MODEL ANSWER
WINTER– 18 EXAMINATION

Subject Title: 'C' Programming Language Subject Code: 22218

3 Hours / 70 Marks

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. No.	Sub Q.N.	Answer	Marking Scheme
Q.1		**Solve any FIVE :**	**10-Total Marks**
	A)	**List 4 datatypes used in C.**	**2M**
	Ans:	***(Note: Any other correct data type shall be considered)*** **Data types:** • int • float • double • char • void	**Any four data types:1/2 M each)**
	B)	**State use of * and & used in pointers.**	**2M**
	Ans:	*** operator:-** It is used to declare a pointer variable. It is also used as value at operator i.e. it is used to refer value stored at the address (memory location) pointed by pointer variable. **& operator: -** It is used to retrieve address (memory location) of a variable from memory.	**(Correct use of each-1M)**
	C)	**Give syntax of switch case statements.**	**2M**
	Ans:	switch (expression) { Case constant-expression 1: Statement; break; /* optional */ Case constant-expression 2:	**(Correct syntax:2M)**

```
                    Statement;
                    break; /* optional */

        /* you can have any number of case statements */

        default: /* Optional */
                Statement;
    }
```

D)	**State any four control statements.**	**2M**
Ans:	Control statements:- 1. if 2. if-else 3. break 4. continue 5. switch 6. goto 7. while 8. for	**(Any four statements:1/ 2 M each)**
E)	**Define Array.**	**2M**
Ans:	An array is a collection of similar type of elements.	**(Correct definition:2M)**
F)	**List 2 mathematical functions used in C programming.**	**2M**
Ans:		**(Any two functions:1M each)**

sqrt() **pow()** **floor()**	**round()** **ceil()** **sin()**	**cos()** **cosh()** **exp()**	**tan()** **tanh()** **sinh()**	**log()** **log10()** **trunc()**

G)	**Define structure.**	**2M**
Ans:	**Structure:** A structure is a collection of one or more variables of same or different data types grouped together under a single name.	**(Correct definition:2M)**

Q 2		Solve any THREE :	12-Total Marks
	A)	**Distinguish between compiler and interpreter.**	4M
	Ans:		(Any four differences- 1M each)

Interpreter	Compiler
It translates one statement at time from program.	It scans entire program and translates complete program at as time.
Debugging is easy as if any error occurs it stops the execution after translating that particular step.	Debugging takes time as error occurs (if any) after complete program is scanned.
It does not produce any intermediate object code.	It generates intermediate object code.
It requires less memory as it does not create intermediate object code.	Memory requirement is more due to the creation of object code.
It takes less amount of time to analyze the source code but the overall execution time is slower.	It takes large amount of time to analyze the source code but the overall execution time is comparatively faster.

	B)	**Explain while loop with syntax and example.**	4M
	Ans:	While loop is an entry – controlled loop statement. The test- condition associated with while is evaluated and if the condition is true, then only body of the loop is executed. After execution of the body control passes to test condition and the test condition is once again evaluated and if it is true, the body is executed again. The process of test condition evaluation and execution of the body continues until the test condition becomes false. if test condition is false then control is transferred out of the loop. On exit, the program continues with the statement immediately after the body of the loop.	(Explanation- 2M,Syntax- 1M,Example -1M)

Syntax:

```
while(test condition)
{
   Body of the loop
 }
```

Example :

```
main()
{
  int i=1;
  while(i <=10)
 {
  printf("%d  ",i);
  i++;
 }
}
```

This will produce the output as 1 2 3 4 5 6 7 8 9 10

C)	**Explain the use of the following function with syntax:** **(i) Strcmp()** **(ii) Strlen()**	**4M**
Ans:	i) **strcmp()** :This library function is used to compare two strings. If the strings are equal then function returns value as 0 and if they are not equal then the function returns ASCII value difference of the first mismatched characters from the strings. **Syntax**: strcmp(string1,string2); **Example:** Consider str1=”abc” and str2=”abc” i=strcmp(str1,str2) strcmp function compares characters from str1 and str2 and returns 0 as both the strings are same. **ii)strlen():** This library function is used to count the length of the string i.e. number of characters including blank spaces from a string. **Syntax** : strlen(string1); **Example** : int i; char string1[]=”abc”; i=strlen(string1); strlen function counts number of characters from string1 and stores the count in the variable i.	**(Use of each function:1M , syntax of each function:1M)**
D)	**Write a program to calculate n^{th} power of a number using function.**	**4M**

Ans:	***(Note: Any other correct logic shall be considered.)***
	
	#include<stdio.h>
	#include<conio.h>
	#include<math.h>
	void power(int no,int n)
	{
	int p;
	p=pow(no,n);
	printf("\n power of number=%d",p);
	}
	void main()
	{
	int no,n;
	clrscr();
	printf("\n Enter number:");
	scanf("%d",&no);
	printf("\n Enter power:");
	scanf("%d",&n);
	power(no,n);
	getch();
	}
		(correct logic-2M,correct syntax-2M)

Q. 3		Solve any THREE :	12-Total Marks
	A)	**Write a program to accept ten numbers in array and arrange them in ascending order.**	4M

	Ans:	#include<stdio.h>
		#include<conio.h>
		void main()
		{
		int arr[10],repeat,temp=0,i;
		clrscr();
		for(i=0;i<=9;i++)
		{
		printf("Enter elements of arr a:");
		scanf("%d",&arr[i]);
		}
		temp=arr[0];
		for(repeat=0;repeat<=9;repeat++)
		{
		for(i=0;i<=9;i++)
		{
		if(arr[i+1]<arr[i])
		{
		temp=arr[i];
		arr[i]=arr[i+1];
		arr[i+1]=temp;
		}
		}
		}
		printf("\n Array in asending order is:");
		for(i=0;i<10;i++)
		{
		printf("\n %d",arr[i]);
		}
		getch();
		}
			(Correct logic 2 M, Correct syntax 2M)

	B)	**Explain use of arrow (⟶) operator with example.**	4M
	Ans:	Use of (->) arrow operator To access members of a structure through a pointer, the arrow operator is used. arrow (->) is used to access the data using pointer variable. The -> (arrow) operator are used to reference individual members of classes, structures, and unions. If p_emp is a pointer to an object of type Employee, then to assign the value "tara" to the first_name member of object emp, you would write something as follows − strcpy(p_emp->first_name, "tara"); The -> is called the arrow operator. It is formed by using the minus sign followed by a greater than sign. **EXAMPLE :** In this program, “my_structure” is normal structure variable and “ptr” is pointer structurevariable. In this, Dot(.) operator is used to access the data using normal structure	**(Use of arrow operator 2 M, Example 2 M)**

variableand arrow(->) is used to access data using pointer variable.

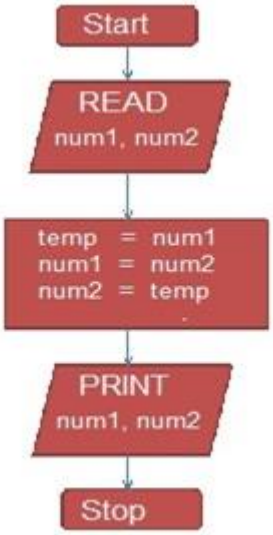
Accessing Structure Members with Pointer

To access members of structure using the structure variable, we used the dot . operator. Butwhen we have a pointer of structure type, we use arrow -> to access structure members.

```
#include <stdio.h>
struct my_structure
{
char name[20];
int number;
int rank;
};
int main()
{
struct my_structure variable = {"Ganesh", 34, 1};
struct my_structure *ptr;
ptr = &variable;
printf("NAME: %s\n", ptr->name);
printf("NUMBER: %d\n", ptr->number);
printf("RANK: %d", ptr->rank);
return 0;
}
```

```
NAME: Ganesh
NUMBER: 34
RANK: 1
```

C)	**Write an algorithm and flowchart to swap the contents of two variables.**	**4M**
Ans:	**Algorithm:** • Step 1 : Start • Start 2 : READ num1, num2 • Start 3 : temp = num1 • Start 4 : num1 = num2 • Start 5 : num2 = temp • Start 6 : PRINT num1, num2 • Start 7 : Stop Flowchart:	**(Correct algorithm 2 M, Flowchart 2 M)**

			
	d)	**Write a program to find whether the character entered through keyboard is a vowel or not.**	**4M**

	Ans:	#include<stdio.h>
		void main()
		{
		char ch;
		printf("Enter the character");
		scanf("%c",&ch);
		if(ch=='A'||ch=='E'||ch=='I'||ch=='O' ||ch=='U'||ch=='a'||ch=='e' ||ch=='i' || ch=='o' ||ch== 'u')
		printf("\n Entered character is Vowel");
		else
		printf("\n Entered character is consonant");
		}
			(Correct logic 2 M, Correct syntax 2M)

Q. 4	**A)**	**Solve any THREE :**	**12-Total Marks**
	A)	**Explain how to initialize two dimensional array with example.**	**4M**

	Ans:	**Initializing Two-Dimensional Arrays**
		Multidimensional arrays may be initialized by specifying bracketed values for each row. Following is an array with 3 rows and each row has 4 columns.
		int a[3][4] = {
		{0, 1, 2, 3} , /* initializers for row indexed by 0 */
		{4, 5, 6, 7} , /* initializers for row indexed by 1 */
		{8, 9, 10, 11} /* initializers for row indexed by 2 */
		};
		The nested braces, which indicate the intended row, are optional.
		Thefollowing initialization is equivalent to the previous example −
		int a[3][4] = {0,1,2,3,4,5,6,7,8,9,10,11};
		Example:
		#include <stdio.h>
		int main ()
		{
		/* an array with 5 rows and 2 columns*/
			(Explanation 2M, Example 2 M)

```
int a[5][2] = { {0,0}, {1,2}, {2,4}, {3,6},{4,8}};
int i, j;
/* output each array element's value */
for ( i = 0; i< 5; i++ )
{
for ( j = 0; j < 2; j++ )
{
printf("a[%d][%d] = %d\n", i,j, a[i][j] );
}
}
return 0;
}
```

B) Explain recursive function with suitable example. **4M**

Ans: **(Explanation 2 M, Example 2 M)**

A function that calls itself is known as a recursive function. And, this technique is known as recursion.

But while using recursion, programmers need to be careful to define an exit condition from the function, otherwise it will go into an infinite loop.

Recursive functions are very useful to solve many mathematical problems, such as calculating the factorial of a number, generating Fibonacci series, etc.

```
#include<stdio.h>
int find_factorial(int);
int main()
{
int num, fact;
printf("\nEnter any integer number:");
scanf("%d",&num);
//Calling our user defined function
fact =find_factorial(num);
//Displaying factorial of input number
printf("\nfactorial of %d is: %d",num, fact);
return 0;
}
int find_factorial(int n)
{
if(n==0) //Factorial of 0 is 1
return(1);
return(n*find_factorial(n-1)); //Function calling itself: recursion
}
```

Output:

```
Enter any integer number: 4
factorial of 4 is: 24
```

C) State and explain four arithmetic operations perform on pointer. **4M**

Ans: **(Explain any four arithmetic operations 2 M, Example 2 M)**

Arithmetic operations perform on Pointer:

Basic operations +, -, *, / , ++ , -- can be done using pointer notation.

Some of the following operations are possible:

e.g.

add = *p1 + *p2	Adds the value of pointer p1 and p2
y= *p1 -*p2	Subtracts values of pointer p1 and p2
x= *p1 / *p2	Divide the values of p1 and p2

		x= *p1 ** p2 Multiplies values of p1 and p2
		(*pl)++ :- This statement increments value, stored at the memory address pointed by pointerpl, by 1.
		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 conditional operator with example.**	**4M**
	Ans:	**Conditional Operator (Ternary Operator):** It takes the form "?:" to construct conditional expressions. The operator "? :" works as follows: Syntax: exp1? exp2 : exp 3 ; Where exp1, exp2 and exp3 are expressions. exp1 is evaluated first, Ifit is true, then expression exp2 is evaluated. If exp1 is false, exp3 isevaluated. Example: int a=10,b=5,x; x=(a>b) ? a : b; In the above example x will take value 10 because condition given isif a>b.	**(Explanation 3 M, Example 1 M)**
Q.5		**Solve any TWO :**	**12-Total Marks**
	A)	**Write a program to add two 3×3 matrices.**	**6M**

	Ans:	#include<stdio.h>
		#include<conio.h>
		void main()
		{
		int a[3][3], b[3][3], add[3][3], i, j;
		clrscr();
		printf("Enter values for first matrix: \n");
		for(i=0;i<3;i++)
		{
		for(j=0;j<3;j++)
		{
		printf("Enter matrix 1 entry(%d,%d): ",i,j);
		scanf("%d",&a[i][j]);
		}
		}
		printf("Enter values for second matrix: \n");
		for(i=0;i<3;i++)
			(Correct logic : 3M, Correct syntax: 3M)
			
			(Any other logic can be considered)

```
        {
                for(j=0;j<3;j++)
                {
                        printf("Enter matrix 2 entry(%d,%d): ",i,j);
                        scanf("%d",&b[i][j]);
                }
        }
        //Performing addition
        for(i=0;i<3;i++)
        {
                for(j=0;j<3;j++)
                {
                        add[i][j] = a[i][j] + b[i][j];
                }
        }
        printf("Addition matrix is: \n");
        for(i=0;i<3;i++)
        {
                for(j=0;j<3;j++)
                {
                        printf("%d\t",add[i][j]);
                }
                printf("\n");
        }
        getch();
}
```

B) **Write a program to add two numbers using function.** **6M**

Ans:

```
#include<stdio.h>
#include<conio.h>
void add(int, int);
void main()
{
        int a, b;
        clrscr();
        printf("Enter two number: ");
        scanf("%d%d",&a,&b);
        add(a,b);
        getch();
}
void add(int a, int b)
{
        printf("Addition of %d and %d is %d",a,b,a+b);
}
```

Correct program : 4M

(Any other logic can be considered)

C) **Write a program to exchange values of two variables using pointers.** **6M**

Ans:

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

(Correct Logic: 3M, Correct Program :

		int a, b, *p;
		clrscr();
		printf("Enter value of a: ");
		scanf("%d",&a);
		printf("Enter value of b: ");
		scanf("%d",&b);
		printf("Before swapping: a:%d b:%d",a,b);
		*p = a;
		a = b;
		b = *p;
		printf("\nAfter swapping: a:%d b:%d",a,b);
		getch();
		}
			3M)
			
			(Any other logic can be considered)

Q.6		**Solve any TWO :**	**12-Total Marks**
	A)	**Write a program to declare a structure student having data members roll_no, name and agg_marks. Accept data and display this information for one student.**	**6M**

	Ans:	#include<stdio.h>
		#include<conio.h>
		struct student
		{
		int roll_no;
		char name[20];
		float agg_marks;
		}s;
		void main()
		{
		clrscr();
		printf("Enter student roll no, name, aggregrate marks: ");
		scanf("%d%s%f",&s.roll_no,&s.name,&s.agg_marks);
		printf("\nRollNo\tName\tAggregrate");
		printf("\n%d\t%s\t%f",s.roll_no,s.name,s.agg_marks);
		getch();
		}
			(Structure declaration :2M, Accept elements: 2 M, Display elements:2 M

	B)	**Write a program to print table of a given number.**	**6M**

	Ans:	#include<stdio.h>
		#include<conio.h>
		void main()
		{
		int n, i;
		clrscr();
		printf("Enter a number: ");
		scanf("%d",&n);
		printf("\nTable of %d :\n",n);
		for(i=1;i<=10;i++)
		{
		printf("%d * %d = %d\n",n,i,n*i);
		}
			(Correct Logic : 3M,Correct Syntax : 3M)
			
			(Any other logic can be considered)

		getch(); }	
	C)	**Write a program to concatenate two strings.**	**6M**

	Ans:	#include<stdio.h>
		#include<conio.h>
		#include<string.h>
		void main()
		{
		char str1[40], str2[20];
		clrscr();
		printf("Enter two strings: ");
		scanf("%s%s",&str1,&str2);
		strcat(str1,str2);
		printf("Concatenated string is: %s",str1);
		getch();
		}
			(correct logic: 3M,correct syntax :3M)
			
			(Any other logic considered)