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.
8) As per the policy decision of Maharashtra State Government, teaching in English/Marathi and Bilingual (English + Marathi) medium is introduced at first year of AICTE diploma Programme from academic year 2021-2022. Hence if the students in first year (first and second semesters) write answers in Marathi or bilingual language (English +Marathi), the Examiner shall consider the same and assess the answer based on matching of concepts with model answer.

Q. No.	Sub Q.N.	Answer	Marking Scheme
Q.1		**Attempt any <u>FIVE</u> of the following:**	**10-Total Marks**
	a)	**Define the terms** **i) Compiler** **ii) Interpreter**	**2M**
	Ans:	**<u>Compile</u>**r Compiler scans the entire program and translates the whole of it into machine code at once. **<u>Interpreter</u>** Interpreter translates just one statement of the program at a time into machine code.	**Defination-2M**
	b)	**Give syntax of if-else.**	**2M**

	Ans:	**Syntax of if-else statement** :
		if (test expression)
		{
		True-block statement (s)
		}
		else
		{
		False-block statement (s)
		}
			Syntax -2M

	c)	**Give use of break statement**	**2M**
	Ans:	**Use of break statement:** The break is a keyword in C which is used to bring the program control out of the loop. The break statement is used inside loops or switch statement. The break statement breaks the loop one by one, i.e., in the case of nested loops, it breaks the inner loop first and then proceeds to outer loops. The break statement in C can be used in the following two scenarios: 1. With switch case 2. With loop	**Use-2M**
	d)	**Give any two math function.**	**2M**

	Ans:	**Math function:**
		sqrt()
		pow()
		floor()
		round()
		ceil()
		sin()
		cos()
		cosh()
		exp()
		tan()
		tanh()
		sinh()
		log()
		log10()
		trunc()
			**Any
			two
			functi
			o
			ns 1M
			each**

	e)	**Define array and its syntax.**	**2M**
	Ans:	**An array** is a collection of similar type of elements. **Syntax to declare an array:** data type arr_name[size] ; In the above syntax, data type specifies type of data that can be stored in an array. arr_name specify name of array. size specify number of values that can be stored inside an array. Example: int arr[5] = {10, 20, 5, 3, 55};	**Defination 1M** **Syntax 1M**
	f)	**State any four features of pointer**	**2M**
	Ans:	**Features of pointer:** 1. Pointers used to access the address of the variable. 2. Pointers increase the execution speed of program. 3. Pointers are an important concept in data structures. 4. Pointers are used for dynamic memory allocation. 5. Pointers makes possible to return more than one value in functions 6. Pointer enables us to access variables that are declared outside the functions 7. Strings and arrays are more efficient with pointers	Any four points 2M
	g)	**Define structure.**	**2M**
	Ans:	**A structure** is a collection of one or more variables of same or different data types grouped together under a single name.	**Defination-2M**
Q.2		**Attempt any THREE of the following:**	**12-Total Marks**
	a)	**Explain algorithm & flowchart.**	**4M**
	Ans:	**Explanation Of Algorithm:** An algorithm is a finite sequence of instructions, a step-by-step procedure for solving a problem from the beginning. This is the first step of the procedure. An algorithm includes calculations, reasoning and data processing. Algorithms can be presented by natural languages, pseudo code etc. To write the algorithm one first must know how to solve the problem. **Explanation Of Flowchart:** A flowchart is the graphical or pictorial representation of an algorithm with the help of different symbols, shapes and arrows in order to demonstrate a process or a program. Flowcharting uses symbols that have been in use for a number of years to represent the type of operations and/or processes being performed. The standardised format provides a common method for people to visualise the process.	**Explanation-4M**

b) **Write a program to find whether given number is positive, negative or zero.** **4M**

Ans:

Program : To find whether given number is positive, negative or zero.

```
#include <stdio.h>
int main()
{
    int A;
      printf("Enter the number A: ");
    scanf("%d", &A);
      if (A > 0)
        printf("%d is positive.", A);
    else if (A < 0)
        printf("%d is negative.", A);
    else if (A = = 0)
        printf("%d is zero.", A);
      return 0;
}
```

Correct logic : 4M

c) **Explain strlen() & strcpy() function with example.** **4M**

Ans:

strlen( ):- This function is used to find the length of a string. It counts the number of characters comprising the string.

Syntax : strlen(string1);

Example:

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
void main() {
char str[] = "mystring";
int len=0;
clrscr();
len=strlen(str);
printf("Length of string is :%d",len);
getch();
}
```

strcpy( ):- This string function is used to copy the content of one string to the other string.

Syntax: strcpy(string1,string2);

string1 and string2 are character arrays.

		When strcpy( ) function executes the contents of string2 are copied into string1.
		Example:
		#include<stdio.h>
		#include<conio.h>
		#include<string.h>
		void main() {
		char source[]="mystring";
		char dest[10];
		clrscr();
		printf("%s%s",source,dest);
		strcpy(dest,source);
		printf("\n%s %s",source, dest);
		getch();
		}
			Explanation-4M

	d)	**Write a program to add two numbers using the call by value.**	**4M**

	Ans:	**<u>Program : To add two numbers using the call by value.</u>**
		#include<stdio.h>
		int main() {
		int num1, num2, res;
		printf("\nEnter the two numbers : ");
		scanf("%d %d", &num1, &num2);
		//Call Function Sum With Two Parameters
		res = sum(num1, num2);
		printf("nAddition of two number is : ");
		return (0);
		}
		int sum(int num1, int num2) {
		int num3;
		num3 = num1 + num2;
		return (num3);
		}
			Correct logic:4 M

Q.3 **Attempt any THREE of the following:** **12-Total Marks**

a) Enlist any four types of arithmetic operator used in C and give one example of each. **4M**

Ans:

The following table shows all the arithmetic operators supported by the C language. Assume variable A holds 10 and variable B holds 20 then −

Sr. No.	Operator	Name	Description	Example
1	+	Addition	Adds two operands.	A + B = 30
2	-	Subtraction	Subtracts second operand from the first.	A − B = 10
3	*	Multiplication	Multiplies both operands.	A * B = 200
4	/	Division	Divides numerator by de-numerator.	B / A = 2
5	%	Modulus	Modulus Operator and remainder of after an integer division.	B % A = 0
6	++	Increment	Increment operator increases the integer value by one.	A++ = 11
7	– –	Decrement	Decrement operator decreases the integer value by one.	A-- = 9

Enlisting (Any 4)

2M

Example (Any 4)

2M

b) Write a program to add two matrices of 3*3 size, store addition in third matrix for given data element. **4M**

Ans:

(Note: Any other correct logic shall be considered).

```
#include <stdio.h>
#include <conio.h>
void main()
{
int a[3][3],b[3][3],c[3][3],i,j;
clrscr();
printf("\n Enter first matrix");
for(i=0;i<3;i++)
{
for(j=0;j<3;j++)
{
scanf("%d",&a[i][j]);
}
}
```

Deceleration of variables 1M

Input matrices 1M

```
printf("\n Enter 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]=a[i][j]+b[i][j]; }
}
printf("\n Addition:\n");
for(i=0;i<3;i++) {
for(j=0;j<3;j++) {

printf("%d\t",c[i][j]);

}

printf("\n");

} getch();

}
```

Calculating addition 1M

Display addition-1M

c) **Write a program to access the array elements using pointer.** **4M**

Ans:

(Note: Any other correct logic shall be considered).

```
#include <stdio.h>
int main() {
   int data[5];
   printf("Enter elements: ");
   for (int i = 0; i < 5; ++i)
{
   scanf("%d", data + i);
   printf("You entered: \n");
}
for (int i = 0; i < 5; ++i)
       printf("%d\n", *(data + i));
   return 0;
}
```

Deceleration of variables -2M

Correct Logic-2M

	d)	**Explain how structure can be initialized with suitable example.**	**4M**

Ans:

Declaration of structure:-

```
struct structure_name
 {
data_type member 1;
data_type member 2;
. . .
data_type member n; }

 structure variable 1, structure variable 2,..., structure variable n;
```

Example:-

```
struct student
 {
int rollno;
char name[10];
}s1;
```

Initialization:-

```
struct student s ={ 1,"abc" };
```

OR

The structure variable contains two members as rollno and name. The above example initializes rollno to 1 and name to "abc".

Program:-

```
#include<stdio.h>
#include <conio.h>
struct college                    //declaration of structure
 {
int collegeid;
char collegename[20];
};

void main()
{
struct colleget s={123,"Polytechnic"};   //initialization of structure
clrscr();
printf("\n College id=%d",s.collegeid);
printf("\n College name=%s",s.collegename);
getch();
}
```

Declaration of structure- 2M

Example- 2M

Q.4		Attempt any THREE of the following:	12-Total Marks
	a)	**Explain conditional and bitwise operator with example.**	4M

Ans:

Conditional Operator (Ternary Operator):

It takes the form "?:" " to construct conditional expressions The operator "? :" works as follows: exp1 ? exp2 : exp 3

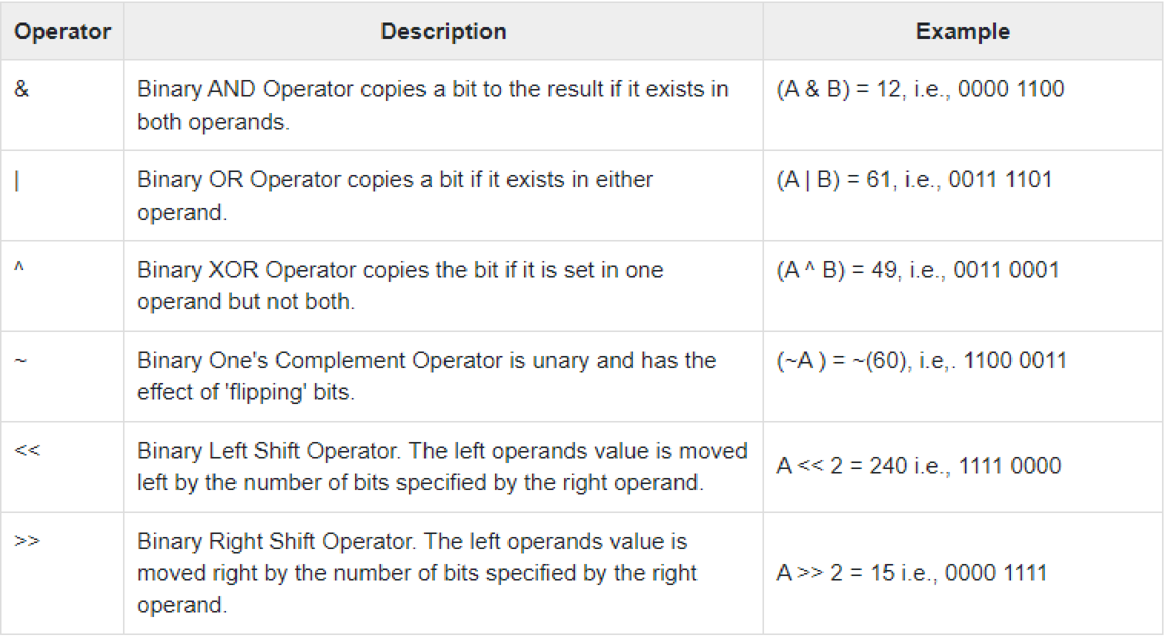
Where exp1, exp2 and exp3 are expressions.

exp1 is evaluated first, If it is true, then the expression exp2 is evaluated and becomes the value of the expression. If exp1 is false, exp3 is evaluated and its value becomes the value of the expression.

E.g.

```
int a=10,b=5,x;
x=(a>b) ? a : b;
```

BITWISE OPERATOR:



Operator	Description	Example
&	Binary AND Operator copies a bit to the result if it exists in both operands.	(A & B) = 12, i.e., 0000 1100
|	Binary OR Operator copies a bit if it exists in either operand.	(A | B) = 61, i.e., 0011 1101
^	Binary XOR Operator copies the bit if it is set in one operand but not both.	(A ^ B) = 49, i.e., 0011 0001
~	Binary One's Complement Operator is unary and has the effect of 'flipping' bits.	(~A) = ~(60), i.e,. 1100 0011
<<	Binary Left Shift Operator. The left operands value is moved left by the number of bits specified by the right operand.	A << 2 = 240 i.e., 1111 0000
>>	Binary Right Shift Operator. The left operands value is moved right by the number of bits specified by the right operand.	A >> 2 = 15 i.e., 0000 1111

Example:

Conditional operator

1M Explain

1M Example

and

Bitwise operator

1M Explanation

1M Example

```
#include <stdio.h>
main() {
  unsigned int a = 60;         /* 60 = 0011 1100 */
  unsigned int b = 13;         /* 13 = 0000 1101 */
  int c = 0;

  c = a & b;     /* 12 = 0000 1100 */
  printf("Line 1 - Value of c is %d\n", c );

  c = a | b;     /* 61 = 0011 1101 */
  printf("Line 2 - Value of c is %d\n", c );

  c = a ^ b;     /* 49 = 0011 0001 */
  printf("Line 3 - Value of c is %d\n", c );

  c = ~a;        /*-61 = 1100 0011 */
  printf("Line 4 - Value of c is %d\n", c );

  c = a << 2;    /* 240 = 1111 0000 */
  printf("Line 5 - Value of c is %d\n", c );

  c = a >> 2;    /* 15 = 0000 1111 */
  printf("Line 6 - Value of c is %d\n", c );
}
```

b) **Write a program to perform addition, subtraction, multiplication and division using switch case statement for given data.** **4M**

(Note: Any other correct logic shall be considered).

```
#include<stdio.h>
#include<conio.h>
void main()
{
int a,b,ch,add,sub,mul,div;
clrscr();
printf("\n1 for addition \n2 for substraction");
printf("\n3 for multiplication \n4 for division");
printf("\nEnter two numbers:");
scanf("%d%d",&a,&b);
printf("\nEnter your choice:");
scanf("%d",&ch);
switch(ch)
```

	Ans:	{
		case 1:
		add=a+b;
		printf("Addition of a & b=%d",add);
		break;
		case 2:
		sub=a-b;
		printf("Substraction of a & b=%d",sub);
		break;
		case 3:
		mul=a*b;
		printf("Multiplication of two numbers=%d",mul);
		break;
		case 4:
		div=a/b;
		printf("Division of two numbers=%d",div);
		break;
		default:
		printf("Invalid choice....");
		}
		getch();
		}
			Correct logic 2M
			Correct syntax 2M

	c)	**Explain strcat() strcmp() with a suitable example.**	**4M**
	Ans:	**<u>strcat() function:</u>** In C programming, strcat() concatenates (joins) two strings. It concatenates the source string at the end of the destination string. Syntax: **strcat(destination source, source string);** Example: strcat(str1,str2); returns abcdef in str1 and str2 remains unchanged. **<u>strcmp() function</u>** The strcmp function compares two strings which are passed as arguments to it. If the strings are equal then the function returns value 0 and if they are not equal the function returns some numeric value. Syntax: **strcmp(str1, str2);** Example: Then strcmp(str1,str2) returns 0 as both the strings are the same.	**Explanation -4M**

d) **Write a program to display hexadecimal, decimal, octal binary format of entered number.** **4M**

Ans:

```
#include <stdio.h>
 int main()
{
   int n, value=2567;
for(i=0;n>0;i++)
{
a[i]=n%2;
n=n/2;
}
printf("\nBinary of Given Number is=");
for(i=i-1;i>=0;i--)
{
printf("%d",a[i]);
}
   printf("Decimal value is: %d\n",value);
   printf("Octal value is: %o\n",value);
   printf("Hexadecimal value is (Alphabet in small letters): %x\n",value);
   printf("Hexadecimal value is (Alphabet in capital letters): %X\n",value);
       return 0;
}
```

Correct logic 2M

Correct syntax 2M

e) **Write structure DATE having members day, month, year & assign initial values to that structure.** **4M**

Ans:

```
struct date
{
    int month;
    int day;
    int year;
};
int main()
{
struct date d1;
   int monthMain, dayMain, yearMain;
   printf("Enter a Month: ");          //requesting user to input the month
   scanf("%d", &date.month);           //accepting the user input for month
   printf("Enter a Day: ");            //requesting user to input the day
   scanf("%d", &date.day);             //accepting the user input for day
   printf("Enter a Year: ");           //requesting user to input the year
   scanf("%d", &date.year);            //accepting the user input for year
   printf("the date you entered = %d / %d / %d days", d1.day, d1.month, d1.year);
return 0;
}
```

Correct logic 2M

Correct syntax 2M

Q.5		Attempt any TWO of the following:	12-Total Marks
	a)	**i) Differentiate between while() & do while().** **ii) Write a program to display even numbers in 1 to 100 using while()**	6M

Ans:

i)

	While( )	do While( )
1	Entry controlled loop	Exit controlled loop
2	Condition is checked first	Condition is checked last
3	Executes only if satisfies the Condition	Executes at least once even if the condition is not satisfied.
4	There is no semicolon at the end of while	The semicolon is compulsory at the end of do-while

5	**Syntax :**
	while(condition)
	{
	Code;
	}
		Syntax:
		do
		{
		Code;
		} while(condition);

Any 3 difference points 1M each

```
#include <stdio.h>
int main()
{
    int i, n;
      // Input upper limit of even number from user
    i=1;
   n=100;
    while(i<=n)
    {
        /* Check even condition before printing */
        if(i%2==0)
        {
            printf("%d\n", i);
        }
        i++;
    }
    return 0;
}
```

Correct Syntax : 1 ½ M,

Correct logic:1 ½ M

b) **Write a C program to count the number of character present in the entered text without using string function.** **6M**

Ans:

```
#include <stdio.h>
 void main()
{
    char string[50];
    int i, length = 0;
     printf("Enter a string \n");
    gets(string);
    /*  keep going through each character of the string till its end */
    for (i = 0; string[i] != '\0'; i++)
    {
        length++;
    }
    printf("The length of a string is the number of characters in it \n");
    printf("So, the length of %s = %d\n", string, length);
}
```

Correct Logic : 3M,

Correct Syntax: 3M

(Any other logic can be considered)

c) **Explain call by reference with suitable example.** **6M**

Ans:

Call by Reference: Inside the function, the address is used to access the actual parameter used in the call. It means the changes made to the formal parameters affect the actual parameters.

- Call by reference does not waste memory, as new variables are not created.
- Pass a pointer that contains the memory address of an object that gives access to its contents.
- Call by references works more efficiently than call by value.

Example :

```
#include <stdio.h>
void main()
{
        int a=10, b=12;
        swap (&a,&b);
        printf(“\n a=%d b=%d”, a,b);
}
swap(int * x, int * y)
{
     int t;
     t= *x;
     x=*y ;
    *y=t;
     printf(“\n a=%d b=%d” , x,y);
}
output:
a=12 b=10
a=10 b=12
```

Explanation : 3M, Example : 3M

(Any other example can be considered)

Q.6		**Attempt any TWO of the following:**	**12-Total Marks**
	a)	**Write a program to find highest marks in a class of n students using array.**	**6M**

Ans:

```
#include <stdio.h>
#include< conio.h>
void main()
{
  int n, i, marks[100],highest;
  clrscr();
  printf("Enter number of students: ");
  scanf("%d", &n);
  printf("Enter marks of %d students: \n", n);
  for(i=0; i<n; i++) {
        printf("Marks of Student %d: ", i+1);
        scanf("%d", &marks[i]);
  }
  highest = marks[0];
  for(i=1; i<n; i++) {
    if(marks[i] > highest)
      highest = marks[i];
  }
  printf("Highest marks  acquired by student is: %d", highest);
  getch();
}
```

Correct Logic : 3M, Correct Syntax : 3M

(Any other logic can be considered)

b) **Explain scope of variable with suitable example.** **6M**

Ans:

Scope of variable is defined in which part of the program the variable is actually available and beyond that variable cannot be accessed.

1. **LOCAL VARIABLE**

Scope: Local to the block in which variable defined.
Life: Till the control remains within the block in which the variable is defined.

Example:

```
main()
{
      int i=4;
      printf(“%d”,i);

}
```

Output: 4
Here i value is 4 only within main block

2. **GLOBAL VARIABLE**

Scope: Globally accessed till the termination of the program all sub function can access this variable.
Life: As long as program does not come to an end.
Example:

```
void sub();
int i=4;
main()
{
       sub();
      printf(“%d”,i);
}
void sub()
{
      printf(“%d”,i);
}
```

Output: 4 4
Here i value is 4 both sub() and main block

Definition of Scope of variable and types : 2M,

Local variable: explanation:1 M Example: 1M

Global variable: explanation:1 M Example: 1M

	c)	**Write a program to access structure members using pointers.**	**6M**

Ans:

<u>Program Of to access structure members using pointers;-</u>

```
#include <stdio.h>
#include <conio.h>
struct person
{
  int age;
  char pname[30];
};
void main()
{
   struct person *personPtr, person1;
   clrscr();
   personPtr = &person1;

   printf("Enter age: ");
   scanf("%d", &personPtr->age);

   printf("Enter person name: ");
   scanf("%s", &personPtr->pname);

   printf("Displaying:\n");
   printf("Age: %d\n", personPtr->age);
   printf("Person name: %s", personPtr->pname);
   getch();
  }
```

Correct Logic : 3M, Correct Syntax : 3M

(Any other program can be considered)