Important Instructions to examiners:
1) The answers should be examined by key words and not as word-to-word as given in the model answer scheme.
2) The model answer and the answer written by candidate may vary but the examiner may try to assess the understanding level of the candidate.
3) The language errors such as grammatical, spelling errors should not be given more Importance (Not applicable for subject English and Communication Skills.
4) While assessing figures, examiner may give credit for principal components indicated in the figure. The figures drawn by candidate and model answer may vary. The examiner may give credit for any equivalent figure drawn.
5) Credits may be given step wise for numerical problems. In some cases, the assumed constant values may vary and there may be some difference in the candidate's answers and model answer.
6) In case of some questions credit may be given by judgement on part of examiner of relevant answer based on candidate's understanding.
7) For programming language papers, credit may be given to any other program based on equivalent concept.

Q.1.   **Attempt any TEN of the following:**   **20M**
     a) **List the formatted input and output instructions.(each 1M)**
         printf() is a predefined standard C function for printing the output on standard output device such as screen (console) and printer.
         scanf() is used to take input from keyboard by the user.

     b) **Give the syntax of do-while statement.(2M)**
         Syntax:
         do
         {
         body of the loop
         }
         while(test-condition);

     c) **Syntax to declare two Dimensional array(2M)**
         The array which is used to represent and store data in a tabular form is called as two dimensional array. Such type of array is specially used to represent data in a matrix form.
         Declaration and initialization of two dimensional array:

             Syntax :-
             Data type arrayname [row size] [column size];

d) Differentiate between call by value and call by reference(Any two points, 2M)

Call by value	Call by reference
1. When function is called by passing values then it is call by value.	1. When function is called by passing address of variable then it is called as call by reference.
2. Copy of actual variable is created when function is called.	2. No copy is generated for actual variable rather address of actual variable is passed.
3. In call by value, memory required is more as copy of variable is created.	3. In call by reference, memory required is less as there is no copy of actual variables.
4.Example:- Function call - Swap (x,y); Calling swap function by passing values.	4.Example:- Function call – Swap (&x, &y); Calling swap function by passing address.
5. Original (actual) parameters do not change. Changes take place on the copy of variable.	5. Actual parameters change as function operates on value stored at the address.

e) Define pointer and declaration of pointer.(Define 1M, declaration 1M)
Pointer is a variable which stores the address of another variable.
Declaration of pointer:
data_type *ptr_name;

f) Write any four operators in C.(each ½ M)
1. Relational operators
2. Bitwise operators
3. Conditional operators
4. Logical operators
5. Assignment operators

g) State different data types in C.(any 4, each ½ M)
1. Char or Signed Char
2. Unsigned Char
3. Int or Signed int
4. Unsigned int
5. Short int or Signed short int
6. Unsigned short int
7. Long int or signed long int
8. Unsigned long int
9. Float
10. Double

h) Write output of following program.(2M)

```
void main()
{
int i=10,*p;
p=&i;
printf(value is":-,*p);
}
```

Output: 10

i) Explain pre-increment and post increment operator.(each 1M)

Pre increment will get incremented by 1 in any expression first and then remaining operators will work.

Eg: ++I;

Post increment will get incremented after the expression gets over.

Eg: i++;

j) Give syntax of switch-case statement.(2M)

The switch case statement:
Syntax:

```
switch(expression)
{
case value-1:
block-1;
break;

case value-2:
block-2;
break;
……..
……..
default:
default-block;
break;
}
statement-x;
```

k) Give syntax of strlen () function.(2M)

strlen() function counts and returns the number of characters in a string.

syntax: n=strlen(string);

where n is an integer variable, which receives the value of the length of the string.

l) Explain initialization of structure.(2M)
Initialization of structure
Syntax:
struct structure_name variable = { list of values};
Example(optional)
struct book
{
  int book_id;
  char title[20];
  char author[20];
  float price;
}b;

struct book b={2,"PIC","Balgurusamy",450.00};

Q.2.   **Attempt any Four of the following:**   **16M**

a) Write program to display following.(2M – correct logic, 2M – correct syntaxes)
****
***
**
*
Note: any other logic can be considered

```
#include<stdio.h>
#include<conio.h>
void main()
{
int i;
for(i=0; i<=3; i++)
{
for (j=3; j>=i; j--)
{
printf("*");
}
printf("\n");
}
getch();
}
```

b) What is array? How to declare and initialize array? Explain.
An array is a fixed size sequence collection of elements of similar data types which share a common name. It is simply grouping of same type of data which can be accessed by a common name**.   (2M)**
Declaration of Array: **(1M)**

Considering array of 5 integer elements:
int a[5];
Initialization of an array of 5 elements :
An array can be either initialized at design time or runtime.
1. Design time initialization **(1M)**

(Considering an array of 5 integers)
int num[5]={1,3,4,10,12};

2. Run time initialization
This requires some loop statement for repeating input to all locations of an array.
Eg.int num[5];
int i; for(i=0; i<5; i++)
 {
scanf("%d", &num[i]);
}

- Any other data type of array can be considered.
- Any one method out of two for initialization can be considered.

c) **Explain any two string functions.(Any 2, each 2 marks)**
strcat():- This function joins two strings together. The contents of 2nd string argument are added at the end of first string.

 Syntax: strcat (string1, string2);
e.g. char word[20], word1[5];
 word="ABC";
word1="Company";
strcat(word1,word2);
Output: word1=ABCCompany
 word2=Company 2.
2.strcmp():- This function compares two strings identified by arguments and returns zero if both strings are equal, otherwise it returns the difference between ASCII values of first non matching character pair from the strings.
 Syntax: strcmp (string1, string2);
strcmp ("there", "their")= difference between r & e
3. strlen():- It can be used to get the length of the string. It uses string argument which is name of character array. It returns an integer showing no. of characters from the string excluding last'\0' (null character).

Syntax:
intVariable_name=strlen(string name);

4. strcpy() :- This function accepts two string arguments and as a result contents of string2 are copied in string1.
 Ex. char s1[20]={"ABC"};
char s2[8]={"Company"};
strcpy(s1,s2)
 Output: s1=Company s2=Company

d) Explain the recursive functions with example. (explanation - 2 M, example 2M)

Recursion is a process of function calling itself again and again. A recursive function contains function call to itself in the body of function. Recursive function can be effectively used to solve problems where solution is expressed in terms of successively applying the same solution to subsets of problem. When recursive function is written, an if statement should be there in the body of function to force the function to return to calling function without the recursive call being executed. If statement is not included then function will never return.

Ex: recursive function for calculating factorial of a no.

```
int factorial ( int no)
        {

                if ( no==1)
                return 1;
                else
                return (no * factorial ( no -1));

        }
```

Any other recursive function example can be written & considered.

e) Distinguish between variable and constant.(any four point ,4M)

Variable	Constant
Value of Variable can be changed at any time throughout the program.	Constant variable are the variables whose value once assigned cannot be changed throughout the program.
Variable are declared along with its data type	Variable can be declared using 'const' key word.
Syntax:<data type><variable name>	**Syntax:** const<data type> <variable name>
Eg: int a;	**Eg: const float pi=3.14**

f) Explain '*' and '&' operators used with pointers.(explanation 2M, example 2M)

& (ampersand) operator can be used to determine address of any variable.

*(asterisk) operator can be used to display value of the pointer variable that is original value

```
void main()
{
int i=10,*p;
p=&i;
printf(value is":-,*p);
}
```

Output: 10

Q.3.   Attempt any four of the following: 16M

a) Write program to reverse the number 1234 (i.e. 4321) using function.
(2M – correct logic, 2M correct syntax)

```
#include<stdio.h>
#include<conio.h>
void rev(int no);
```

```
void main()
{
int no;
printf("\n Enter number:");
scanf("%d",&no);
rev(no);
getch();
}
void rev(int no)
{
int sum=0,rem;
while(no>0)
{
rem=no%10;
no=no/10;
sum=sum*10+rem;
}
printf("\nsum=%d",sum);
}
```

b) **Explain structure in C with suitable example.(structure definition & explanation 2M, syntax of declaration 1M, example 1 M)**

A structure is a collection of one or more variables of same or different data types grouped together under a single name.

Declaration of structure:
Syntax:
```
struct structure _name
{
        Data_type  member1;
        Data_type  member2;
        Data_type  member n;
}variable;
```

Example
```
struct book
{
 int book_id;
 char title[20];
 char author[20];
 float price;
}b;
```

c) **Write a program to find the length of the string.**
(2M – correct logic, 2M correct Syntax)
```
#include<stdio.h>
#include<conio.h>
#include<string.h>
void main()
{
int l,i=0;
char str1[10];
clrscr();
```

```
printf("\n Enter string:");
gets(str1);
while(str1[i]!='\0')
{
i++;
}
printf("%d",i);
getch();
}
```

or
```
#include<stdio.h>
#include<conio.h>
#include<string.h>
void main()
{
int l;
char str1[10];
clrscr();
printf("\n Enter string:");
gets(str1);
l=strlen(str1);
printf("%d",l);
getch();
}
```

d) **Explain use of if else statement(use 1M, syntax 1M, example 2M)**

If else statement is used for decision making. It is used to control flow of execution of statement.

Syntax:-   if(test expression)
```
                {
                   True block statement
                }
                else
                {
                False block statement
                }
```

If the test expression is true then the true block statement, immediately following the if ,statements are executed.

If the test expression is false then the false block statement, immediately following the else, statements are executed.

Example:
int num;
num=2;
```
if(num<0)
{
printf("num is negative");
}
else
{
printf("num is positive");
```

}
num is greater than 0 so false block statement will be executed.

e) Write a program to accept ten member ir and print average of it.
(2M – correct Logic, 2M correct syntax)

```
#include<stdio.h>
#include<conio.h>
void main()
{
int a[10],avg,sum=0,i;
clrscr();
printf("\n Enter number");
for(i=0;i<10;i++)
scanf("%d",&a[i]);
for(i=0;i<10;i++)
sum=sum+a[i];
avg=sum/10;
printf("Avg=%d",avg);
getch();
}
```

f) Write program to find whether a number even or odd.
Note: any other logic can be consider
(2M – correct Logic, 2M correct syntaxes)

```
#include<stdio.h>
#include<conio.h>
void main()
{
 int no;
clrscr();
 printf("Enter a number:");
scanf("%d",&no);
if(no%2==0)
{
printf("\nNumber is even");
}
 else
{ printf("\nNumber is odd");
 }
 getch();
 }
```

Q.4.   Attempt any four of the following: 16M

a) State use of increment and decrement operators with examples.(use of operator 1M each, example 1M each)

Increment operator:
Use: Increment operator (++) is an unary operator. It operates on one operand.
It is used to increment value of a variable by 1.
Example:

```
int a=6;
a++;
        printf("\n%d",a);
```

In above example initially value of a is 6.due to increment operator (++) value of a will be incremented by 1. So output value of a will be 7.

Decrement operator:
Use: Decrement operator (--) is an unary operator. It operates on one operand.
It is used to decrement value of a variable by 1.
Example:
```
int a=6;
a--;
printf("\n%d",a);
```

In above example initially value of a is 6.due to decrement operator (--) value of a will be decremented by 1. So output value of a will be 5.

b) **Explain initialization of one dimensional of array.(explanation 2M; example 2M)**
Compile time initialization: an array is initialized when it is declared.
Syntax initialization of one dimensional of array:

datatype array_name[size]={list of valuses};

Example: int number[3]={1,2,3};
The above statement will declare variable number as an array of size 3 and will assign specified values to each element.

Run time initialization: an array is initialized explicitly at run time that is after declaration initialization is done separately**.**
Example:
Int a[3], I;
For(i=0; i<3; i++)
Scanf("%d",&a[i]);

(any one type of initialization can be considered)

c) **Explain pointer arithmetic with example.**
(02-marks for explanation, 02-marks for example)
Explanation:
Pointer arithmetic is concerned with the arithmetic operations with the pointers. Like basic operation +, -, *, / can be done using pointer notation.
Some of the following operations are possible:
e.g. add = *p1 + *p2
y= *p1 -*p2
x= *p1 / *p2

Example:
```
#include<stdio.h>
#include<conio.h>
void main()
{
int a=10,b=2,sum,mul;
int *p1,*p2;
clrscr();
p1=&a;
p2=&b;
sum=*p1+*p2;
mul=*p1**p2;
printf("\nAddition=%d\nMultiplication=%d",sum,mul);
getch();
}
```

d) **Explain pointer expressions. Explain meaning of (*pl)++ with example.(pointer expression 2M, meaning of (*pl)++ 2M)**

A pointer expression contains pointer variable as an operand and arithmetic operators such as +, -, *, /

Example:
x=*p1+*p2

(*pl) :–pl is a pointer variable.*pl indicates value stored at the memory address present inside pointer pl.

(*pl)++ :- This statement increments value, stored at the memory address pointed by pointer pl, by 1.

```
#include<stdio.h>
#include<conio.h>
void main()
{
 int a,*pl;
 a=2;
 pl=&a;
 printf("\n%d",(*pl)++);-> displays output as 2
 printf("\n%d",(*pl));-> displays output as 3
 getch();
}
```

e) **Differentiate while and do while statement.**
Any other valid difference can be considered. Each difference 1 M (4 Diff.)

While loop	do while loop

1. Syntax: while (expression)
       {
         Statements
       }
	syntax: do
	       {
	         Statements
	       }
	     while (expression);

2. while loop checks the condition first and then	do while loop first executes statements and then checks the condition.

executes statements.	
3. If expression is true then only statements inside block are executed otherwise loop terminates.	If expression is false then also at least once statements inside block are executed.

4. Example: a=1;
while (a<0)
{
  printf ("%d",a);
  a = a+1;
}
	a = 1;
	do
	{
	  printf("%d",a);
	  a = a+1;
	}while(a<0);

f) **What is function prototype? Explain with example. (function prototype explanation 2M, example 2M)**

Function prototype is a function declaration before it is invoked. It consists of function return type, function name, parameter list and terminating semicolon.

Syntax:

Function_type function_name(parameter list);

function_type specifies the data type of value that the function is expected to return to the calling function.

Function name specifies a unique identity of the defined function in the program.

Parameter list declares the variables that will receive data sent by calling function. They act as input to the function.

Example:

void display(int a)

{

printf("%d",a);

}

Above function does not return any value as its function type is void. Name of the function is display. This function accepts 1 int value from the calling function and stores it in variable a. Inside the function body value of a is display on the screen.

Q.5. **Attempt any Four of the following:**   **16M**

a) **Write to read two strings and find whether they are equal or not equal**

(Logic 2 marks syntax 2marks)

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
void main()
{
char st1[20],st2[20];
printf("enter string 1");
scanf("%s",st1);
printf("enter second string");
scanf("%s",st2);
if(strcmp(st1,st2)==0)
printf("\nboth strings are equal");
else
printf("\nstrings are not equal");
}
```

b) Describe use of break and continue statement with example.
(Explanation 2 M, Examples – 2Marks)
Break: It breaks the execution of the loop which allows exiting from any loop or switch, such that break statement skips the remaining part of current iterations of the loop.
Syntax: break;
Continue: It is used when it is required to skip the remaining portion of the loop without breaking loop it will transfer control directly to next iteration
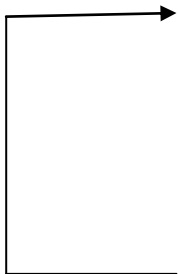Syntax: continue;
Note:- In example any other correct example is also considerable)

```
do
{
Statement 1
Statement 2
.
.
.
if(conditional statement )
break;
Statement n
.
.
.

} while (conditional statement);
Statement y
```

```
do
{
Statement 1
Statement 2
.
.
if(conditional statement )
continue;
statement n
.
.
.
.

} while (conditional statement);
```

in given program sequence if '**break** 'executes then execution control will jump out of loop & next statement after loop i.e. statement y will be executed.
in given program sequence if' **continue** 'executes then execution control will skip remaining statements of loop & will start next iteration of loop

c) **Write program to determine the string is palindrome or not.**

(Logic 2 marks syntax 2marks)
note:- character by character comparison logic is also valid)
#include<stdio.h>
#include<string .h>
#include<conio.h>
void main ()
{
char str1[20],str2[20];
clrscr();
printf("enter string which to be checked");
scanf("%s",str1);
strcpy(str2,str1);
strrev(str2);
if(strcmp(str1,str2)==0)
        printf("string is palindrome ");
else
        printf("string is not palindrome ");
getch();
}

d) **Explain concept of Array of characters with example.**
(Explanation 2 marks example 2 marks)
Character array is collection of values arranged in sequential manner in memory. Contents of character array can be accessed or filled as one character at a time or entire string at once.
Declaration of character array:- character array declaration can be done as follow
char array_name[size_of_array];
here size_of_array specify maximum no of elements which we can store in array. In compile time initialization size field can be kept blank .If character array stores string then it is ended with '\0'.
Initialization of character array :- character array can be initialized at compile time or run time .
Accessing character array: - character array elements can be accessed character by character using format specifier %c & as whole string using format specifier %s .
Ex: char myarry[10]={'a','e','i','o','u'};
here character array myarr of size 10 is declared & initialized with members 'a','e','i','o','u'
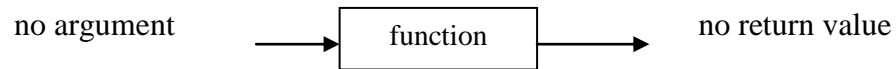
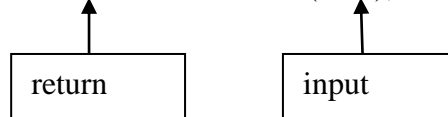| a	e	i	o	u
1000	1001	1002	1003	1004

e) **Explain No-Argument-No-Return value function type with example.**
(Explanation 2 marks example 2 marks)
In this category of function, function does not take any input & it also does not provide any return value to caller of function.

no argument → function → no return value

syntax of declaration:-
void function_name(void);

return   input

example:-
#include<stdio.h>
void myfun(void);
void main()
{
myfun()
}
void myfun(void)
{
printf("hello");
}
In given example function myfun() does not take any input or provide any return value

f) **Explain problem-solving techniques in detail. (List 1 Mark, explanation 3 Marks)**
algorithm
flowchart
Pseudo-code
Alogrithm :- a process or set of rules to be followed in calculations or other problem-solving operations, especially by a computer.
Flowchart:- a graphical representation of a computer program in relation to its sequence of functions and operations. (as distinct from the data it processes).
Pseudocode :-is an informal high-level description of the operating principle of a computer program or other algorithm.

Q.6.   **Attempt any Four of the following: 16M**
a) **Write program to sum all the odd numbers between 1 to 20.**
(Logic 2 marks syntax 2marks)

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

```
{
int sum=0,i;
for(i=1;i<=20;i++)
{
            if(i%2==1)
            sum=sum+i;
}
printf("sum of odd no's between 1 to 20 is %d",sum);
}
```

b) Write syntax of declaration of function with example.
(Syntax 2 marks example 2 marks)
Function declaration (function prototyping):- this statement specifies to compiler details of user defined functions
The details specified by declaration statement are return type, name of function, parameter list with their data type

Syntax :-
Return type name_of_function(data type para-1, data type para-2……., data type para-N);
Ex.

```
#include<stdio.h>
void area(int radius);
void main()
{
int r;
printf("enter radius%d",&r);
scanf("%d",&r);
area(r);
}
void area(int radius)
{
float a;
a=3.14*radius*radius;
printf("area is %f",a);
}
```

c) Write program to accept 10 numbers in array and arrange them in ascending order.

(Logic 2 marks syntax 2marks)

```
#include<stdio.h>
#include<conio.h>
void main()
{
intarr[10],i,j,temp;
clrscr();
printf("Enter array elements:");
for(i=0;i<10;i++)
 {
scanf("%d",&arr[i]);
```

```
 }
printf("\n\nArray elements are:");
for(i=0;i<10;i++)
 {
printf("%d ",arr[i]);
 }
for(j=0;j<10;j++)
 {
for(i=0;i<10;i++)
  {
if(arr[i+1]<arr[i])
   {
temp=arr[i];
arr[i]=arr[i+1];
arr[i+1]=temp;
     }
     }
     }
printf("\n\nArray elements in ascending order are:");
for(i=0;i<10;i++)
 {
printf("%d  ",arr[i]);
 }
getch();
}
```
Note : Any other sorting logic can be considered.

d) **Write program to give the factorial of a number.**

   (Logic 2 marks syntax 2marks)

```
#include<stdio.h>
#include<conio.h>
void main()
{
inti,fact=1,no;
printf("enter no of which you wants to  factorial");
scanf("%d",&no);
for(i=1;i<=no;i++)
        {
        fact=fact*i;
        }
printf("factorial is  %d",fact);
}
```
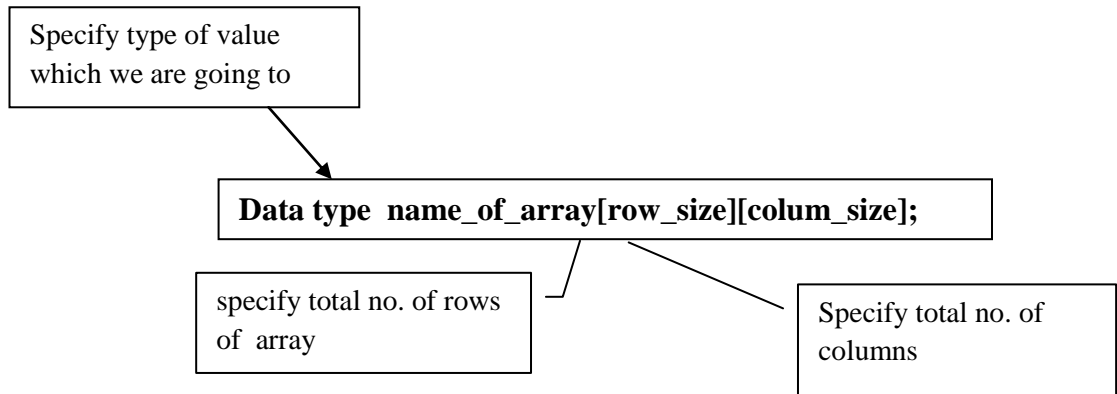
e) Explain two dimensional array with example.(Explanation 2 marks example 2 marks)

It is collection of constants of same data type arranged in sequentially in column & row manner.
Declaration of Two dimension array: - array declaration is statement which is used for creating array. Array is declared similar to other primary variable. 2-D array declaration is done as follow

Specify type of value which we are going to

Data type name_of_array[row_size][colum_size];

specify total no. of rows of array

Specify total no. of columns

ex. int arr[3][2] - here array of type integer having 3 rows and 2 column is declared.
Initializing array (one dimension array):- array initialization means inserting values in array. There are two methods of array initialization
1) Compile time initialization
2) Run time array initialization

<table>
<tr><td>intmy_array[3][2]={
            {1,2},
            {3,4},

            {5,6}
                };</td><td colspan="2">column -0   column-1</td></tr>
</table>

	column -0	column-1
row - 0	1	2
row-1	3	4
row-2	5	6

f) Explain any two logical operators with example.(explanation 2 marks example two marks)note:- any two operators can be considered)

Operator	Syntax	Working	Example
&&	Argument 1 && argument2	This operator will check logical value of argument 1 and argument 2 And if logical values of both argument is true then it will give output 1(true) otherwise will return zero.	a=5 ,b=4,c=3,d=1 (a>b)&&(c>d) This will give output1 as both of its arguments a>b and c>d are having logical value 1
||	Argument 1 || argument2	This operator will check logical value of argument 1 and argument 2 And if logical values of at least one argument is true then it will give	a=5 ,b=4,c=3,d=1 (a<b)||(c>d) This will give output1 as its arguments c>d is having logical value 1

		output 1(true) otherwise will return zero.	
!	!argument	This operator will compliment (negate) value of it argument .that means it will convert any non zero value into 0 and will convert 0 into 1	a=5 !a Output->0 As a is having nonzero value hence will be converted into 0