V2V EDTECH LLP

Online Coaching at an Affordable Price.

OUR SERVICES:

- Diploma in All Branches, All Subjects
- Degree in All Branches, All Subjects
- BSCIT / CS
- Professional Courses

+91 93260 50669

v2vedtech.com

V2V EdTech LLP

v2vedtech

Winter – 19 EXAMINATION

Subject Name: Programming in C Model Answer Subject Code: 22218

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.

<table>
<tr><th>Q. No.</th><th>Sub Q. N.</th><th>Answer</th><th>Marking Scheme</th></tr>
<tr><td>1.</td><td></td><td>Attempt any Five of the following:</td><td>10M</td></tr>
<tr><td></td><td>a</td><td>State any four relational operators in C.</td><td>2M</td></tr>
<tr><td></td><td>Ans</td><td>There are following Relational Operators Available in C:

<table><tr><th>Operator</th><th>Use</th></tr><tr><td>==</td><td>equal to</td></tr><tr><td>!=</td><td>Not equal to</td></tr><tr><td><</td><td>less than</td></tr><tr><td>></td><td>Greater than</td></tr><tr><td><=</td><td>Less than equal to</td></tr><tr><td>>=</td><td>Greater than equal to</td></tr></table></td><td>Each operator with its use ½ M</td></tr>
</table>

	b	**Give the syntax for switch case statement.**	**2M**

	Ans	Syntax:
		switch(variable)
		{
		case value1:
		statements
		break;
		case value2:
		statements;
		break;
		. . .
		default:
		statements;
		break;
		}
			Correct syntax 2M

	c	**State the use of continues statement.**	**2M**
	Ans	**Continue** statement is mostly used inside loops. Whenever it is encountered inside a loop, control directly jumps to the beginning of the loop for next iteration, skipping the execution of statements inside loop's body for the current iteration.	Use 2M
	d	**Define the term function.**	**2M**
	Ans	A function is a group of statements that together perform a task. Every C program has at least one function, which is main().	Correct definition2M
	e	**State any two advantages of pointer.**	**2M**
	Ans	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.	Each advantage : 1M
	f	**State the use of '&' and '*' operators used with pointer**	**2M**
	Ans	* **Operator: -** It is used to declare a pointer variable. Example: int *ptr; The above statement declares ptr as an integer pointer variable. OR	& Operator use 1M *Operator use 1M

		It is also used as value at operator i.e. it reads the value from the address stored in pointer variable. Example: printf("%d", *ptr); The above statement displays value present at the address stored in ptr variable. **& operator: -** It is used to retrieve address of a variable from memory. Example: int *ptr,a; ptr=&a; The above statement stores the address of variable a in the pointer variable ptr.	
	g	**Write any two features of structure.**	**2M**
	Ans	1. C Structure is a collection of different data types which are grouped together and each element in a C structure is called member. 2. If you want to access structure members in C, structure variable should be declared. 3. Many structure variables can be declared for same structure and memory will be allocated for each separately. 4. It is a best practice to initialize a structure to null while declaring, if we don't assign any values to structure members.	1M for each feature
2.		**Attempt any Three of the following:**	**12M**
	a	**Describe scanf () with syntax and example.**	**4M**
	Ans	In C programming language, scanf() function is used to read character, string, numeric data from keyboard Syntax: Scanf("format specifier", &variable); Example: Scanf("%d", &n);	Description 2M,Syntax 1M,Example 1M
	b	**With suitable example, describe importance of break statement used in switch statement.**	**4M**

	Ans	#include <stdio.h>
		int main()
		{
		int i=2;
		switch (i)
		{
		case 1:
		printf("Case1 ");
		break;
		case 2:
		printf("Case2 ");
		break;
			Use:2M, Example: 2M

```
            case 3:
              printf("Case3 ");
              break;
            case 4:
              printf("Case4 ");
              break;
            default:
              printf("Default ");
        }
        return 0;
    }
```

In **switch case,** the break statement is used to terminate the switch case. Basically it is used to execute the statements of a single case statement. If no break appears, the flow of control will fall through all the subsequent cases until a break is reached or the closing curly brace '}' is reached.

	c	**State any two advantages and any two limitations of an array.**	**4M**
	Ans	**Advantages:** 1. Pointers reduce the length and complexity of a program. 2. They increase execution speed. 3. A pointer enables us to access a variable that is defined outside the function. 4. Pointers are more efficient in handling the data tables. 5. The use of a pointer array of character strings results in saving of data storage space in memory. 6. It supports dynamic memory management. **Limitations:** 1. Array is Static data Structure 2. Elements belonging to different data types cannot be stored in array 3. Inserting element is very difficult because before inserting element in an array we have to create empty space by shifting other elements one position ahead. 4. Deletion is not easy because the elements are stored in contiguous memory location. 5. Wastage of Memory , if array of large size is defined	Each advantage and limitation1M

	d	**Differentiate between call by value and call by reference methods for passing parameter. (any four points)**	**4M**
	Ans		Each point 1M

Call by value	**Call by reference**
A copy of actual arguments (value) is passed to respective formal arguments	Address of actual arguments is passed to formal arguments.
Actual arguments will remain safe, they cannot be modified accidentally.	Alteration to actual arguments is possible within from called function; therefore the code must handle arguments carefully else you get unexpected results.
Address of the actual and formal arguments are different	Address of the actual and formal arguments are the same
Changes made inside the function are not reflected in other functions	Changes made in the function are reflected outside also.

Example:
`#include <stdio.h>`
`void swapnum( int var1, int var2 )`
`{`
`int tempnum ;`
`tempnum = var1 ;`
`var1 = var2 ;`
`var2 = tempnum ;`
`}`
`int main( )`
`{`
`int num1 = 35, num2 = 45 ;`
`printf("Before swapping: %d, %d", num1, num2);`
`swapnum(num1, num2);`
	Example:
	`#include <stdio.h>`
	`void swap(int *n1, int *n2);`
	`int main()`
	`{`
	`int num1 = 5, num2 = 10;`
	`swap( &num1, &num2);`
	`printf("num1 = %d\n", num1);`
	`printf("num2 = %d", num2);`
	`return 0;`
	`}`
	`void swap(int* n1, int* n2)`
	`{`

		`printf("\nAfter swapping: %d, %d", num1, num2);`
		`}` | `int temp;`
		`temp = *n1;`
		`*n1 = *n2;`
		`*n2 = temp;`
		`}`	

3.		**Attempt any Three of the following:**	**12M**
	a	**Describe with suitable example difference between pre increment and post increment operator.**	**4M**

	Ans	**Pre Increment operator(++i):**
		When prefix ++ is used in an expression, the variable is incremented first and then the expression is evaluated using the new value of the variable.
		Example:
		main()
		{
		int a,b=10;
		a=++b;
		printf(“ a=%d ”,a);
		}
		Output:
		a=11
		
		Post increment operator (i++):
		When postfix ++ is used with a variable in an expression, the expression is evaluated first using the original value of the variable and then the variable is incremented by one.
		Example:
		main()
		{
		int a,b=10;
		a=b++;
		printf(“ a=%d ”,a);
		}
		Output:
		a=10
			Pre increment - 2M, Post increment- 2M

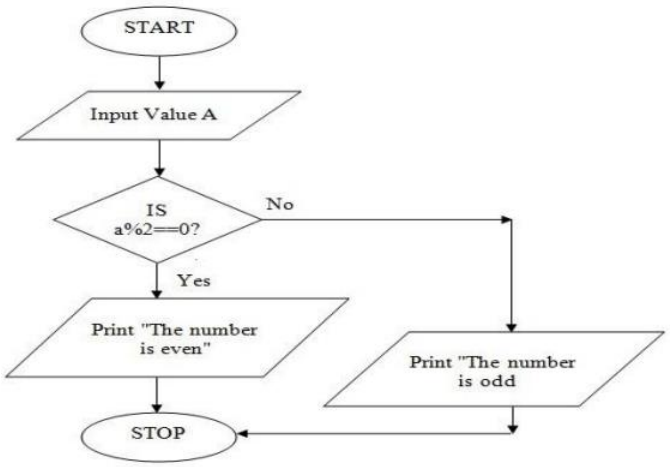
	b	**Describe declaration and initialization of two dimensional arrays.**	**4M**
	Ans	The array which is used to represent and store data in a tabular form is called as two dimensional array. Such type of array is specially used to represent data in a matrix form. **Declaration of two dimensional arrays:**	Declaration – 2M, Initalization- 2M

		Syntax:- Datatype array name [row size] [column size]; Eg: int arr[3][4]; It will declare array "arr" with 3 rows and 4 columns. **Initializing Two-Dimensional Arrays** Multidimensional arrays may be initialized by specifying bracketed values for each row. Example int a[3][4] = { {0, 1, 2, 3} , {4, 5, 6, 7} , {8, 9, 10, 11}}; a is an integer array with 3 rows and each row has 4 columns. OR Example int a[3][4] = {0,1,2,3,4,5,6,7,8,9,10,11}; The nested braces, which indicate the intended row, are optional. So, array can also be initialized using above method.	
	c	**Describe pointer arithmetic with any two operations.**	**4M**
	Ans	The pointer arithmetic is done as per the data type of the pointer. The basic operations on pointers are **Increment:** It is used to increment the pointer. Each time a pointer is incremented, it points to the next location with respect to memory size. Example ptr++ ; If ptr is an integer pointer stored at address 1000, then ptr++ shows 1002 as incremented location for an int. **Decrement:** It is used to decrement the pointer. Each time a pointer is decremented, it points to the previous location with respect to memory size. Example ptr- -; If the current position of pointer is 1002, then decrement operation ptr-- results in the pointer pointing to the location 1000 in case of integer pointer as it require two bytes storage. **Addition:** When addition operation is performed on pointer, it gives the location incremented by the added value according to data type. Example ptr+2; If ptr is an integer pointer stored at address 1000, Then ptr+2 shows 1000+ (2*2) = 1004 as incremented location for an int.	Any two operations Explanation – 4 M

		Subtraction: When subtraction operation is performed on the pointer variable, it gives the location decremented by the subtracted value according to data type. Example ptr-2; If ptr is an integer pointer stored at address 1004, Then ptr-2 shows 1004-(2*2) = 1000 as decremented location for an int.	
	d	**With example describe enumerated data type.**	**4M**

	Ans	**Enumerated data type**
		
		• Enumeration (or enum) is a user defined data type in C.
		• It is mainly used to assign names to integral constants, the names make a program easy to read and maintain.
		• The keyword 'enum' is used to declare new enumeration types in C
		• **Example**
		
		#include<stdio.h>
		
		enum year{Jan, Feb, Mar, Apr, May, Jun, Jul,
		Aug, Sep, Oct, Nov, Dec};
		
		int main()
		{
		int i;
		for (i=Jan; i<=Dec; i++)
		printf("%d ", i);
		return 0;
		}
		Output: 0 1 2 3 4 5 6 7 8 9 10 11
			Explanation 2M, Example 2M

4.		**Attempt any Three of the following:**	**12M**
	a	**Write an algorithm and draw flowchart to find whether the entered number is even or odd.**	**4M**
	Ans	**Algorithm** Step1: Start Step2: Declare integer variable a Step3: Input value of a Step4: if (a%2 == 0) is true then Print "The number is Even" else print "The number is Odd". Step 5: Stop	algorithm 2M, flowchart 2M

		Flowchart 	
	b	**Write a program in C to print table of entered number.**	**4M**

	Ans	#include <stdio.h>
		#include<conio.h>
		void main()
		{
		int n, i;
		clrscr();
		printf("Enter an integer: ");
		scanf("%d",&n);
		for(i=1; i<=10; ++i)
		{
		printf("%d * %d = %d \n", n, i, n*i);
		}
		getch();
		}
			Logic 2M, syntax-2M

	c	**Describe the following functions with their syntax and example.** **i)strcat()** **ii)strcmp()**	**4M**
	Ans	**1. strcat ( ) -** This string function is used to join two strings together. **Syntax:** strcat (string1, string2); string1 and string2 are character arrays. When the function strcat ( ) is executed, string2 is appended to string1 i.e. contents of string2 are added at the end of string1. **Example:** Consider str1=”abc” and str2=”xyz” strcat(str1,str2); strcat function will append string “xyz” at the end of string “abc” and str1 will become ”abcxyz”	strcat() syntax-1M,example-1M, strcmp() syntax1M, example-1M

		2. 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.	
	d	**Write a c program to calculate sum of elements of given array using pointer.**	**4M**

	Ans	#include<stdio.h>
		#include<conio.h>
		int main()
		{
		int array[5]={1,2,3,4,5};
		clrscr();
		int sum=0;
		int i ;
		int *ptr;
		ptr = array[0]; //pointer points to base of an array
		for(i=0;i<5;i++)
		{
		//*ptr refers to the value at address
		sum = sum + *ptr;
		ptr++;
		}
		printf("\nThe sum is: %d",sum);
		getch();
		}
		
		OR
		
		#include<stdio.h>
		#include<conio.h>
		
		int main()
		{
		int array[5];
		clrscr();
		int i,sum=0;
		int *ptr;
		printf("\nEnter array elements (5 integer values):");
			logic -2M, syntax-2M

		for(i=0;i<5;i++)
		{
		scanf("%d",&array[i]);
		}
		ptr = array; //pointer points to base of an array
		for(i=0;i<5;i++)
		{
		//*ptr refers to the value at address
		sum = sum + *ptr;
		ptr++;
		}
		printf("\nThe sum is: %d",sum);
		getch();
		}	

	e	**Write a c program to create structure with members as day, month and year. assign initial values to that structure and display it**	**4M**

	Ans	#include <stdio.h>
		#include<conio.h>
		
		struct date
		{
		int day;
		int month;
		int year;
		};
		void main ()
		{ struct date d1;
		clrscr();
		d1.day=25;
		d1.month=04;
		d1.year=2019;
		printf("The date is: %d/%d/%d",d1.day,d1.month,d1.year);
		getch();
		}
			Logic -2M , syntax- 2M

5.		**Attempt any Two of the following:**	**12M**
	a	**Describe use of nested if-else statement with syntax and example.**	**6M**

	Ans	**Definition:**
		If...else statement used inside if statement used in a program is called as nested if...else statement. When series of decisions are involved in a program we can use nested if...else statement.
		Syntax :
		if(test condition1)
		{
			Definition 2M
			syntax 2M
			Example 2M

```
                if(test condition2)
                {
                        statement-1;
                }
                else
                {
                        statement-2;
                }
        }
        else
        {
            statement-3;
        }
        statement-x;
```

If test condition-1 is true, then condition-2 is checked.
If condition-2 is true, then statement-1 is evaluated.
If condition-2 is false then statement-2 is evaluated and then control is transferred to the statement-x.
If condition-1 is false then control passes to statemtn-3 and it is executed. Then control passes to statement-x

Program:-

```
#include <stdio.h>
#include <conio.h>
void main()
{
  int var1, var2;
  clrscr();
  printf("Input the value of var1:");
  scanf("%d", &var1);
  printf("Input the value of var2:");
  scanf("%d",&var2);
  if (var1 != var2)
  {
    printf("var1 is not equal to var2\n");
    //Nested if else
    if (var1 > var2)
    {
            printf("var1 is greater than var2\n");
    }
    else
    {
            printf("var2 is greater than var1\n");
    }
  }
```

		else
		{
		printf("var1 is equal to var2\n");
		}
		getch();
		}
		
		Output:-
		
		Input the value of var1:12
		Input the value of var2:21
		var1 is not equal to var2
		var2 is greater than var1	

	b	**Write a 'C' program to find largest number from an array of 10 numbers.**	**6M**

	Ans	**Program:-**
		#include <stdio.h>
		#include <conio.h>
		
		void main()
		{
		int a[10],i,largest;
		clrscr();
		printf("Enter array elements\n");
		for(i=0;i<10;i++)
		{
		scanf("%d",&a[i]);
		}
		largest=a[0];
		for(i=1;i<10;i++)
		{
		if (a[i]>largest)
		{
		largest=a[i];
		}
		}
		printf("The largest element in the array is : %d",largest);
		getch();
		}
		Output:-
		Enter array elements
		10 90 80 50 30 20 60 40 70 78
		The largest element in the array is : 90
			Correct Logic 3M
			Correct syntax 3M

	c	**Write a 'C' program to display Fibonacci series using recursion.**	**6M**

	Ans	**Program:-**
		
		#include<stdio.h>
		#include<conio.h>
		int Fibonacci(int n)
		{
		if(n == 0 || n == 1)
		return n;
		else
		return(Fibonacci(n-1) + Fibonacci(n-2));
		}
		
		void main()
		{
		int n, m= 0, i;
		clrscr();
		printf("Enter Total terms: ");
		scanf("%d", &n);
		for(i = 1; i <= n; i++)
		{
		printf("%d\t", Fibonacci(m));
		m++;
		}
		getch();
		}
		Output:-
		Enter Total terms: 10
		0 1 1 2 3 5 8 13 21 34
			Correct Logic 3M
			Correct syntax 3M

6.		**Attempt any TWO of the following:**	**12M**
	a	**Write a 'C' program to accept two strings from user. Display length of both the strings. Also concatenate two strings and display the output.**	**6M**

	Ans	**Program:-**
		
		#include <stdio.h>
		#include <conio.h>
		#include <string.h>
		void main()
		{
		char s1[20],s2[20];
		int a,b;
		clrscr();
		printf("Enter first string\n");
		scanf("%s",s1);
			Correct Logic 3M
			Correct syntax 3M

		printf("Enter second string\n");
		scanf("%s",s2);
		a=strlen(s1);
		b=strlen(s2);
		printf("Length of first string is : %d",a);
		printf("Length of second string is : %d",b);
		strcat(s1,s2);
		printf("Concatenated string is : %s",s1);
		getch();
		}
		Output:-
		Enter first string
		Programming
		Enter second string
		Networking
		Length of first string is : 11
		Length of second string is : 10
		Concatenated string is : ProgrammingNetworking	

	b	**Write a 'C' program to accept two numbers. Write a function add() to display addition of entered number. Write a function multiply() to display multiplication of entered number.**	**6M**

	Ans	**Program:-**
		#include <stdio.h>
		#include <conio.h>
		int a,b;
		void add()
		{
		printf("Sum = %d ",a+b);
		}
		void multiply()
		{
		printf("Product = %d ",a*b);
		}
		void main()
		{
		clrscr();
		printf("Enter first number\n");
		scanf("%d",&a);
		printf("Enter second number\n");
		scanf("%d",&b);
		add();
		multiply();
		getch();
		}
			Correct Logic 3M
			Correct syntax 3M

		Output:- Enter first number 10 Enter second number 5 Sum = 15 Product = 50	
	c	**Write a 'C' program to declare structure employee having data members as empid, empname. Accept data for 5 employees and display it.**	**6M**

	Ans	**Program:-**
		#include <stdio.h>
		#include <conio.h>
		struct employee
		{
		int empid;
		char empname[20];
		}e[5];
		void main()
		{
		int i;
		clrscr();
		printf("Enter employee details: \n");
		for (i=0;i<5;i++)
		{
		printf("Enter employee Id and employee name\n");
		scanf("%d%s",&e[i].empid,&e[i].empname);
		}
		printf("Employee details are: \n");
		for (i=0;i<5;i++)
		{
		printf("Employee Id is %d \n Employee name is %s \n",e[i].empid,e[i].empname);
		}
		getch();
		}
		Output:-
		Enter employee details:
		Enter employee Id and employee name
		1 ram
			Correct Logic 3M
			Correct syntax 3M

		Enter employee Id and employee name 2 john Enter employee Id and employee name 3 sita Enter employee Id and employee name 4 geeta Enter employee Id and employee name 5 rohan Employee details are: Employee Id is 1 Employee name is ram Employee Id is 2 Employee name is john Employee Id is 3 Employee name is sita Employee Id is 4 Employee name is geeta Employee Id is 5 Employee name is rohan	