**B.Tech.** (ECS-101)

# First Semester Examination 2010-11

## Computer Concepts and Programming in C

**Time : 3 Hours]** **[Total Marks : 100**

*Note : (1) This question paper contains **three** sections.*
*(2) Attempt **all** questions.*

**Section-A**

**Q. 1. There are 10 multiple choice types of questions. Only one of the answers is correct. State correct answer : (10 × 1 = 10)**

**Q. 1. (a) Which of the following is not an operating system ?**

**(A) Linux**
**(B) JAVA**
**(C) DOD**
**(D) WINDOWS**

**Q. 1. (h) Program written in High Level Language is converted to Machine Code hy :**

**(A) Operating System**
**(B) Assemhler**
**(C) Compiler**
**(D) Machine Language**

**Q. 1. (c) Which of the following doesn't makes use of condition in its execution ?**

**(A) Loop**
**(B) Case**
**(C) Function**
**(D) Decision**

**Q. 1. (d) Which of the following is not a hinary numher ?**

**(A) 0**
**(B) 1**
**(C) 2**
**(D) 10**

**Q. 1. (e) Which of the following clause is used to include I/O function lihrary ?**

**(A) include # <io.h>**
**(B) include # <stdio.h>**
**(C) # include <stdio.h>**
**(D) none of the ahove**

**Q. 1. (f) Which one of the following is a corrcct statement for checking a condition in IF THEN ELSE statemcnt ?**

**(A) IF (A = B)**
**(B) IF (A ≈ B)**
**(C) IF (A & B)**
**(D) IF (A * B)**

**Q. 1. (g) Which keyword is used to define structure ?**

**(A) structure**
**(B) STRUCTURE**
**(C) STRUCT**
**(D) None of the above**

**Q. 1. (h) Which method of input-output is used in stack ?**

**(A) First In First Out**
**(B) First In Last Out**
**(C) Both A and B**
**(D) None of the ahove**

**Q. 1. (i) How many elements will be there in A[5][5] array ?**

**(A) 5**
**(B) 25**
**(C) 50**
**(D) None of the ahove**

**Q. 1. (j) In a C program, thc statement for (i=0; i<5: i++) will iterate the loop :**

**(A) 4 Times**
**(B) 5 Times**
**(C) 6 Times**
**(D) None of thc ahove**

**Answers**

**(a)** (B); **(h)** (C); **(c)** (C); **(d)** (C); **(e)** (C);
**(f)** (A); **(g)** (D); **(h)** (B); **(i)** (B); **(j)** (B).

**Q. 2. State whcther the following statements are True or False : (5 × 1 = 5)**

**Q. 2. (a) An identifier in C must start with a lotter or underscore. It is not allowed to have a space or a hyphen.**

**Ans.** True

**Q. 2. (b) Inltiálization of all elements of an array can be done at the time of declaration and definltion.**

**Ans.** True

**Q. 2. (c) List is a non linear data structure.**

**Ans.** True

**Q. 2. (d) A function in C cannot be invoked by other function.**

**Ans.** False

**Q. 2. (e) The name of an array is a pointer only to the first element, not the whole array.**

**Ans.** True

**Q. 3. Fill in the blanks :** **(5 × 1 = 5)**

**Q. 3. (a) The ..... function reads data from keyboard.**

**Q. 3. (b) An ..... is a sequence of operators and operands tbat reduces to a single value.**

**Q. 3. (c) The statement ..... will declare *pa* as pointer to character variable.**

**Q. 3. (d) Mode ..... opens an existing file for read operations only.**

**Q. 3. (e) ..... is a predefined macro of C Preprocessor.**

**Answers :** (a) scan f; (b) expression;
(c) char * pa; (d) f open; (e) #

## SECTION-B

**Q. 4. There are seven (07) parts. Attempt any five (05) parts.** **(6 × 5 = 30)**

**Q. 4. (a) (i) What are dlfferences in GUI interface and CUI interface ? Give one example of each to explain your answer.**

**Ans.** A graphical user interface (GUI), is a type of user interface that allows users to interact with programs in more way than typing such as computers; hand-held devices such as MP3 players, portable media players or gaming devices; household appliances and office equipment with images rather than text commands. A GUI offers graphical icons, and visual indicators, as opposed to text-based interfaces, typed command labels or text navigation to fully represent the information and actions available to a user. The actions are usually performed through direct manipulation of the graphical elements.

The term GUI is historically restricted to the scope of character user Interface or CUI is like graphical user interface which is used for the input and output of the data in the computer, except in the GUI apart from text there are also graphical contents present, in CUI there is only use of text typed one after another just as commands used in MS DOS.

**Q. 4. (a) (ii) Convert the following decimal numbers to equivalent binary numbers :**

**575, 311.55.**

**Ans.** 575, 311 · 55

| | | |
|---|---|---|
| 2 | 575, 311 | |
| 2 | 287655 | 1 |
| 2 | 143827 | 1 |
| 2 | 71913 | 1 |
| 2 | 35956 | 1 |
| 2 | 17978 | 0 |
| 2 | 8989 | 0 |
| 2 | 4494 | 1 |
| 2 | 2247 | 0 |
| 2 | 1123 | 1 |
| 2 | 561 | 1 |
| 2 | 280 | 1 |
| 2 | 140 | 0 |
| 2 | 70 | 0 |
| 2 | 35 | 0 |
| 2 | 17 | 1 |
| 2 | 8 | 1 |
| 2 | 4 | 0 |
| 2 | 2 | 0 |
| 2 | 1 | 0 |
| 2 | 0 | 1 |

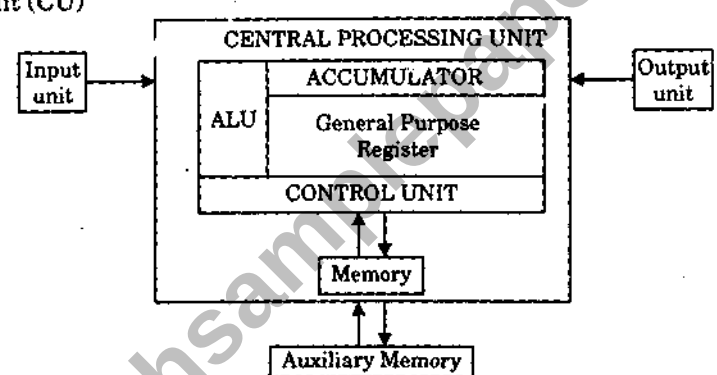
$0.55 \times 2 = 1.10 = 1$
$0.10 \times 2 = 0.20 = 0$
$0.20 \times 2 = 0.40 = 0$
$0.40 \times 2 = 0.80 = 0$
$0.80 \times 2 = 1.60 = 1$
$0.60 \times 2 = 1.20 = 1$

$\therefore (575311.55)_{10} = (10001100011101001111 \cdot 100011)_2$

**Q. 4. (h) (i) Draw a neat schematic of a digital computer and explain the role of each function unit.**

**Ans.** The fundamental units of computer is

(i) Central Processing Unit (CPU)
(ii) Memory
(iii) Input Unit
(iv) Output Unit
(v) Auxiliary Memory
(vi) Arithmetic and Logical Unit (ALU)
(vii) Control Unit (CU)




**(i) Central Processing Unit (CPU) :** It is the brain of a computer where all kinds of processing is done. This unit takes the input data from the input devices and processes it according to the set of instructions called program. The output of processing of the data are directed to the functions devices for used in the output world. The major functions of the CPU is to store the data temporily in the registers and perform arithmetic and logical computations.

**(ii) Memory :** The memory is a computer is analogous to a note-book where we may note down various things for future reference. It is also known by the term storage and its function is to store coded form of information from the human operator through the input device or from other computers connected to it.

**(iii) Input Unit :** It represents the various input devices which are used to input the data into the computer. The func.. ..n of the input unit is to accept coded information from the human operator or from an electro mechanical device or from other computers connected to it through the internet or by any other media.

**(iv) Output Unit :** It is used to represent the information processing by the digital computer. The function of the output unit is to store the processed information and display it as and when needed by the user. The CRT terminal and printers are example of output devices.

**(iv) Auxiliary Memory :** It is also known as secondary storage devices. They are used to take the back-up (copy) of important files and data. The data and the programs are loaded into the main memory prior to the execution of the program. The processed data and the results are kept in the secondary storage device for future reference.

**(v) Arithmetic and Logical Unit (ALU) :** It is the unit of computer system, responsible for all calculation works, arithmetic as well as logical. The control unit supplies the calculative data to ALU, so that ALU-can perform, the arithmetic operations (addition, subtraction, multiplication, division etc) and also logical operations (comparisons and decision making). After performing the calculations, the result is again stored to the memory unit by control unit.

**(vi) Control Unit (CU) :** As the name indicates, this unit controls all other units of the computer system. It instructs the input unit to receive the data and also to store the data. Similarly, it controls the data flow from memory to ALU and vice-versa. It also controls the data flow to output unit. In all, we can say that the Control Unit works as the Nerrous System for the entire computer system.

**Q. 4. (b) (ii) What is Algorithm ? Discuss basic characteristic of an algorithm.**

**Ans.**

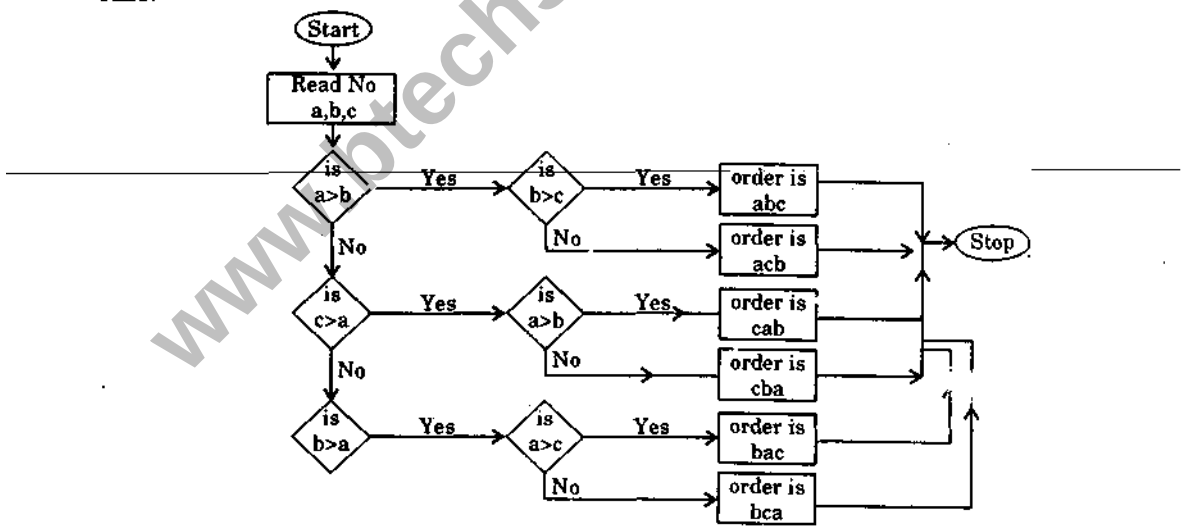
- An algorithm is a set of rules for carrying out calculations either by hand or on a machine.
- An algorithm is a sequence of computational steps that transform the input into the output.
- An algorithm is a sequence of operations performed on data that have to be organized in data structures.
- An algorithm is an obstruction of a program to be executed on a physical machine.

**The basic characteristic of an algorithm are :**

1. Input
2. Output
3. Definiteness
4. Effectiveness
5. Termination

**Q. 4. (c) (i) Draw a flow chart for arranging three numbers a, b and c in ascending order.**

**Ans.**




**Q. 4. (c) (ii) What is a role of SWITCH statement in C programming language ? Give the syntax of SWITCH statement with a suitable example.**

**Ans.** The control statement that allows us to make a decision from the number of choices is called a switch, or more correctly a switch-case-default, since these keywords go together to make up the control statement.

(a) The program that used switch, we can put the cases like 1, 2, 3, etc. in any order as we please.

(b) We are also allowed to use char values in case and switch, In fact when we use 'v', 'a', 'x' they are actually replaced by the ASCII values (118, 97, 120) of these character constants.

(c) We can make use of the fact that once a case is satisfied the control simply falls through the case till it doesn't encounter a break statement.

(d) Even if there are multiple statements to be executed in each case there is no need to enclose them within a pair of braces (unlike if, and else).

(e) Every statement in a switch must belong to some case or the other. If a statement doesn't belong to any case the compiler won't report an error. However, the statement would never get executed.

(f) We can check the value of any expression in a switch. Thus the following switch statement are legal :

```
switch (i + j * k)
switch (23 + 45% 4 * k)
switch (a < 4 && b > 7)
```

(g) The switch statement is very useful while writing menu driven programs.

**Syntax of Switch :**

```
switch (integer expression)
{
case constant 1:
    do this;
case constant 2:
    do this;
case constant 3:
    do this;
default :
    do this;
}
```

Now, according to Question WAP using switch # include <stdio.H>

```
void main ( )
{
switch (i)
{
case 1:
printf ("I am in case 1/n");
case 2 :
printf ("I am in case 2/n");
case 3 :
printf ("I am in case 3/n");
default :
printf ("I am in default/n");
    }
    }
```

The output of this program would be :

I am in case 1
I am in case 2
I am in case 3
I am in default

**Q. 4. (d) (i) What are Functions ? What are advantages of using multiple functions in a program ?**

**Ans.** A function is a self-contained block of statements that perform a coherent task of some kind. Every C program can be thought of as a collection of these functions. If a program has only one function then it must be the main ( ) function.

**Advantages of using multiple functions :**

(a) To avoid repetition of code and bulky programs functionally related statements are isolated into a function.

(b) Function declaration specifies the return type of the function and the types of parameters it accepts.

(c) Function definition defines the body of the function.

(d) Variables declared in a function are't available to other functions in a program. So, there won't be any clash even if we give same name to the variables declared in different functions.

(e) A function can be called either by value or by reference.

(f) Pointers can be used to make a function return more than one value simultaneously.

**Q. 4. (d) (ii) What do you mean by Call by Value technique for function call ? Explain with a suitable example.**

**Ans.** Arguments can generally by passed to functions in one of the two ways :

(a) Sending the values of the arguments.

(b) Sending the addresses of the arguments.

In this method, the 'value' of each of the actual arguments in the calling function is copied into corresponding formal arguments of the called function. With this method the changes made to the formal arguments in the called function have no effect on the values of actual arguments in the calling function. The following program illustrates the "call by value".

```
main ( )
{
int a = 10, b = 20;
swapv (a, b);
printf ("\n a = % d, b = %d", a, b);
}
swapv (int x, int y)
{
int t;
t = x;
x = y;
y = t;
printf ("\n x = %d  y = %d", x, y);
}
```

The output :

x = 20, y = 10

a = 10, b = 20

The values of a and b remain unchanged even after exchanging the values of x and y.

**Q. 4. (e) (i) What is difference in searching and aorting ? Write an algorithm to sort a list contalning ten numbers.**

**Ana. Searching :** Searching is a process of finding an element within the list of elements stored in any, order or randomly. Searching is divided into two categories linear and binary search. Linear searching is the basic and simple method of searching. Binary search takes some less time to search an element from the sorted list of element. So we can say that binary search method is more efficient then the linear search only drawback is that binary search work on the sorted list where there is no prerequisite for the linear search.

**Sortlng :** Sorting refers to the operation of arranging data in some given sequence *i.e.*, Increasing order or decreasing order. Sorting is categorised as Internal sorting and External sorting. By internal sorting means we are arranging the numbers within the array only which is in computer primary memory, where the external sorting is the sorting of numbers from the external file by reading it from secondary memory.

Now, according to question the algorithm to sort a list containing ten numbers.

1. Initialisation
   Set I = 0
2. Repeat steps 3 to 5 until I < N
3. Set J = 0
4. Repeat step 5 until J < N – C – 1
5. If A[J] > A[J+1] then
   Set temp = A[J]
   Set A[J] = A[J + 1]
   Set A [J+ 1] = temp
   End if
6. Exit

**Q. 4. (e) (ii) What are differences In Array and Structures ? Explain with an example.**

**Ans. Array :** An array is defined as a set of finite number of homageneous elements or data items. It means an array can contain one type of data only, either all integers, all floating-point numbers, or all characters. Declaration of arrays is as follows :

```
int a [10];
```

Where int specifies the data type or type of elements array stores "a" is the name of array, and the number specified inside the square brackets is the numbers of elements an array can store, this is called the length or size of array.

**For example :** Program to input values into an array and display them

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

```
main ( )
{
int arr [5], i;
for (i = 0, i < 5, i++)
{
printf ("Enter the value for arr [%d]:", i);
scanf ("%d", & arr [i]);
}
printf ("The array elements are:\n");
for (i = 0; i < 5; i++)
printf ("%d\t", arr [i]);
printf ("\n");
}
```

**Output :**

```
Enter the value for arr [0] : 12
Enter the value for arr [1] : 45
Enter the value for arr [2] : 59
Enter the value for arr [3] : 98
Enter the value for arr [4] : 21
The array elements are :
12      45      59      98      21
```

**Structure :** Structure creates a template or format that describes the characteristics of its members. All the variables that would be declared of this structure type, will take the form of this template. The general syntax of a structure definition is :

```
struct tagname {
datatype member 1;
datatype member 2;
.....
.....
datatype member N;
};
```

Here struct is a keyword, which tells the compiler that a structure is being defined member 1, member 2, ..... member N are known as members of the structure.

**For example :** Program to display the values of structure members

```
# include <stdio.h>
# include <string.h>
    struct student {
    char name [20];
    int rollno ;
    float marks;
    };
main ( )
{
struct student stu 1 = {"Mary", 25, 68};
struct student stu 2, stu3;
strcpy (stu2. name, "John");
stu 2, roll no = 26;
stu 2. marks = 98;
printf ("Enter name, roll no and marks for stu 3:");
scanf ("%s %d %f ", stu3.name, & stu 3 roll no, & stu 3. marks);
printf ("stu 1: %s %d %2f\n", stu 1.name, stu1.roll no, stu 1 marks);
printf ("stu 2: %s %d %2f\n", stu 2. name, stu 2. roll no. stu2.marks);
printf ("stu 3: %s %d %s2f\n", stu 3.name, stu 3.rull no. stu3.marks);
}
```

```
output
Enter name, roll no and marks for
stu 3 : Tom 27 79.5
stu 1 : mary 25 68.00
stu 2 : John 26 98.00
stu 3 : Tom 27 79.50
```

**Q. 4. (f) (i) What is a pointer ? How pointers are declared in C programming language ? Illustrate with a suitable example.**

**Ans.** A pointer provides a way of accessing a variable without referring to the variable directly. A pointer variable is a variable that contains the address of another variable. Pointers can be used in the following situations :

(1) Call by address.

(2) Pass array and string from one function to another.

(3) To return more than one value from function.

(4) Manipulation in arrays more easily.

(5) Creates complex data structure such as linked list, graph, trees, etc.

(6) Compile faster, more efficient code.

*e.g.*, of an array of pointers, Array of pointers is a collection of more than one pointers. *i.e.*, it contains the address of more than one variable

```
main ( )
{
int *p [3], 11 declaration of array pointers
int x = 1, y = 2, z = 3;
p[0] = & x;
p[1] = & y;
p[2] = & z;
printf ("%d%d%d", *p[0], *p[1], *p[2]);
}
```

**Q. 4. (f) (ii) Write a program segment in C to swap the values of two variables using pointers.**

**Ans.**

```
#include < stdio.h>
#include <conio.h>
void main ( )
{
int a, b,
printf ("/n enter the valuea of a & b");
scanf ("%d%d", & a, & b);
a = a + b;
b = a – b;
a = a – b;
printf ("the swapped values are a = %d, b = %d");
getch ( );
}
```

**Q. 4. (g) (i) What are differences in Stack and List ? Explain how an element can be deleted from a atack and List.**

**Ans. Stack :** A stack is an ordered list in which all insertions and deletions are made at one end, called the top. Given a stack s = (a [1], a [2], ..... a [n]) then we say that a[1] is the bottommost element and element a[1] is on top of element a[i – 1]. The restrictions on a stack imply that if the elements A, B, C, D, E are added to the stack, n that order, then the first element to be removed/deleted must be E.Equivalently we say that the last element to be inserted into the stack will be the first to be removed. For this reason stacks are sometimes referred to as Last In First Out (LIFO) lists.

**Deletion from stack :**

```
Procedure delete (var item : items);
{remove top elei ent from the stack stack and put it in the item}
begin
if top = 0 then stack empty;
item : = stack (top);
top : = top – 1;
end; {of delete}
```

The above procedure are so simple that they perhaps need no more explanation. Procedure delete actually combines the functions TOP and DELETE, stackfull and stackempty are procedures which are left unspecified since they will depend upon the particular application often a stackfull condition will signal that more storage needs to be allocated and the program re-run stackempty is often a meaningful condition.

**List :** A linked list is one of the fundamental data structures, and can be used to implement other data structures. It consists of a sequence of nodes, each containing arbitrary data fields and one or two references ("links") pointing to the next and/or previous nodes. The principal benefit of a linked list over a conventional array is that the order of the linked items may be different from the order that the data items are stored in memory or a disk, allowing the list of items to be traversed in a different order. A linked list is a self-referential datatype because it contains a pointer or link to another datum of the same type. Linked lists permit insertion and removal of nodes at any point in the list in constant time, but don't allow random access. Several different types of linked list exist : singly-linked list, doubly-linked list, and circularly-linked lists.

**Deleting the first node from the list**

```
Step 1. [check for under flow ?]
    if START = NULL, then
    print 'Linked list empty'
    Exit
End if
```

Step 2. Set PTR = START
Step 3. Set START = START → next
Step 4. Print, element deleted is, pra → Info
Step 5. free (Ptr)
The (code for the above algorithm is void
dele_beg (void)
{Node * P;
if (start = = NULL)
{return;}
else
{P = start;
start = start → next;
Printf ("Element deleted is = y.d") P
→ num);
free (P);
}
}

**Q. 4. (g) (ii) What do you mean by C Preprocessor ? Give example of any two preprocessor directives.**

**Ans.** The C preprocessor is exactly what its name implies. It is a program that processes our source program before it is passed to the compiler. Preprocessor commands (often known as directives) from what can almost be considered a language within C language. We can certainly write C programs without knowing anything about the preprocessor or its facilities. But preprocessor is such a great convenience that virtually all C programmers rely on it.

The preprocessor offers several features called preprocessor directives. Each of these preprocessor directives begins with a # symbol. The directives can be placed anywhere in a program but are most often placed at the begining of a program, before the first function definition.

Following are the preprocessor directives :
(a) Macro expansion
(b) file inclusion
(c) Conditional compilation
(d) Miscellaneous directives

**Macro Expansion**

# include <stdio.h>
# define UPPER 25
void main ( )
{
int i;
for (i = 1; i < = UPPER; i++)
printf ("\n % d", i);
}

In this program, instead of writing 25 in the for loop we are writing it in the form of UPPER, which has already been defined before main ( ) through the statement,

# define UPPER 25

This statement is called 'macro definition' or more commonly, just a 'macro'.

**File Inclusion :** This directive causes one file to be included in another. The preprocessor command for fill inclusion looks like this :

# include "filename"

and it simply causes the entire contents of filename to be inserted into the source code at that point in the program.

Actually there exist two ways to write # include statement. These are :

# include "filename"
# include <filename>

The meaning of each of these forms is given below

**# include "goto.b" :** This command would look for the file goto.h in the current directory as well as the specified list of directories as mentioned in the include search path that might have been set up.

**# include <goto.h> :** This command would look for the file goto.h in the specified list of directories only.

## SECTION-C

**Q. 5. This section contains seven (07) programming problems.**

**Attempt any five (05) problems.**

**(5 × 10 = 50)**

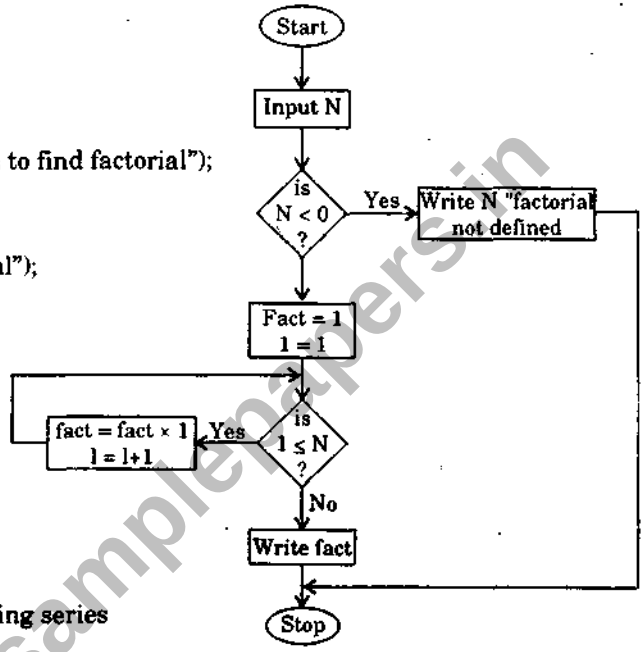
**Q. 5. (a) Draw a flowchart and write a function in C to calculate factorial of given number and also write a program to**

**calculate the sum of the following series using the above fn :**

$$S = 1! + 2! + 3! + \ldots + N!$$

**Ans.** WAP to calculate factorial of given number

```
# include <stdio.h>
# include <conio.h>
void main ( )
{
clrscr ( );
printf ("hello world that is my program to find factorial");
int a, b, c, n, i;
b = 1;
printf ("\n enter the no. to find factorial");
scanf ("%d", & n);
printf ("\n factorial as follows");
for (i = n; i > 1; i – –)
{ c = b * i;
b = c;
}
printf ("%d", c);
getch ( );
}
```




**WAP to calculate the sum of the following series**

```
# include <stdio.h>
# include (conio.h>
int fact (int);
void main ( )
{int n, result =0;
clrser ( );
printf ("Enter the number");
scanf ("%d", & n);
for (i = 1, i ≤ n; i ++)
result = fact (i);
result = result + fact (i)
printf ("\n the factorial of series %d", result);
getch ( );
}
int fact (int x)
{
if (x = = 1)
{
return (x);
}
else
{
return (x * fact (x–1));
}
}
```

**Q. 5. (h) Write a program in C that accepts roll numher and name of student's of a class size of one hundred students along with the marks ohtained hy them in Physics, Chemlstry and Mathematics. Print roll number and the name of top ten students in the order of merit. The merit is hased on the sum of the marks ohtained in the three subjects.**

**Ans.**

```
# include < stdio.H.
# define N 5
struct student
{
char name [20];
int roll no;
int marks [3];
int total,
```

```
char grade;
};
void display (struct student arr);
void calculate (struct student arr [ ]);
void sort (struct student arr [ ]);
main ( )
{
struct student stu [N], temp;
int i, j;
for (i = 0; i < N, i++)
{
printf ("Enter name:");
scanf ("%s", stu [i].name);
printf ("Enter roll no : ");
scanf ("%d", & stu [i].roll no);
stu [i].total = 0;
printf ("Enter marks in 3 subjects : ");
for (j = 0, j < 3, j++)
scanf ("%d", & stu[i] marks [j]);
}
calculate (stu);
sort (stu);
for (i = 0; i < N, i++)
display (stu [i]);
}
void calculate (struct student stu [ ])
{
int i, j;
for (i = 0; i < N; i++)
{
for (j = 0; j < 3, j++)
stu [i]. total + = stu [i].marks [j];
if (stu [i]. total > 500)
stu [i].grade = 'A';
else if (stu [i]. total > 400)
stu [i].grade = 'B';
else if (stu [i]. total > 250)
stu [i].grade = 'C';
else
stu [i].grade = 'D';
}
}
void sort (struct student stu [ ])
{
int i, j;
struct student temp;
for (i = 0; i < N – 1; i ++)
for (j = i + 1; j < N; j++)
if (stu [i] . total < stu [j]. total)
{
temp = stu [i];
stu [i] = stu [j];
stu [j] = temp;
}
}
void display (struct student stu)
{
int i;
printf ("Roll no – %d\t", stu. roll no);
printf ("Name – %s\n", stu.name);
printf ("Total – %d\t", stu.total);
printf ("Grade – %(\n\n", stu.grade);
}
```

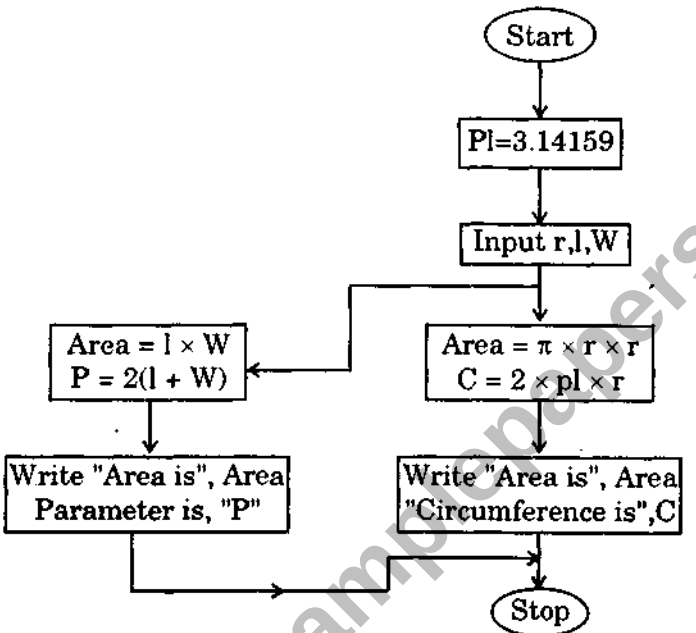
**Q. 5. (c) Write a program in C that accept the length and width of a rectangle and print the area. The area of a rectangle is calculated hy a function and returns it value to main program where it is printed. The values of length and width of rectangle are accepted from the keyboard. Also give the flowchart and algorithm.**

**Ans.**

```
main ( )
{
float area;
clrscr ( );
area = rect_area ( );
printf ("The area of the rectangle is % f", area);
getch ( );
}
float rect-area ( )
{
float length, width;
printf ("Enter the length of rectangle");
scanf ("%f", & length);
printf ("Enter the width of rectangle");
```

```
scanf ("%f", & width);
return length * width;
}
```

**Flowchart :**




**Q. 5. (d) Write a program in C to calculate the sum of the following series upto *n*th term.**

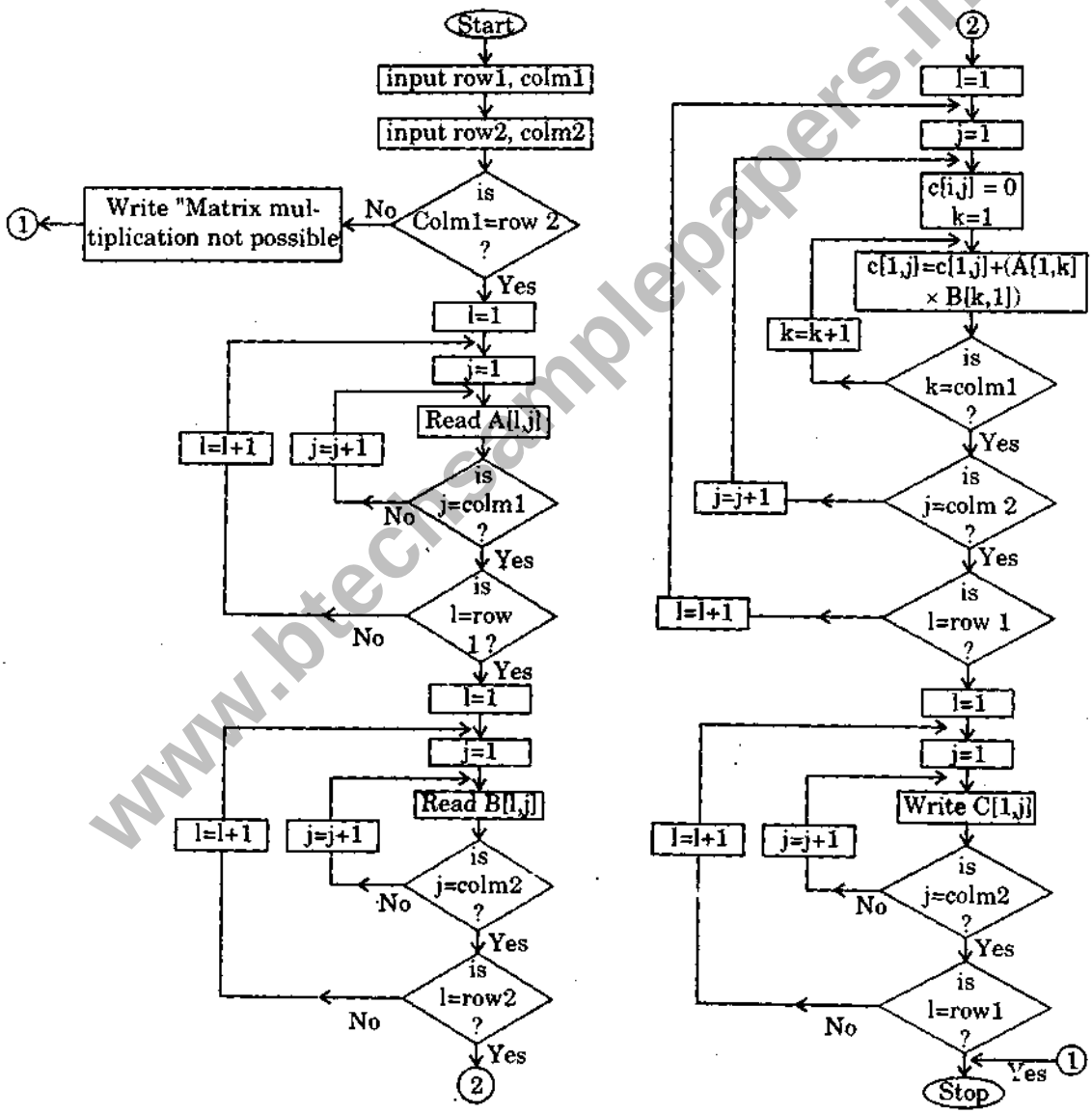
$$F(x) = x^1 - x^3 + x^5 - x^7 + \ldots$$

**Ans.**
```
# include <stdio.h>
# include <conio.h>
# include <math.h>
void main ( )
{
int x, s = 0, l, n;
clrscr ( );
pointf ("Input the value of nth term");
scanf ("%d", & n);
for (i = 0; i < =n; i++)
{
if (i % 2 = = 0)
{
}
else
{
s = s + pow (x, i);
```

```
}
}
printf ("sum = %d", s);
getch ( );
}
```

**Q. 5. (e) Draw flow chart and write a program in C to print the multiplication of two matrices A and B of size N × N.**

**Ans. Flow chart :**

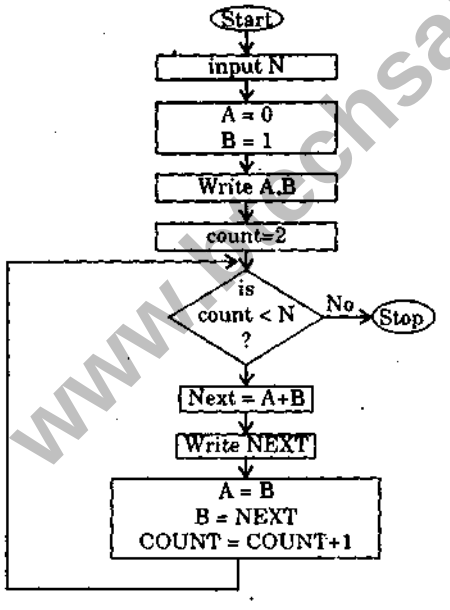


```
void main ( )
{
int m1 [10] [10], i, j, k, m2 [10] [10], add [10] [10], mult [10] [10], r1, c1, r2, c2;
printf ("Enter number of rows and columns of first matrix MAX 10\n");
printf ("Enter number of rows and columns of first matrix MAX 10\n");
scanf ("%d%d", &r1, &c1);
printf ("Enter number of rows and columns of second matrix MAX 10\n");
scanf ("%d%d", &r2, &(c2);
if (r2 = = c1)
{
printf ("Enter rows and columns of first matrix\n");
printf ("Row wise\n");
for (i = 0; i < r1; i++)
{
for (j = 0; j<<1; j++)
scanf ("%d, &m1 [i] [j]);
}
printf ("You have entered the first matrix as follows :\n");
for (i = 0; j < c1; j++)
printf ("%d\ t", m1 [i] [j]);
printf ("\n");
}
printf ("Enter rows and columns of second matrix\n");
printf ("Again row wise\n");
for (i = 0, i<r2; i++)
{
for (j=0; j< c2; j++)
scanf ("%d", & m2 [i] [j];
]
printf ("You have entered the second matrix as follows.:\n");
for (i=0; i<r2; i++)
{
for (j=0; j<c2; j++)
printf ("%d\t", m2 [i] [j]);
printf ("\n");
}
if (r1 = = r2 && c1 = = c2)
{
printf ("Now we add both the above matrix\n");
printf ("The result of the addition is as follows;\n");
for (i=0; i<r1; i++)
{
for (j=0; j<<1; j++)
{
add[i] [j]= m1 [i] [j] + m2[i] [j];
printf ("%d\t", add[i] [j]);
}
printf ("\n");
}
}
else
{
{printf ("Now we multiply both the above matrix\n");
printf ("The result of the multiplication is as follows :\n");
/* a11 × A11 + a12 × A21 + a13 × A31 + a11 × A12 + a12 × A22 + a13 × A32 + a11 × A13 + a12 × A23 + a13 × A33*/
for (i=0; i<r1; i++)
{
for (j=0; j<c2; j++)
{
mult [i] [j] = 0;
for (k=0; k<r1; k++)
{
mult[i] [j] + =m1[i] [k] * m2 [k] [j];
/*mult[0] [0] = m1 [0] [0] * m2 [0] [0] + m1[0] [1]*
m2[1] [0} + m1[0] [2] * m2[2] [0]; */
```

```
}
printf ("%d\t", mult[i] [j]);
}
printf ("\n");
}
getch ( );
}
else
{
printf ("Matrix multiplication cannot be done");
}
}
```

**Q. 5. (f) Draw flowchart and write a program in C to calculate the sum of Fibonacci series upto 100 terms.**

**Ans. Flowchart :**




WAP for the sum of Fibonacci series upto IDO terms

```
# include <stdio.h>
void main ( )
{
int n;
int a = 0, b = 1, c = 0;
printf ("Enter the Nth number required of Fibonacci series");
scanf ("%d", &n);
while (n! = 0)
{
c = a+b;
a = b;
b = c;
– – n;
}
printf ("Result is %d", c);
}
```

**Q. 5. (g) Using dynamic memory allocation, write a program in C to accept element of a matrix of size 3 × 3 and print transpose of it.**

**Ans.**

```
main ( )
{
int a[3] [3], b[3] [3], c[3][3], i, j, k;
printf ("Enter the elements of first matrix/n");
for (i=0; i<3; i++)
{
for (j=0; j<3; j++)
{
scanf ("%d", &a [i] [j]);
}
}
printf ("Enter the elements of second matrix");
```

```
for (i=0; i<3; i++)
{
for (j=0; j<3; j++)
{
scanf ("%d", &b[i] [j]);
}
}
for (i=0; i<3; i++)
{
for (j=0, j<3, j++)
{
c[i] [j] = 0;
}
}
for (i = 0; i < 3; i++)
{
for (j=0; j<3; j++)
{
for (k=0; k<3; k++)
{
c[i] [j] = c[i] [j] + a[i] [k] * b[k] [j];
}
}
}
print ("\n");
}
}
```