
99
Enter the student's name, roll number and marks in three subjects:Vijay
24
```

```
45
45
90
Enter the student's name, roll number and marks in three subjects:Sujoy
5
50
50
100
Name    Roll No Physics Chem    Maths   Total
---------------------------------------------
Sujoy   5     50     50     100    200
Ajay    12    49     49     99     197
Vijay   24    45     45     90     180
Jay     1     40     40     80     160
```

➢Program 6.10.4 :Write a program to store the name, matches played and goals scored by 'n' hockey players using structure. Generate a list with goals scored in descending order i.e. display the output in tabular form in order of maximum goals scored to minimum goals scored.

```
#include<conio.h>
#include<stdio.h>
struct hockey
{
char name[20];
int matches,goals;
};
void main ()
{
struct hockey s[100],temp;
int n,i,j;
clrscr();
printf("Enter the number of hockey players:");
scanf("%d",&n);
for(i=0;i<=n–1;i++)
{
printf("Enter the player's name, matches played and goals scored:");
scanf("%s %d %d",s[i].name, &s[i].matches, &s[i].goals);
}
for(i=0;i<=n–1;i++)
{
```

```
for(j=0;j<=n-2;j++)
{
if(s[j].goals<s[j+1].goals)
{
temp=s[j];
s[j]=s[j+1];
s[j+1]=temp;
}
}
}
printf("Name\t\tMatches\tGoals\n");
printf("----------------------------------\n");
for(i=0;i<=n-1;i++)
{
printf("%s %d %d\n",s[i].name,s[i].matches,s[i].goals);
}
getch();
}
```

Output

```
Enter the number of hockey players:4
Enter the player's name, matches played and goals scored:DhyanChand
100
500
Enter the player's name, matches played and goals scored:BalbirSingh
75
450
Enter the player's name, matches played and goals scored:AjitPalSingh
67
761
Enter the player's name, matches played and goals scored:DhanrajPillay
120
1058
Name          Matches Goals
----------------------------------
DhanrajPillay  120    1058
AjitPalSingh  67     761
DhyanChand    100    500
```

```
BalbirSingh   75    450
```

➢Program 6.10.5 :Write a program to store the name, matches played and runs scored by 'n' cricket players using structure. Generate a list with runs scored in descending order i.e. display the output in tabular form in order of maximum runs scored to minimum runs scored.

```
#include<conio.h>
#include<stdio.h>
struct cricketer
{
char name[20];
int matches,runs;
};
void main ()
{
struct cricketer s[100],temp;
int n,i,j;
clrscr();
printf("Enter the number of cricketers:");
scanf("%d",&n);
for(i=0;i<=n-1;i++)
{
printf("Enter the player's name, matches played and runs scored:");
scanf("%s %d %d",s[i].name,&s[i].matches,&s[i].runs);
}
for(i=0;i<=n-1;i++)
{
for(j=0;j<=n-2;j++)
{
if(s[j].runs<s[j+1].runs)
{
temp=s[j];
s[j]=s[j+1];
s[j+1]=temp;
}
}
}
printf("Name\t\tMatches\tRuns\n");
printf("----------------------------------\n");
for(i=0;i<=n-1;i++)
```

```
{
printf("%s\t%d\t%d\n",s[i].name,s[i].matches,s[i].runs);
}
getch();
}
```

Output

```
Enter the number of cricketers:5
Enter the player's name, matches played and runs scored:SachinT.
171
14006
Enter the player's name, matches played and runs scored:SunilGavaskar
125
10122
Enter the player's name, matches played and runs scored:RahulDravid
144
11581
Enter the player's name, matches played and runs scored:BrianLara
131
11953
Enter the player's name, matches played and runs scored:AllanBorder
156
11174
Name        Matches Runs
----------------------------------
SachinT.    171   14006
BrianLara    131   11953
RahulDravid   144   11581
AllanBorder   156   11174
SunilGavaskar 125   10122
```

Syllabus Topic : Structure within Structure

Q. What do mean by nested structure ? ***(Dec. 2013, Dec. 2015, May 2016)***

Ans. :

- If one of the members of structure is also a structure, then such a structure is called as a nested structure.
- The syntax of a nested structure can be as given below :

```
struct structure_name
{
```

```
data_type variable_name;
-
struct
{
data_type variable_name;
-
-
} internal_structure_name;
-
-
};
```

For example to describe a circle we need its radius and centre. The centre requires x and y co–ordinates. Hence, this can be described by a nested structure as shown below :

```
struct circle
{
float radius;
struct
{
float x,y;
}centre;
}
```

- To access the element of an internal structure the syntax is as given below:

structure_variable name.internal_structure_name.element_name;

For example if we make an variable of structure circle as struct circle c, then to access the "x" co-ordinate of the structure we will have the syntax as "c.centre.x"

➢Program 6.11.1 :Write a program to store the radius and centre of a circle using a nested structure. Also display this information.

```
#include<conio.h>
#include<stdio.h>
struct circle
{
float radius;
struct
{
float x,y;
}centre;
};
```

```
void main ()
{
struct circle c;
clrscr();
printf("Enter the radius and x and y co-ordinates of circle:");
scanf("%f %f %f",&c.radius,&c.centre.x,&c.centre.y);
printf("Circle Information\nRadius=%f\nCentre co-ordinates:(%f, %f)",c.radius, c.centre.x,c.centre.y);
getch();
}
```

Output

```
Enter the radius and x and y co-ordinates of circle:5.6
2.3
3.6
Circle Information
Radius=5.6
Centre co-ordinates:(2.3,3.6)
```

➢Program 6.11.2 :Write a nested structure to store the information of a book i.e. its title, author's name, publisher's name, edition and ISBN number. The author's name and publisher's name must be a structure with two elements each namely first name and surname.

```
struct book
{
char title[20];
struct
{
char f_name[20],surname[20];
}author;
   struct
{
char f_name[20],surname[20];
}publisher;
    int edition;
    float ISBN;
};
```

Syllabus Topic : Union – Definition, Operation on Union

Q.What is Unions ?

Ans. :

- Union is a collection of multiple data elements that can be of different data types. But, only one of these data items can be stored in the union variable.
- The memory space required to store one variable of union is equal to the memory space required by the largest element in the union.
- Let us see the syntax of a union declaration.

```
Syntax of structure declaration:
union union_name
{
data_type variable_name;
data_type variable name;
-
-
};
```

For example, a union to store the some information like name, roll number or id of a student in college can be declared as shown below.

```
union student
{
char name[20];
int roll_no;
float id;
};
```

Syntax of declaration of a variable of the union:

union union_name union_variable_name;

For example, to declare a variable of the above defined union student, we can write it as :

union student s1;

- The memory space required by the variable "name" of the union is 20 bytes (since; one byte of one char data); for "roll_no", the memory space required is 2 bytes while for "id" the space required is 4 bytes. Hence, the space allotted to the variable of the union is 20 bytes as the maximum space is required to store the "name", which requires 20 bytes.

Syllabus Topic : Difference between Structure and Union

Q. What is difference between structure and union?

Ans. :

Defference between structure and union

Sr. No.	Structure	Union
1.	Memory allotted for a structure is equal to the space require collectively by all the members of that structure.	Memory allotted for a union is equal to the space required by the largest member of that union.
2.	Data is more secure in structures.	Data can be corrupted in a union.
3.	Structure provide ease of programming.	Unions are comparatively difficult for programming.
4.	Structure requires more memory.	Union requires less memory.
5.	Structures must be used when information of all the member elements of a structure are to be stored.	Unions must be used when only one of the member elements of the union is to be stored.

➢Program 6.12.1 :Write a program to store the information of a person as his name or ID number using union. Ask the user for the information choice.

```
#include<conio.h>
#include<stdio.h>
union info
{
char name[20];
long id;
};
void main ()
{
union info i1;
    int choice;
clrscr();
printf("Enter your information\n1. Name\n2.ID number\nEnter your choice:");
scanf("%d",&choice);
switch(choice)
{
case 1:printf("Enter your name:");
scanf("%s",i1.name);
break;
case 2:printf("Enter your ID number:");
scanf("%ld",&i1.id);
break;
default:printf("Invalid Choice");
}
if(choice==1)
```

```
printf("Your Name entered is %s",i1.name);
else if(choice==2)
printf("Your ID number entered is %ld",i1.id);
getch();
}
```

Output

```
Enter your information
1. Name
2.ID number
Enter your choice:1
Enter your name:HarishNarula
Your Name entered is HarishNarula
```

➢Program 6.13.1 :Define a structure "Hockey" consisting of following elements :

(i)Player name

(ii)Name of the country

(iii)Number of matches played

(iv)Number of goals scored

Write a program to read records of 'N' players and to prepare following lists :

(i)List prepared according to player's name

(ii)List prepared according to country's name

(iii)List prepared according to number of matches played

(iv)List prepared according to number of goals scored

```
#include<stdio.h>
#include<conio.h>
#include<stdio.h>
struct hockey
{
char name[20],country[20];
int matches,goals;
};
void main ()
{
struct hockey s[100],temp;
int n,i,j,k;
clrscr();
printf("Enter the number of hockey players:");
scanf("%d",&n);
for(i=0;i<=n-1;i++)
{
```

```
printf("Enter the player's name, country, matches played and goals scored:");
scanf("%s %s %d %d",s[i].name,s[i].country,&s[i].matches,&s [i].goals);
}
for(i=0;i<=n–1;i++)
{
for(j=0;j<=n–2;j++)
{
if(s[j].goals<s[j+1].goals)
{
temp=s[j];
s[j]=s[j+1];
s[j+1]=temp;
}
}
}
printf("\nList according to goals scored\nName\t\tCountry\tMatches\tGoals\n");
printf("----------------------------------------\n");
for(i=0;i<=n–1;i++)
{
printf("%s\t%s\t%d\t%d\n", s[i].name, s[i].country, s[i].matches, s[i].goals);
}
for(i=0;i<=n–1;i++)
{
for(j=0;j<=n–2;j++)
{
if(s[j].matches<s[j+1].matches)
{
temp=s[j];
s[j]=s[j+1];
s[j+1]=temp;
}
}
}
printf("\nList according to matches played\nName\t\tCountry\tMatches\tGoals\n");
printf("----------------------------------------\n");
for(i=0;i<=n–1;i++)
{
printf("%s\t%s\t%d\t%d\n", s[i].name, s[i].country, s[i].matches, s[i].goals);
}
for(i=0;i<=n–1;i++)
```

```
{
for(j=0;j<=n-2;j++)
{
for(k=0;s[j].name[k]!='\0';k++)
{
if(s[j].name[k]!=s[j+1].name[k])
{
if(s[j].name[k]>s[j+1].name[k])
{
temp=s[j];
s[j]=s[j+1];
s[j+1]=temp;
}
break;
}
}
}
}
printf("\nList according to Name\nName\t\tCountry\tMatches\tGoals\n");
printf("----------------------------------------\n");
for(i=0;i<=n-1;i++)
{
printf("%s\t%s\t%d\t%d\n", s[i].name, s[i].country, s[i].matches, s[i].goals);
}
for(i=0;i<=n-1;i++)
{
for(j=0;j<=n-2;j++)
{
for(k=0;s[j].name[k]!='\0';k++)
{
if(s[j].country[k]!=s[j+1].country[k])
{
if(s[j].country[k]>s[j+1].country[k])
{
temp=s[j];
s[j]=s[j+1];
s[j+1]=temp;
}
```

```
break;
}
}
}
}
printf("\nList according to Country\nName\t\tCountry\tMatches\tGoals\n");
printf("----------------------------------------\n");
for(i=0;i<=n-1;i++)
{
printf("%s\t%s\t%d\t%d\n", s[i].name, s[i].country, s[i].matches, s[i].goals);
}
getch();
}
```

Output

```
Enter the number of hockey players:3
Enter the player's name, country, matches played and goals scored:DanielSedin
Sweden
705
208
Enter the player's name, country, matches played and goals scored:ZachParise
USA
407
160
Enter the player's name, country, matches played and goals scored:JoeThornton
Canada
915
285
List according to goals scored
Name          Country Matches Goals
---------------------------------------------
JoeThornton    Canada  915    285
DanielSedin    Sweden  705    208
ZachParise    USA    407    160
List according to matches played
Name          Country Matches Goals
---------------------------------------------
JoeThornton    Canada  915    285
```

```
DanielSedin   Sweden  705    208
ZachParise    USA    407    160
List according to Name
Name          Country Matches Goals
---------------------------------
DanielSedin   Sweden  705    208
JoeThornton   Canada  915    285
ZachParise    USA    407    160
List according to Country
Name          Country Matches Goals
---------------------------------
JoeThornton   Canada  915    285
DanielSedin   Sweden  705    208
ZachParise    USA    407    160
```

➢Program 6.13.2 :Define structure within structure consisting of following elements : ***(Dec. 2014)***

(i)Employee code

(ii)Employee name

(iii)Employee salary and

(iv)Employee date_of_joining

Write a C program to read at least 10 records and display them.

```
#include<conio.h>
#include<stdio.h>
struct employee
{
char name[20];
int salary,code;
struct
{
int date,month,year;
}date_joining;
};
void main ()
{
struct employee e[100];
int i;
clrscr();
for(i=0;i<=9;i++)
{
```

```
printf("Enter the employee's name, code, salary and date of joining as date, month and year separately:");
scanf("%s %d %d %d %d %d",e[i].name,&e[i].code,&e[i].salary,
&e[i].date_joining.date, &e[i].date_joining.month,&e[i].date_joining.year);
}
printf("\nEmployee list\nName\tCode\tSalary\tDate of joining\n");
printf("-----------------------------------------------\n");
for(i=0;i<=9;i++)
{
printf("%s\t%d\t%d\t%d/%d/%d\n",e[i].name,e[i].code,e[i].salary,e[i].date_joining.date,e[i].date_joining.mon
th,e[i].date_joining.year);
}
getch();
}
```

Output

```
Enter the employee's name, code, salary and date of joining as date, month and year separately:Manish
0001
30000
1
8
2005
Enter the employee's name, code, salary and date of joining as date, month and year separately:Satish
0002
25000
1
8
2009
Enter the employee's name, code, salary and date of joining as date, month and year separately:Girish
0003
29500
10
8
2009
Enter the employee's name, code, salary and date of joining as date, month and year separately:Sunit
0004
21000
10
9
2010
```

```
Enter the employee's name, code, salary and date of joining as date, month and year separately:sumit
0005
29300
1
1
2001
Enter the employee's name, code, salary and date of joining as date, month and year separately:saumil
0006
26636
1
12
2000
Enter the employee's name, code, salary and date of joining as date, month and y ear separately:santosh
0007
12500
1
1
2011
Enter the employee's name, code, salary and date of joining as date, month and year separately:Prashant
8
18250
1
1
2011
Enter the employee's name, code, salary and date of joining as date, month and year separately:Ruchi
9
12500
1
1
2011
Enter the employee's name, code, salary and date of joining as date, month and year separately:Rashmi
10
12000
1
1
2009
Employee list
Name        Code    Salary  Date of joining
```

```
--------------------------------------------------
Manish1300001\8\2005
Satish225000 1\8\2009
Girish329500  10\8\2009
Sunit4 21000  10\9\2010
Sumit5   29300  1\1\2001
Saumil6   26636 1\12\2000
Santosh7   12500  1\1\2011
Prashant8   18250  1\1\2011
Ruchi 9    12500  1\1\2011
Rashmi 10  12000  1\1\2009
```

➢Program 6.13.3 :Design a structure named employeeto print names and nos. of employee who have 5 years or more experience and salary less than Rs. 10,000 using array of structure (Name, No, Experience and Salary as member).

```
#include<conio.h>
#include<stdio.h>
struct employee
{
char name[20];
int salary,exp,number;
};
void main ()
{
struct employee e[100];
int i;
clrscr();
for(i=0;i<=9;i++)
{
printf("Enter the employee's name, number, exp and salary");
scanf("%s %d %d %d",e[i].name,&e[i].number,&e[i].exp,&e[i].salary);
}
printf("\nEmployee list with 5+ years experience and salary 10000 or less\nName\tNumber\n");
printf("-------------------------\n");
for(i=0;i<=9;i++)
{
if(e[i].exp>=5 && e[i].salary<10000)
printf("%s\t%d\n",e[i].name,e[i].number);
}
```

```
getch();
}
```

Output

```
Enter the employee's name, number, exp and salaryManish
1234123
9
30000
Enter the employee's name, number, exp and salarySatish
1235151
12
12000
Enter the employee's name, number, exp and salaryGirish
12414515
3
5000
Enter the employee's name, number, exp and salarySunit
12424352
8
9000
Enter the employee's name, number, exp and salarySumit
123135213
6
12000
Enter the employee's name, number, exp and salarySaumil
12414124
6
26636
Enter the employee's name, number, exp and salarySantosh
124124
4
8500
Enter the employee's name, number, exp and salaryPrashant
1241241
5
8500
Enter the employee's name, number, exp and salaryRahul
1241221
```

```
7
9900
Enter the employee's name, number, exp and salaryAjay
7893911
16
8900
Employee list with 5+ years experience and salary 10000 or less
Name        Number
-------------------------
Sunit       12424352
Prashant     1241241
Rahul        1241221
Ajay        7893911
```

➢Program 6.13.4 :Define a structure called cricket that will describe the following information-player's name, country name, best score, batting average. Develop a program that will store information of 25 cricket players around the world using this structure. Also display names of these cricketers in descending order with respect to their batting average.***(May 2015)***

```
#include<conio.h>
#include<stdio.h>
struct cricketer
{
char name[20], country[20];
int best;
float avg;
};
void main ()
{
struct cricketer c1[25],temp;
   int i,j;
float dummy;
clrscr();
for(i=0;i<=24;i++)
{
printf("Enter the cricketer's name, country, best score and average runs scored:");
scanf("%s %s %d %f",c1[i].name,c1[i].country,&c1[i].best,&dummy);
c1[i].avg=dummy;
}
```

```
for(i=0;i<=24;i++)
{
for(j=0;j<=23;j++)
{
if(c1[j].avg<c1[j+1].avg)
{
temp=c1[j];
c1[j]=c1[j+1];
c1[j+1]=temp;
}
}
}
for(i=0;i<=24;i++)
printf("%s\t%s\t%d\t%f\n", c1[i].name, c1[i].country, c1[i].best, c1[i].avg);
getch();
}
```

Output

```
Enter the cricketer's name, country, best score and average runs scored:Sachin
India
200
45.16
Enter the cricketer's name, country, best score and average runs scored:Jayasuriya
SriLanka
189
32.36
Enter the cricketer's name, country, best score and average runs scored:Ponting
Australia
164
42.69
Enter the cricketer's name, country, best score and average runs scored:Inzamam
Pakistan
137
39.52
Enter the cricketer's name, country, best score and average runs scored:Ganguly
India
183
41.02
```

```
Enter the cricketer's name, country, best score and average runs scored:Kallis
SouthAfrica
139
45.45
Enter the cricketer's name, country, best score and average runs scored:Dravid
India
153
39.43
Enter the cricketer's name, country, best score and average runs scored:Lara
WestIndies
169
40.48
Enter the cricketer's name, country, best score and average runs scored:Jayawardene
SriLanka
144
33.44
Enter the cricketer's name, country, best score and average runs scored:Yousuf
Pakistan
141
41.71
Enter the cricketer's name, country, best score and average runs scored:Gilchrist
Australia
172
35.89
Enter the cricketer's name, country, best score and average runs scored:Sangakkara
SriLanka
138
38.15
Enter the cricketer's name, country, best score and average runs scored:Azharuddin
India
153
36.92
Enter the cricketer's name, country, best score and average runs scored:DeSilva
SriLanka
145
34.90
Enter the cricketer's name, country, best score and average runs scored:Anwar
Pakistan
```

194

39.21

Enter the cricketer's name, country, best score and average runs scored:Chanderpaul

WestIndies

150

41.60

Enter the cricketer's name, country, best score and average runs scored:Haynes

WestIndies

152

41.37

Enter the cricketer's name, country, best score and average runs scored:Atapattu

SriLanka

132

37.57

Enter the cricketer's name, country, best score and average runs scored:Waugh

Australia

173

39.35

Enter the cricketer's name, country, best score and average runs scored:Gibbs

SouthAfrica

175

36.13

Enter the cricketer's name, country, best score and average runs scored:Gayle

WestIndies

153

39.06

Enter the cricketer's name, country, best score and average runs scored:Yuvraj

India

139

37.62

Enter the cricketer's name, country, best score and average runs scored:Fleming

NewZealand

134

32.40

Enter the cricketer's name, country, best score and average runs scored:Shewag

India

175

35.11

```
Enter the cricketer's name, country, best score and average runs scored:Waugh
Australia
120
32.9
KallisSouthAfrica13945.45
SachinIndia20045.16
Dravid India15339.43
PontingAustralia16442.69
YousufPakistan14141.71
Chanderpaul WestIndies15041.6
HaynesWestIndies15241.37
GangulyIndia18341.02
LaraWestIndies16940.48
Inzamam  Pakistan13739.52
WaughAustralia17339.35
Anwar Pakistan19439.21
GayleWestIndies15339.06
SangakkaraSriLanka13838.15
Yuvraj India13937.62
AtapattuSriLanka13237.57
AzharuddinIndia15336.92
GibbsSouthAfrica17536.13
GilchristAustralia17235.89
ShewagIndia17535.11
DeSilvaSriLanka14534.9
JayawardeneSriLanka14433.44
WaughAustralia12032.9
FlemingNewZealand13432.4
JayasuriyaSriLanka18932.36
```

➢Program 6.13.5 :Write a c=program to create an array of structure to store the details of almost 100 employees. And sort it according to employee ID. Employee details are as follows :

(i)Employee Name

(ii)Employee ID

(iii)Employee salary ***(May 2013)***

```
#include<conio.h>
#include<stdio.h>
struct employee
{
```

```
char name[20];
int salary,code;
};
void main ()
{
struct employee e[100],temp;
int i,j;
clrscr();
for(i=0;i<=99;i++)
{
printf("Enter the employee's name, code and salary:");
scanf("%s %d %d ",e[i].name,&e[i].code,&e[i].salary);
}
for(i =0;i<=98;i++)
{
for(j =0; j <=98;j++)
{
if(e[j].code > e[j+1].code)
{
temp=e[j];
e[j]=e[j+1];
e[j+1]=temp;
}
}
}
printf("\nEmployee list\nName\tCode\tSalary\n");
printf("-----------------------------------------------\n");
for(i=0;i<=99;i++)
{
printf("%s\t%d\t%d\n",e[i].name,e[i].code,e[i].salary);
}
getch();
}
```

Output

```
Enter the employee's name, code, salary and date of joining as date, month and year separately:Manish
0001
30000
```

```
1
8
2005
Enter the employee's name, code, salary and date of joining as date, month and year separately:Satish
0002
25000
1
8
2009
Enter the employee's name, code, salary and date of joining as date, month and year separately:Girish
0003
29500
10
8
2009
Enter the employee's name, code, salary and date of joining as date, month and year separately:Sunit
0004
21000
10
9
2010
Enter the employee's name, code, salary and date of joining as date, month and year separately:sumit
0005
29300
1
1
2001
Enter the employee's name, code, salary and date of joining as date, month and year separately:saumil
0006
26636
1
12
2000
Enter the employee's name, code, salary and date of joining as date, month and y ear separately:santosh
0007
12500
1
1
```

```
2011
Enter the employee's name, code, salary and date of joining as date, month and year separately:Prashant
8
18250
1
1
2011
Enter the employee's name, code, salary and date of joining as date, month and year separately:Ruchi
9
12500
1
1
2011
Enter the employee's name, code, salary and date of joining as date, month and year separately:Rashmi
10
12000
1
1
2009
Employee list
Name        Code   Salary  Date of joining
-----------------------------------------------
Manish1300001\8\2005
Satish225000 1\8\2009
Girish329500  10\8\2009
Sunit4 21000  10\9\2010
Sumit5   29300  1\1\2001
Saumil6   26636 1\12\2000
Santosh7   12500  1\1\2011
Prashant8   18250  1\1\2011
Ruchi 9    12500  1\1\2011
Rashmi 10  12000  1\1\2009
```

➢Program 6.13.6 :A sports club of cricket needs to maintain data about players. Description is given below: Club wants to maintain players name, age, no. of matches played, no. of runs and average. For above description declare a structure and comment about the size of structure of your declaration. ***(Dec. 2013)***

```
struct cricketer
{
```

```
char name[20];
int age, mathces_played, runs;
float average;
};
```

- The memory space required for a "char" type data is one byte, for "int" type data is 2 bytes and for "float" type data it is 4 bytes. Thus, 20 bytes will be required for name, 2 bytes each for age, matches_played and runs, while 6 bytes for average. Hence the total space required for the variable created of structure "cricketer" will be 20 + 2 + 2 + 2 + 4 = 30 bytes.

➢Program 6.13.7 :A Hospital needs to maintain details of patients . Details to be maintained are, First name, Middle name, Surname, Date of Birth, Disease. Write a C program which will print the list of all patients with given disease. ***(May 2014)***

```
#include<stdio.h>
#include<conio.h>
struct patient
{
char fname[20],sname[20],mname[20],birthdate[20],disease[20];
};
void main ()
{
struct patient s[100];
int n,i,j,k;
char d[20];
clrscr();
printf("Enter the number of patients:");
scanf("%d",&n);
for(i=0;i<=n-1;i++)
{
printf("Enter the first name, middle name, surname, date of birth and disease:");
scanf("%s %s %s %s %s",s[i].fname,s[i].mname,s[i].sname,s[i].birthdate,s[i].disease);
}
printf("\nEnter the disease whose patients are to be listed:");
scanf("%s",d);
printf("Patients with the said disease are:\n");
for(j=0;j<=n-1;j++)
{
for(k=0;d[k]!='\0';k++)
{
if(d[k]!=s[j].disease[k])
```

```
{
break;
}
}
if(d[k]=='\0') printf("%s %s %s\n",s[j].fname,s[j].mname,s[j].sname);
}
getch();
}
```

➢Program 6.13.8 :A company needs to maintain data about their employees. Details to be maintained are Employee name, Department, Date of joining, Salary. Write a program which will store these details and list the employees whose salary is greater than Rs. 50000.00. ***(May 2016)***

```
#include<conio.h>
#include<stdio.h>
struct employee
{
char name[20],dept[20];
int salary;
struct
{
int date,month,year;
}date_joining;
};
void main ()
{
struct employee e[100];
int i;
clrscr();
for(i=0;i<=9;i++)
{
printf("Enter the employee's name, department, salary and date of joining as date, month and year
separately:");
scanf("%s %s %d %d %d %d",e[i].name,&e[i].dept,&e[i].salary,
&e[i].date_joining.date, &e[i].date_joining.month, &e[i].date_joining.year);
}
printf("\nEmployee list\nName\tDept\tSalary\tDate of joining\n");
printf("--------------------------------------------------\n");
for(i=0;i<=9;i++)
```

```
{
if(e[i].salary>50000) printf("%s\t%d\t%d\t%d/%d/%d\n",e[i].name,e[i].dept,e[i].salary,e[i].date_joining.date
,e[i].date_joining.month,e[i].date_joining.year);
}
getch();
}
```

Output

```
Enter the employee's name, department, salary and date of joining as date, month and year separately:Manish
Computers
60000
1
8
2005
Enter the employee's name, department, salary and date of joining as date, month and year separately:Satish
Computers
25000
1
8
2009
Enter the employee's name, department, salary and date of joining as date, month and year separately:Girish
Electronics
69500
10
8
2009
Enter the employee's name, department, salary and date of joining as date, month and year separately:Sunit
Mechanical
21000
10
9
2010
Enter the employee's name, department, salary and date of joining as date, month and year separately:sumit
Electronics
29300
1
1
2001
Enter the employee's name, department, salary and date of joining as date, month and year separately:saumil
```

```
Civil
26636
1
12
2000
Enter the employee's name, department, salary and date of joining as date, month and y ear separately:santosh
Mechanical
12500
1
1
2011
Enter the employee's name, department, salary and date of joining as date, month and year separately:Prashant
Mechanical
88250
1
1
2011
Enter the employee's name, department, salary and date of joining as date, month and year separately:Ruchi
Computers
12500
1
1
2011
Enter the employee's name, department, salary and date of joining as date, month and year separately:Rashmi
Civil
12000
1
1
2009
Employee list
Name          Dept    Salary  Date of joining
-----------------------------------------------
ManishComputers600001\8\2005
GirishElectronics69500  10\8\2009
PrashantMechanical    88250  1\1\2011
```

➢Program 6.14.1 :Write a program to create singly linked list of 10 student nodes. The student node contain roll no and percentage. Read information and print this information in the linked list.

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

```
# include<conio.h>
# include "malloc.h"
struct student
{
int roll_no;
float percentage;
struct student *next;
};
void main()
{
int i;
struct student s[10];
struct student p;
float dummy;
clrscr();
for(i=0;i<=9;i++)
{
printf("Enter roll number and percentage of the student:");
scanf("%d %f",&s[i].roll_no,&dummy);
s[i].percentage=dummy;
if(i<9)
s[i].next=&s[i+1];
else
s[i].next=0;
}
p.next=&s[0];
printf("Roll No.\tPercentage\n");
while(p.next!=0)
{
p=*p.next;
printf("%d\t\t%f\n",p.roll_no,p.percentage);
}
getch();
}
```

Output

```
Enter roll number and percentage of the student:1
98
```

```
Enter roll number and percentage of the student:2
56
Enter roll number and percentage of the student:3
76
Enter roll number and percentage of the student:4
87
Enter roll number and percentage of the student:5
93
Enter roll number and percentage of the student:6
81
Enter roll number and percentage of the student:7
90
Enter roll number and percentage of the student:8
67
Enter roll number and percentage of the student:9
37
Enter roll number and percentage of the student:10
75
Roll No.      Percentage
1             98
2             56
3             76
4             87
5             93
6             81
7             90
8             67
9             37
10            75
```

Syllabus Topic : Types of File, File Operations - Opening, Closing, Creating, Reading, Processing File

Q.Explain how to read the contents of file and write into the file with Syntax. ***(May 2013)***

ORWhat do you mean by FILE ? What are the different functions available to read data from the file ? Specify the different modes in which files can be opened along with syntax.***(Dec. 2013, May 2014, Dec. 2014, Dec. 2015, May 2016)***

Ans. :

- For file handling or accessing the contents of file, there are certain predefined functions available in the C programming language.

- An important thing required to access files is the "FILE pointer". This pointer is used to point to the values stored in the file. A file pointer is hence to be created for accessing the files. The syntax for creating a file pointer is as given below:

FILE *<identifier for pointer>;

For e.g. FILE *fp;

- Hence in every program we write in this section to access files, we will use this kind of pointer declaration. This pointer is used to point the data to be accessed in the file i.e. whenever a data is read or written in the file, it is from the location pointed by the file pointer "fp".
- Let us see these functions and there use in the programming.

1.fopen()

- This function is used to open a file to be accessed in the program. The file to be opened is to be passes as a string parameter to the function and also the mode of opening the file is to be passed as the string to the function. Hence the syntax of the function with parameters is as given below :

<file pointer identifier> = fopen("<file name>", "<mode of opening the file>")

For e.g. fp=fopen("test.txt","w");

- This example statement opens the file "test.txt" in write mode and the pointer used along with the function to read/write is the file pointer "fp". the various modes in which a file can be opened are as listed below :

(i)"r" indicates that the file is to be opened indicates in the read mode.

(ii)"w" indicates that the file is to be opened indicates in the write mode. When a file is opened in write mode the data written in the file overwrites the previously stored data in the file.

(iii)"a" indicates that the file is to be opened in the append mode. In this mode the data written into the file is appended towards the end of the already stored data in the file. The earlier stored data remains as it is.

(iv)"w+" indicates that the file is to be opened in write and read mode

(v)"r+" indicates that the file is to be opened in read and write mode

2.fclose()

- This function is used to close the file opened using the file pointer passed to the function. The syntax with parameters to call this function is as given below :

fclose(<file pointer identifier>);

For e.g. fclose(fp);

- This statement closes the file opened using the file pointer variable "fp". It closes the file opened using the function fopen(). The file opened to be accessed in whatsoever mode doesn't matters, it is closed using the fclose() function.

3.feof()

- This function returns true or false based on whether the pointer pointing to the file has reached the end of the file or not. the pointer used to point the file has to be passed as a parameter to the function feof().
- Also the file has to be opened pointed using the pointer "fp", before using this function. The syntax of the function with the parameters is as shown below :

feof(<file pointer identifier>);

For e.g. feof(fp);

4.fputc()

- This function is used to put a character type data into the opened file using the fopen() function, pointed by a file pointer.
- The character to be put into the file as well as the file pointer are to be passed as the parameters to this function.
- The syntax to call this function along with the parameters to be passed is as shown below :

fputc(<char type data>, <file pointer identifier>);

For e.g. : fputc(c,fp);

- This example will store the character value of the char type data variable "c" into the opened file and pointed by the file pointer fp, at the position pointed by the pointer fp in the file.

5.getc()

- This function is used to get a character from the file pointed by the corresponding file pointer passed to the function. It is exactly opposite the fputc() function.
- This function brings the character from the file opened and pointed by the file pointer variable passed to the function.
- The syntax of the function call with the parameters to be passed is as given below :

getc(<file pointer identifier>);

For e.g. getc(fp);

- This function brings the character type data from the opened file using the pointer fp; from the location pointed by the pointer "fp" in that file.

6.rewind()

- This function is used to rewind or bring the file pointer variable to point to the beginning of the file from wherever it is currently pointing in the file. The syntax of the function call with the paramters to be passed is as given below:

rewind(<file pointer identifier>);

For e.g. rewind(fp);

- This function rewinds or brings back the pointer "fp" to the beginning of the file from wherever it was pointing in the file opened using the pointer "fp".

7.fprintf()

- This function is used to store the different data types in the file as the fputc() function is used to store the character in the file.
- This can be used to store integer, float, string etc types of data into the file opened. The function is similar to printf(), except that it writes to the file instead of the monitor.
- The other difference is in the syntax, that the file pointer variable is also to be passed to the function along with the data types and the parameter values or the variables. The syntax shown below explains the concept:

fprintf(<file pointer identifier>, "<format specifiers>",<variable names>);

For e.g.: fprintf(fp,"%d",x);

- This function will print or store the value of the integer variable "x" in the file opened using the pointer "fp". The data will be stored at the location pointed by the pointer variable "fp".

8.fscanf()

- The function is used to read the different types of data as the getc() function is used to read a character from the file.
- This function can be used to read an integer, float string etc types of data into the file opened. The function is similar to scanf(), except that it reads from the file instead of the keyboard.
- The other difference is in the syntax, that the file pointer variable is also to be passed to the function along with the data types and the parameter values or the variables. The syntax shown below explains the concept:

  fscanf(<file pointer variable>,"<format specifiers>",<address of the variables in which the data is to be read>);

For e.g.: fscanf(fp,"%d",&x);

- This function will read the integer type data from the file pointed by the file pointer variable "fp" and store this value in the variable "x". The value will be brought from the file at the position pointed by the file pointer in the file.

➢Program 6.15.1 :Write a program to accept a set of characters from user until the user presses the full stop (' . ') and store it in a text file. Read from the file and display the contents of the file. ***(May 2013)***

```
# include<stdio.h>
# include<conio.h>
void main()
{
FILE *fp;
char c=' ';
clrscr();
fp=fopen("test.txt","w");
printf("Write data to be stored in the file and once completed press the full stop (.):\n");
while(c!='.')
{
scanf("%c",&c);
fputc(c,fp);
}
fclose(fp);
fp=fopen("test.txt","r");
while(!feof(fp))
{
printf("%c",getc(fp));
```

```
}
fclose(fp);
getch();
}
```

Output

```
Write data to be stored in the file and once completed press the full stop (.):
This is file handling program in C.
This is file handling program in C.
```

➢Program 6.15.2 :Write a program to accept a set of characters from user until the user presses the full stop (' . ') and append it in a text file. Read from the file and display the contents of the file.

```
# include<stdio.h>
# include<conio.h>
void main()
{
FILE *fp;
char c=' ';
clrscr();
fp=fopen("test.txt","r");
while(!feof(fp))
{
printf("%c",getc(fp));
}
fclose(fp);
fp=fopen("test.txt","a");
printf("\nWrite data to be stored in the file and once completed press the full stop (.):\n");
while(c!='.')
{
scanf("%c",&c);
fputc(c,fp);
}
fclose(fp);
fp=fopen("test.txt","r");
while(!feof(fp))
{
printf("%c",getc(fp));
}
fclose(fp);
```

```
getch();
}
```

Output

```
This is file handling program in C.
Write data to be stored in the file and once completed press the full stop (.):
This is a program that appends the previously added data in the file test.
This is file handling program in C. This is a program that appends the previously added data in the file test.
```

➢Program 6.15.3 :Write a program to accept a set of characters from user until the user presses the full stop (' . ') and overwrite it in a text file. Read from the file and display the contents of the file.

```
# include<stdio.h>
# include<conio.h>
void main()
{
FILE *fp;
char c=' ';
clrscr();
fp=fopen("test.txt","r");
while(!feof(fp))
{
printf("%c",getc(fp));
}
fclose(fp);
fp=fopen("test.txt","w+");
printf("\nWrite data to be stored in the file and once completed press the full stop (.):\n");
while(c!='.')
{
scanf("%c",&c);
fputc(c,fp);
}
rewind(fp);
while(!feof(fp))
{
printf("%c",getc(fp));
}
fclose(fp);
getch();
```

```
}
```

Output

```
This is file handling program in C. This is a program that appends the previously added data in the file test.
Write data to be stored in the file and once completed press the full stop (.):
Now this is the new data in the file.
Now this is the new data in the file.
```

➢Program 6.15.4 :Write a program to accept the name and roll number of a student and store it in a text file. Read the stored data and display the same from the file.

```
# include<stdio.h>
# include<conio.h>
void main()
{
FILE *fp;
char name[20],temp1[20];
int roll,temp2;
clrscr();
fp=fopen("Student.txt","w+");
printf("Enter name and roll number of the student:");
scanf("%s %d",name,&roll);
fprintf(fp,"%s %d",name,roll);
printf("Name\tRoll no.\n");
rewind(fp);
fscanf(fp,"%s %d",temp1,&temp2);
printf("%s\t%d",temp1,temp2);
fclose(fp);
getch();
}
```

Output

```
Enter name and roll number of the student: Ajay
24
Name    Roll no.
Ajay    24
```

➢Program 6.15.5 :Write a program to accept the name and roll number of a student and store it in a text file. Read the stored data and display the same from the file. It should be menu driven program that can have multiple entries. The previous data should be retained and new data can be appended in the file. All the entries can be displayed if required.

```
# include<stdio.h>
# include<conio.h>
void main()
{
FILE *fp;
char name[20];
int roll,choice=0;
clrscr();
while(choice!=3)
{
printf("1.New Entry\n2.Display all entries\n3.Exit\nEnter your choice");
scanf("%d",&choice);
switch(choice)
{
case 1:fp=fopen("Student.txt","a");
printf("Enter name and roll number of the student:");
scanf("%s %d",name,&roll);
fprintf(fp,"%s %d",name,roll);
fclose(fp);
break;
case 2:fp=fopen("Student.txt","r");
printf("Name\tRoll no.\n");
while(!feof(fp))
{
fscanf(fp,"%s %d",name,&roll);
printf("%s\t%d\n",name,roll);
}
fclose(fp);
break;
case 3: break;
default: printf("Invalid Choice\n");
}
}
getch();
}
```

Output

```
1.New Entry
```

```
2.Display all entries
3.Exit
Enter your choice2
Name    Roll no.
Ajay    24
Hiral   33
Harish  24
Khushi  55
1.New Entry
2.Display all entries
3.Exit
Enter your choice1
Enter name and roll number of the student:Girish
11
1.New Entry
2.Display all entries
3.Exit
Enter your choice1
Enter name and roll number of the student:Harsha
23
1.New Entry
2.Display all entries
3.Exit
Enter your choice2
Name    Roll no.
Ajay    24
Hiral   33
Harish  24
Khushi  55
Girish  11
Harsha  23
1.New Entry
2.Display all entries
3.Exit
Enter your choice3
```

➢Program 6.15.6 :Write a program to copy the contents from one text file to another.

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

```
# include<conio.h>
void main()
{
FILE *fp,*fp1;
char c;
clrscr();
fp=fopen("test.txt","r");
fp1=fopen("test1.txt","w");
while(!feof(fp))
{
c=getc(fp);
fputc(c,fp1);
}
fclose(fp1);
fclose(fp);
fp1=fopen("test1.txt","r");
printf("Displaying data from the new file:\n");
while(!feof(fp1))
{
printf("%c",getc(fp1));
}
fclose(fp1);
getch();
}
```

Output

```
Displaying data from the new file:
Now this is the new data in the file.
```

➢Program 6.15.7 :Write a program to count the number of characters in a text file.

```
# include<stdio.h>
# include<conio.h>
void main()
{
FILE *fp;
int n=0;
clrscr();
fp=fopen("test.txt","r");
while(!feof(fp))
```

```
{
getc(fp);
n++;
}
n--;
printf("The total number of characters in the file is %d:\n",n);
fclose(fp);
getch();
}
```

Output

```
The total number of characters in the file is 37
```

➢Program 6.15.8 :Write a program to copy text from one file to other after converting Lower case letters to Upper case and vice versa. Keep other characters as it is. ***(May 2016)***

➢Program :

```
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>
void main()
{
  FILE *fp1, *fp2;
  char ch;
  clrscr();
  fp1 = fopen("Sample.txt", "r");
  fp2 = fopen("Output.txt", "w");
   while (1) {
   ch = fgetc(fp1);
   if(ch>=65 && ch<=90)
ch=ch+32;
  if (ch == EOF)
    break;
    else
    putc(ch, fp2);
  }
   printf("File copied Successfully!");
  fclose(fp1);
  fclose(fp2);
```

□□□

```
{
```