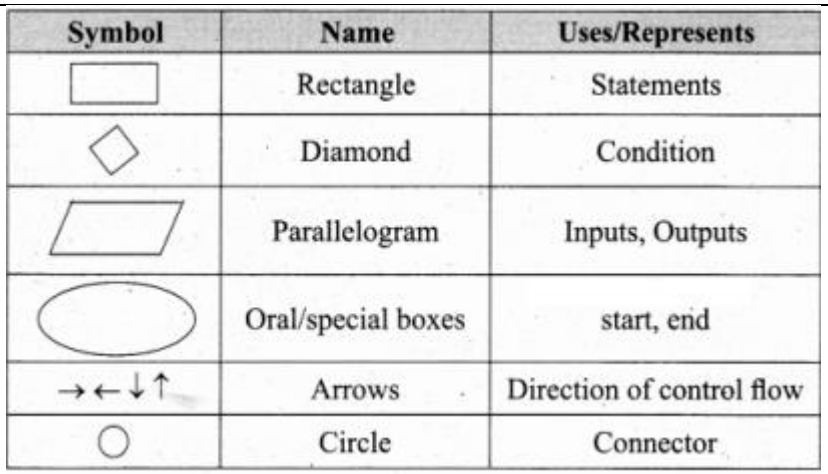
SUMMER – 2022 EXAMINATION

Subject Name: Programming in C **Model Answer** **Subject Code:** 22226

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
1		**Attempt any FIVE of the following:**	**10 M**
	a)	**Draw and label different symbols used in flowchart.**	**2 M**
	Ans		4 symbols with name- each carry ½ M
	b)	**List any four keywords used in 'C'.**	**2 M**
	Ans	Following are the keywords used in 'C':	4 keywords ½ M each

Symbol	Name	Uses/Represents
	Rectangle	Statements
	Diamond	Condition
	Parallelogram	Inputs, Outputs
	Oral/special boxes	start, end
→ ← ↓ ↑	Arrows	Direction of control flow
	Circle	Connector

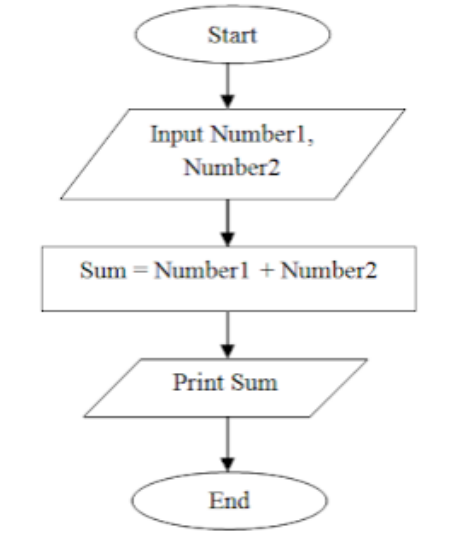
auto	double	int	struct
break	else	long	switch
case	enum	register	typedef
char	extern	return	union
continue	for	signed	void
do	if	static	while
default	goto	sizeof	Volatile
const	float	short	Unsigned

	c)	**State any four decision making statements.**	**2 M**
	Ans	Following are decision making statements: 1. if statement. 2. if..else statements. 3. nested if statements. 4. if-else-if ladder. 5. switch statements. 6. for loop 7. do..while loop 8. While loop	4 decisions making- ½ M each
	d)	**Define array. List its types.**	**2 M**
	Ans	● **Array** is a collection of similar datatype elements stored at contiguous memory locations. ● **Array** is a fixed-size sequential collection of elements of the same type. ● Arrays are the derived data type in C programming language which can store the primitive type of data such as int, char, double, float, etc **Types of Array:** 1. One Dimensional 2. Two Dimensional 3. multi-Dimensional	Definition of an array-1 M Types of array (any 2)-1 M
	e)	**Write syntax and use of PQW ( ) function or <math.h> header file.**	**2 M**

	Ans	**Pow():**
		compute the power of a input value.
		Syntax: double pow (double x, double y);
		Example: pow(2,4); // returns 16
		<math.h> header file declares a set of functions to perform mathematical operations such as: sqrt() to calculate the square root, log() to find natural logarithm of a number etc.
		Syntax: #include<math.h>
			any one written should be given 2 marks

	f)	**State the syntax to declare a pointer variable with example.**	**2 M**

	Ans	**Syntax:** datatype *pointer_variable_name;
		Example: int *p;
		
		Initialization Syntax: pointer = &variable;
		Initialization Example:
		int a; // Step 1
		int *p; // Step 2
		a = 10; // Step 3
		p= &a; // Step 4
			Syntax to declare -1M
			
			Example-1 M

	g)	**Draw flowchart for addition of two numbers.**	**2 M**
	Ans	 Fig. Flowchart to Add two numbers	Appropriate flowchart 2 marks
2.		**Attempt any <u>THREE</u> of the following:**	**12 M**
	a)	**Write an algorithm to find largest of three numbers.**	**4 M**
	Ans	Algorithm for finding the largest among them: Step 1: Start Step 2: Declare three integer variables a, b, c Step 3: If a is greater than b, if a is greater than c, Print "a is largest" else Print "c is largest" else if b is greater than c,	Start and stop: 1M Declaring variable: 1 M Logic: 2M

Print "b is largest"
else
Print "c is largest"
Step 4: Stop

b) Explain do while loop with example. **4 M**

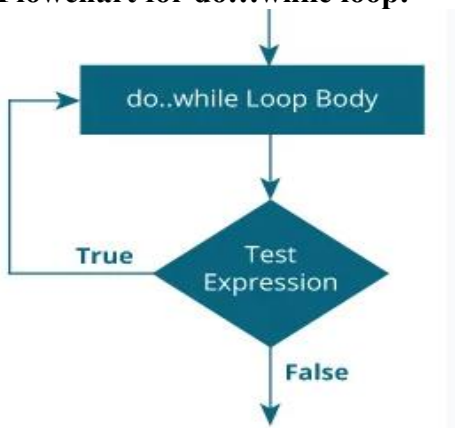
Ans

Use:
In 'do-while' loop the controlling condition appears at the end of the loop.
The iteration occurs at least once even if the condition is false at the first iteration.
It is an exit controlled loop.

Syntax:

```
do
{
//executable code;
} while(condition);
```

Flowchart for do…while loop:



Example:

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

Output:

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

Use: 1M

Syntax: 1M

Example: 1M

Flowchart:1 M

c) Differentiate between character array and integer array with respect to size and initialisation. **4 M**

Size: 2 M

	Ans		Initialization : 2 M

Parameter	Character Array	Integer Array
Size	Size Last location in character array is filled with '\0' so the array size should be so declared that it should have one last location for '\0' character.	No extra location than the number of elements is required.
definition	A char array is an object capable of holding values of type char.	An integer array is an object capable of holding values of type int.
Initialization	Initialization can be done like : char str[4]={'a','b','c','\0'}; char str[4]="abc";	Initialization can be done like : int arr[4]={1,2,3,4};

	d)	**Explain meaning of following statements with reference to pointers** **int *a, b ;** **b = 20 ;** ***a = b ;** **a = &b ;**	**4 M**
	Ans	**int *a,b;** It is declaration of integer pointer a and integer variable b. **b=20;** value 20 is assigned to variable b. ***a=b;** Value of b is assigned to pointer a. **a=&b;** Address of b is assigned to variable a.	Each statement: 1M
3.		**Attempt any <u>THREE</u> of the following:**	**12 M**
	a)	**Describe the following terms**: (i) **Keyword** (ii) **Identifier** (iii) **Variable** (iv) **Constant**	**4 M**

	Ans	**Keywords:** Keywords are special words in C programming which have their own predefined meaning. For Example: int, float, char, if, while, for, do. **Identifier:** Identifiers are user-defined names of variables, functions and arrays. It comprises of combination of letters and digits. For Example : int age1; Here, age1 is an identifier of integer data type. **Variable:** A variable is a name given to a storage area .Variable stores values. Each variable in C has a specific type and size. For Example : int age1; Here, age1 is variable of integer type. **Constant :** Constants refer to fixed values that the program may not change during its execution. For Example : const int a=30; Here, a is an integer constant.	Each term 1 M
	b)	**List the categories of functions and explain any one with example.**	**4 M**
	Ans	**Different categories of function:** 1) Function with no arguments and no return value. 2) Function with arguments and no return value. 3) Function with no arguments and return value.	List 2 M

4) Function with arguments and return value.

1) Function with no arguments and no return value:

This category of function cannot return any value back to the calling program and it does not accept any arguments also. It has to be declared as void.

Explanation of any one category

2 M

For example:

```
void add()
{
  int a,b,c;
  a=2;
  b=3;
  c=a+b;
 printf("%d",c);
}
```

It should be called as add();

2) Function with arguments and no return value:

This category of function cannot return any value back to the calling program but it takes arguments from calling program. It has to be declared as void. The number of arguments should match in sequence, number and data type.

For example:

```
void add(int x,int y)
{
 int z;
 z=x+y;
```

```
printf("%d",z);
}
```

It should be called as add(2,3); where x will take 2 and y will take 3

as their values.

3) Function with no arguments and return value:

This category of function can return a value back to the calling program but it does not take arguments from calling program. It has to be declared with same data type as the data type of return variable.

For example:

```
int add()
{
 int a,b,c;
 a=2;
 b=3;
 c=a+b;
 return(c);
}
```

It should be called as int x = add(); where x will store value returned

by the function.

4) Function with arguments and return value:

This category of function can return a value back to the calling program but it also takes arguments from calling program. It has to be declared with same data type as the data type of return variable.

For example:

```
int add(int x, int y)
```

		{
		int z;
		z=x+y;
		return(z);
		}
		It should be called as int s = add(2,3); where x will have 2 and y will have 3 as their values and s will store value returned by the function.	

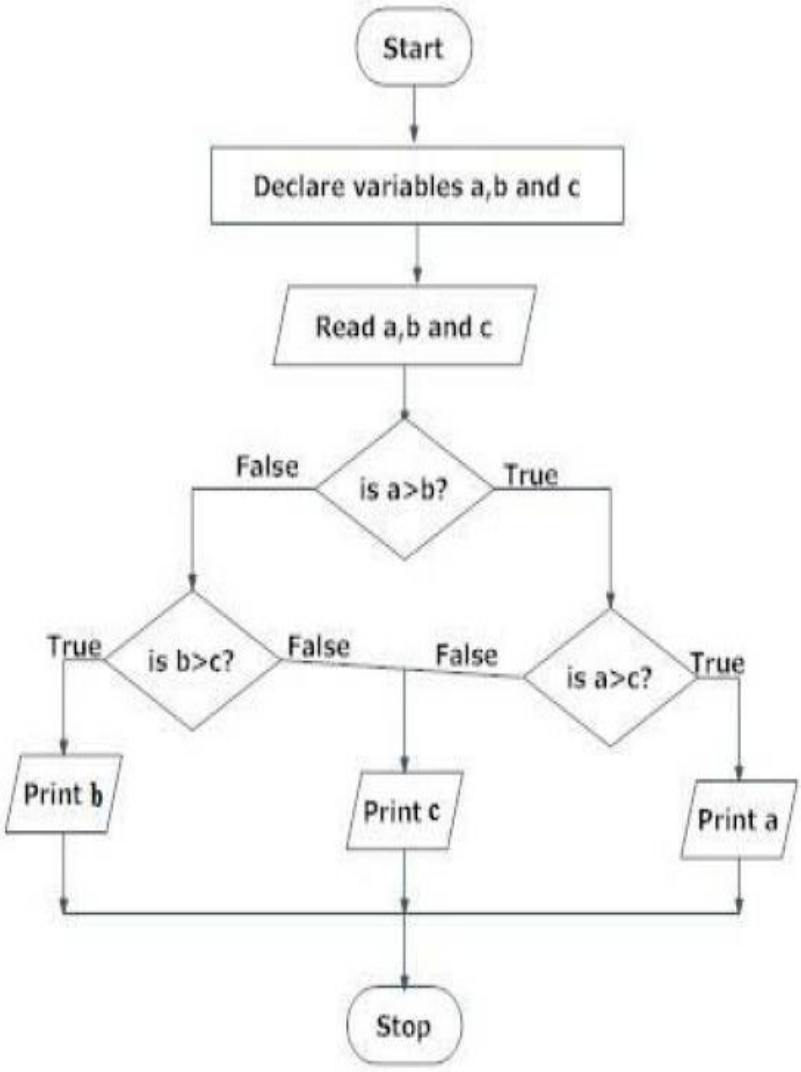
	c)	**State the use of printf( ) and scanf( ) with suitable example.**	**4 M**

	Ans	**printf()** : print() function is used to print the character, string, float, integer, octal and hexadecimal values onto the output screen.
		scanf() : scanf() function is used to read character, string, numeric data from
		Keyboard.
		Example:
		#include<stdio.h>
		void main()
		{
		int i;
		printf("Enter a number");
		scanf("%d", &i);
		printf("Entered number is: %d", i);
		}
			printf() use 1 M
			Scanf() use 1 M
			Example
			2 M

	d)	**Give any four differences between call by value and call by reference.**	**4 M**
	Ans		Any four valid differences 1M each

Sr.No	Call by value	Call by reference
1	When function is called by passing values then it is call by value.	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.	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.	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.	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.	Actual parameters change as function operates on value stored at the address.	
4.		**Attempt any <u>THREE</u> of the following:**			**12 M**
	a)	**Draw flowchart for finding largest number among three numbers.**			**4 M**
	Ans	(Can consider any other logically correct flowchart)			Use of correct symbols 2 M Relevant Logic 2M



b) **Write a program to display table of given number (Accept number from user).** **4 M**

Ans

```
# include <stdio.h>
void main()
{
   int num, i;
   printf (" Enter a number to generate the table in C: ");
   scanf (" %d", &num);
    printf ("\n Table of %d", num);
     for ( i = 1; i <= 10; i++)
   {

        printf ("\n %d * %d = %d", num, i, ( num* i) );

    }  }
```

Relevant logic 2 M

Syntax 2 M

	c)	**Write a program to sum all the even numbers between 1 to 100.**	**4 M**

	Ans	#include <stdio.h>
		void main()
		{
		int i, sum=0;
		
		for(i=2; i<=100; i=i+2)
		{
		sum =sum+ i;
		}
		printf("Sum of all even numbers from 1 to 100= %d", sum);
		
		}
		
		OR
		
		#include <stdio.h>
		void main()
		{
		int i, sum=0,rem;
		
		for(i=1; i<=100; i=i++)
		{
		rem= i%2;
		if (rem= =0)
		{
		sum =sum+ i;
		}
		}
		printf("Sum of all even numbers from 1 to 100= %d", sum);
		}
			Relevant logic 2 M
			
			Syntax 2 M

	d)	**Develop a program to find the factorial of a number using recursion.**	**4 M**

	Ans	#include<stdio.h>
		int factorial(int no)
		{
		if(no==1)
		{
		return(1);
		}
		else
		{
		return(no*factorial(no-1));
			Relevant logic 2 M
			
			Syntax 2 M

```
        }
}
void main()
{
int fact, no;
printf("\n Enter number");
scanf("%d", &no);
fact=factorial(no);
printf("\n Factorial number=%d", fact);
}
```

e) Write a program to accept ten numbers in an array. Sort array elements and display it. — **4 M**

Ans (Can consider code of ascending OR descending sort of an array element) — Relevant logic 2 M

Syntax 2 M

```
#include<stdio.h>
int main()
{
  int element[10],i,j,temp;
  printf("enter 10 integer numbers:");
  for(i=0;i<10;i++)
  {
    scanf("%d",&element[i]);
  }
  for(i=0;i<10-1;i++)
  {
    for(j=i+1;j<10;j++)
    {
      if(element[i]>element[j])
      {
        temp=element[i];      //swapping element[i] with element[j]
        element[i]=element[j];
        element[j]=temp;
      }
    }
  }
  printf("Elements are now in ascending order:");
  for(i=0;i<10;i++)
    {
      printf("%d\n",element[i]);
    }
  return 0;          }
```

5.		**Attempt any <u>TWO</u> of the following:**	**12 M**
	a)	**Write a program to print Fibonacci series starting from 0 and l.**	**6 M**

	Ans	#include<stdio.h>
		int main()
		{
		int n1=0,n2=1,n3,i,number;
		printf("Enter the number of elements:");
		scanf("%d",&number);
		printf("\n%d %d",n1,n2); //printing 0 and 1
		for(i=2;i<number;++i)
		{
		n3=n1+n2;
		printf(" %d",n3);
		n1=n2;
		n2=n3;
		}
		return 0;
		}
		
		Output:
		Enter the number of elements:15
		0 1 1 2 3 5 8 13 21 34 55 89 144 233 377
			Logic: 3 M
			Syntax: 3 M

	b)	**Write a program for addition of 3 x 3 matrices.**	**6 M**

	Ans	#include<stdio.h>
		#include<conio.h>
		int main()
		{
		int matA[3][3],matB[3][3],matC[3][3];
		int r,c,k;
		for(r=0; r<3; r++)
		{
		for(c=0; c<3; c++)
			Any relevant logic : 2 M
			Declaration and initialization of matrix: 2 M
			Syntax : 2 M

```
 {
   printf("Enter first matrix : ");
   scanf("%d", &matA[r][c]);
 }
}
for(r=0; r<3; r++)
{
 for(c=0; c<3; c++)
 {
   printf("Enter second matrix : ");
   scanf("%d", &matB[r][c]);
 }
}
for(r=0; r<3; r++)
{
 for(c=0; c<3; c++)
 {
   matC[r][c]=0;
   for(k=0; k<3;k++)
      matC[r][c] = matA[r][c] + matB[r][c];
 }
}
printf("\n New addition matrix : \n");
for(r=0; r<3; r++)
{
 for(c=0; c<3; c++)
    printf(" %d",matC[r][c]);
 printf("\n");
```

		}
		getch();
		return 0;
		}
		
		Output:
		Enter first matrix :
		1 2 3
		4 5 6
		7 8 9
		Enter second matrix :
		2 1 7
		4 6 3
		8 1 1
		
		New addition matrix :
		3 3 10
		8 11 9
		15 9 10	

	c)	**Write a program to compute the sum of all elements stored in an array using pointers.**	**6 M**

	Ans	#include <stdio.h>
		void main()
		{
		int arr1[10];
		int i,n, sum = 0;
		int *pt;
		
		printf("\n\n Pointer : Sum of all elements in an array :\n");
		printf("----------\n");
			Any relevant logic : 2 M
			Declaration and initialization of Pointers: 2 M
			Syntax : 2 M

```
   printf(" Input the number of elements to store in the array  : ");
   scanf("%d",&n);

   printf(" Input %d number of elements in the array : \n",n);
   for(i=0;i<n;i++)
     {
          printf(" element - %d : ",i+1);
          scanf("%d",&arr1[i]);
pt = arr1; // pt store the base address of array arr1

   for (i = 0; i < n; i++) {
     sum = sum + *pt;
     pt++;
   }

   printf(" The sum of array is : %d\n\n", sum);
}
```

Output:

```
Pointer : Sum of all elements in an array :
-----------------------------------------------
 Input the number of elements to store in the array  : 5
 Input 5 number of elements in the array :
 element - 1 : 2
 element - 2 : 3
 element - 3 : 4
 element - 4 : 5
```

		element - 5 : 6 The sum of array is : 20	
6.		**Attempt any <u>TWO</u> of the following:**	**12 M**
	a)	**Write a program using structure to display information of employee which consist of employee id, name, age and salary.**	**6 M**

	Ans	#include <stdio.h>
		
		/*structure declaration*/
		struct employee{
		int empId;
		char name[30];
		float age;
		float salary;
		};
		
		int main()
		{
		/*declare structure variable*/
		struct employee emp;
		
		/*read employee details*/
		printf("\nEnter details :\n");
		printf("ID ?:"); scanf("%d",&emp.empId);
		printf("Name ?:"); gets(emp.name);
		printf("Age ?:"); scanf("%f",&emp.age);
		printf("Salary ?:"); scanf("%f",&emp.salary);
			Any relevant logic : 2 M
			Declaration and initialization of structure: 2 M
			Syntax : 2 M

```
   /*print employee details*/
   printf("\nEntered detail is:");
   printf("Id: %d"     ,emp.empId);
   printf("Name: %s"   ,emp.name);
   printf("Age: %f"     ,emp.age);

   printf("Salary: %f\n",emp.salary);
   return 0;
}
```

Output:

Enter details :

ID ?:1120

Name ?:Mike

Age?: 25

Salary ?:76543

Entered detail is:

Id: 1120

Name: Mike

Age: 25.000000

Salary: 76543.000000

b) Write a program to demonstrate use of strcmp( ), strcpy( ), strlen( ), strcat( ). — 6 M

Ans

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

Inclusion of string.h -1 M

Program logic : 1 M

Syntax of each function (1

```
   char str1[10]="Hello";
   int l = strlen(str1);
   printf("Length of the string = %d",l);
   char str2[10]="World";
   char str3[10];
   strcat(str1,str2);
   printf("\n %s",str1);
   if(strcmp(str1,str2)==0)
   {
     printf("\n They are equal");
   }
   else
   {
    printf("\n They are not equal");

   }
   strcpy(str3,str1);
   printf("\n %s",str3);
   getch();

   return 0;
}
```

Output:

```
Length of the string = 5
 HelloWorld
 They are not equal
 HelloWorld
```

M each) : 4 M

	c)	**Write a program to perform arithmetic operations on pointer.**	**6 M**

	Ans	#include<stdio.h>
		int main()
		{
		int no1,no2;
		int *ptr1,*ptr2;
		int sum,sub,mult;
		float div;
		printf("Enter number1:\n");
		scanf("%d",&no1);
		printf("Enter number2:\n");
		scanf("%d",&no2);
		
		ptr1=&no1;//ptr1 stores address of no1
		ptr2=&no2;//ptr2 stores address of no2
		
		sum=(*ptr1) + (*ptr2);
		sub=(*ptr1) - (*ptr2);
		mult=(*ptr1) * (*ptr2);
		div=(*ptr1) / (*ptr2);
		
		printf("sum= %d\n",sum);
		printf("subtraction= %d\n",sub);
		printf("Multiplication= %d\n",mult);
		printf("Division= %f\n",div);
		
		return 0;
		}
			Logic:2 M
			Syntax of pointer on arithmetic operation :4M

		Output: Enter number1: 20 Enter number2: 10 Sum=30 Subtraction =10 Multiplication = 200 Division = 2.000000	