Unit - III

Arrays and Structure

Syllabus:

3.1 Characteristics of an array, One dimension and two-dimensional arrays, concept of multi-dimensional arrays.
3.2 Array declaration and Initialization.
3.3 Operations on Arrays.
3.4 Character and String input/output and String related operations.
3.5 Introduction and Features of Structures, Declaration and Initialization of Structures, array of structures.
3.6 Type def, Enumerated Data Type.

Practical:

*Practical No. 16: * Implement C programs using One Dimensional Array. (e.g.-Write C program to input 5 numbers using array and display sum of it)
*Practical No. 17: * Implement C programs using Two-Dimensional Array. (e.g.-Write C program to calculate addition of two 3X3 matrices.)
*Practical No. 18: * Write C program to perform following operations without using standard string functions. i) Calculate Length of given string ii) Print reverse of given string.
Practical No. 19: Implement 'Structure' in C (e.g. - Add and Subtract complex numbers using structure)
*Practical No. 20: * Implement Array of Structure' in C (e.g.-Accept and Display 10 Employee information using structure)

ARRAY:

Q. What is array? How element of array can be accessed?

Ans:

An array is a fixed size sequenced collection of elements of similar data types which share a common name.

Syntax:

```
data_type array_name[size];
```

Eg :

Considering array of 5 integer elements:

```
int a[5];
```

Array elements can be accessed by their position. For example, if its an array of 5 integers, the first position is considered as 0, next 1 and so on last index position is 4.
Eg : int a[5]={2,4,3,5,6};

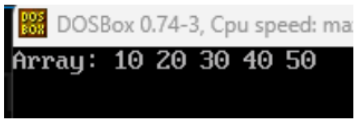
Then a[0]=2, a[1]=4, a[2]=3, a[3]=5, a[4]=6;

The elements can be accessed with help of a looping statement like „for" as follows:

```
for(i=0;i<5;i++)
{
printf("%d",a[i]);
}
```

***Practical No. 16: // Write C program for Implementation of one-dimensional array.**

```
#include <stdio.h>
#include <conio.h>
void main()
{
        int i; // i is Index of Array
        int numbers[5] = {10, 20, 30, 40, 50}; //Declare & Initialize Array
clrscr();
        // Display the values of the array
        printf("Array: ");
        for ( i = 0; i < 5; i++)
        {
        printf("%d ", numbers[i]); // displaying each index value, with white space
        }
getch();
}
```



Q. What are the Operations on Arrays? (any 5)

Ans:
Operations on Arrays are:

1. **Initialization:** Arrays can be initialized when they are declared or afterward.
   For example: int arr[5] = {1, 2, 3, 4, 5}; // Initialization during declaration

2. **Accessing Elements**: Elements of an array can be accessed using their index. Indexing typically starts from 0.
   For example: int x = arr[2]; // Accessing the third element of arr i.e '3'

3. **Updating Elements:** Elements of an array can be updated by assigning new values to them.
   For example: arr[2] = 10; // Updating the third element of arr

4. **Traversal:** Traversing an array involves accessing each element of the array in order to perform some operation on it. This is typically done using loops.
   For example:

```
for (int i = 0; i < 5; i++)
{
   // Access and do something with arr[i]
}
```

5. **Searching:** Searching involves finding a specific element in the array. Common algorithms for searching include linear search and binary search.

6. **Sorting:** Sorting involves arranging the elements of the array in a specific order, such as ascending or descending. Common sorting algorithms include bubble sort, selection sort, insertion sort, merge sort, and quicksort.

7. **Insertion and Deletion:** Arrays in C are static, meaning their size is fixed. Insertion and deletion operations often involve creating a new array with the desired elements and copying elements between arrays as needed.

8. **Concatenation:** Concatenating arrays involves combining two arrays into a single array. This typically requires creating a new array and copying the elements of both arrays into it.

9. **Copying:** Arrays can be copied either entirely or partially into another array.

10. **Merging:** Merging involves combining two sorted arrays into a single sorted array. This is often done as part of the merge sort algorithm.

Q. How the size of an array is calculated? Explain with suitable example.
Ans:
Size of an array is calculated with respect to two parameters:
1. Memory (number of bytes) required by the data type of the variable.
2. Number of elements declared as size in square bracket along with variable name. Size of an array=memory required by data type * number of elements inside array variable.

Example:-
int a[10];
Memory (number of bytes) required by the data type int- 2 bytes Number of elements in array variable a -10
Size of a=2*10=20 bytes

Q. Explain the declaration and initialize of character array.
Ans:
In C character array is used to represent string. String is a group of characters.

Declaration of character array:
Syntax: -char character_array_name[size];
char is a data type, character_array_name represents string name, size determines the number of characters in the character array. When compiler assigns a character string to a character array, it automatically stores a NULL character (\0) at the end of the string.

Example: -
char name[10];

Initialization of character array: Compile time initialization: -
Character array can be initialized when it is declared. Character array can be initialized with any of the following methods:

char city[7]="Mumbai";
city character array is 7 elements long as Mumbai contains 6 characters and 1 space for NULL character.

char city[10]="Mumbai";
city character array is declared as 10 characters long where Mumbai occupy 6 characters and remaining space is initialized with „\0" characters.

char city[7]={'M','u','m','b','a','I','\0'};
city character array is 7 elements long as Mumbai contains 6 characters and 1 space for NULL character.

char city[ ]={'M','u','m','b','a','i','\0'};
character array can be initialized without specifying size. In this case, the size of the array will be determined automatically, based on the number of elements initialized. In the example size of character array city is 7.

Run time initialization: -

An array can be initialized using scanf() function at run time. int a[5];

```
for (i=0;i<5;i++)
        scanf("%d",&a[i]);
```

Q. Syntax to declare two-Dimensional array. (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 and initialization of two-dimensional array:

Syntax :- data_type array_name [row size] [column size];

Example:- int arr[3][4];

This will declare array "arr" with 3 Rows and 4 Columns. A two-dimensional array can be considered as a table which will have x number of rows and y number of columns. A two-dimensional array a, which contains three rows, and four columns can be shown as follows

	Column 0	Column 1	Column 2	Column 3
Row 0	a[0][0]	a[0][1]	a[0][2]	a[0][3]
Row 1	a[1][0]	a[1][1]	a[1][2]	a[1][3]
Row 2	a[2][0]	a[2][1]	a[2][2]	a[2][3]

Thus, every element in the array a is identified by an element name of the form a[i][j], where 'a' is the name of the array, and 'i' and 'j' are the subscripts that uniquely identify each element in 'a'.

Example :

```
#include<stdio.h>
void main()
{
int a[2][2]={{1,2},{4,5});
int i, j;
        for(i=0;i<2;i++)
        {
                for(j=0;j<2;j++)
                {
                        printf("%d", a[i][j]); //here "i" is row and "j" is column
                }
        printf("\n"); // "\n" for next line
        }
getch();
}
```

OUTPUT:

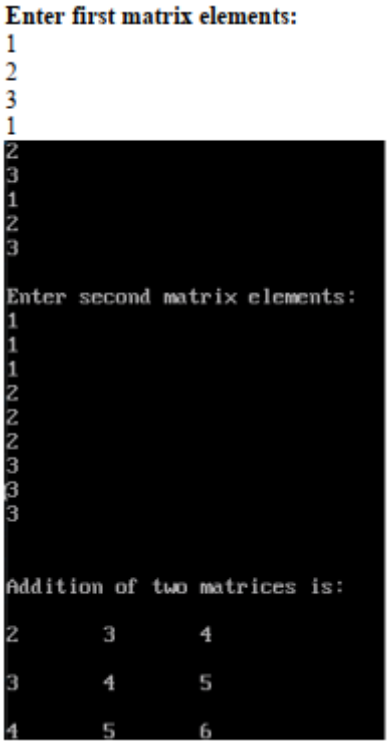
```
1 2
4 5
```

***Practical No. 17:**

Q. Write a program for addition of two 3 x 3 matrix. (Correct logic – 2Marks, correct syntax 2Marks)

Ans:

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

        printf("\nEnter second matrix elements:\n");
        for(i=0;i<3;i++)
        {
                for(j=0;j<3;j++)
                {
                   scanf("%d",&b[i][j]);
                }
        }
        //Adding two matrix
        for(i=0;i<3;i++)
        {
                for(j=0;j<3;j++)
                {
                     c[i][j]=a[i][j]+b[i][j];
                }
        }

        printf("\n\nAddition of two matrices is:");
        for(i=0;i<3;i++)
        {
                printf("\n\n");
                for(j=0;j<3;j++)
                {
                     printf("%d\t",c[i][j]);
                }
        }
getch();
}
```



Q. State any two advantages and any two limitations of an array.

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.

Disadvantage / 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.

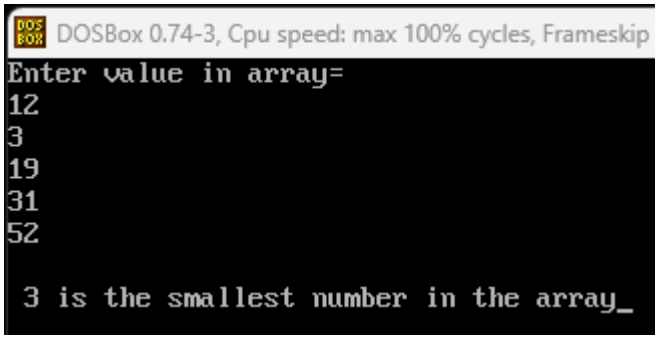
Q. Differentiate between character array and integer array with respect to size and initialization.

Sr.No	integer array	character array
1	An integer array is an array whose elements are all of an integer type which has no fractional component.	character array is an array which contains nothing but character types
2	An integer array is capable of holding values of type int.	A char array is capable of holding values of type char
3	int array take more memory to allocate	char array take less memory to allocate
4	integer array takes 2 bytes for each cell	whereas char takes 1 byte for each cell
5	array 's r not terminated with NULL(\0) character	Strings r terminated with NULL(\0) character

Q. Write a program to find smallest number in 5 element integer arrays. (Program Logic 2M, Correct Syntax 2M) Note: Any other relevant logic can be considered.

Ans:

```
#include<stdio.h>
#include<conio.h>
void main()
{
        int a[5], i=0, small=0;
clrscr();
        printf("Enter value in array=\n");
        for(i=0;i<5;i++)
        {
           scanf("%d", &a[i]);
        }

        small=a[0]; //considering 1st element of array as small

        for(i=0;i<5;i++)
        {
```



```
            if(a[i]<=small)
            {
                small=a[i];
            }
        }
        printf("\n %d is the smallest number in the array", small);
getch();
}
```

Q. Write a 'C' program to find largest number from an array of 10 numbers.
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]; //considering 1st element of array as largest

        for(i=1;i<10;i++)
        {
            if (a[i]>largest)
            {
              largest=a[i];
            }
        }
printf("The largest element in the array is : %d", largest);
getch();
}
```

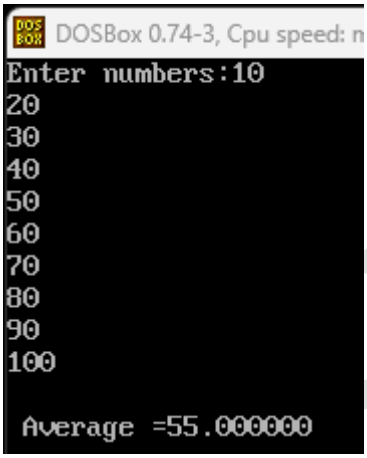
Output: -

```
        Enter array elements:
        10 90 80 50 30 20 60 40 70 78
        The largest element in the array is: 90
```

Q. Write a program to accept ten numbers and print average of it. (6m)

Ans:

```
#include<stdio.h>
#include<conio.h>
void main()
{
        int a[10], i, sum=0;
        float avg;
clrscr();
        printf("Enter numbers:");
        for(i=0;i<10;i++)
        {
          scanf("%d",&a[i]);
        }
        for(i=0;i<10;i++)
        {
           sum=sum+a[i]; //adding all elements of array
        }
        avg=sum/10;
        printf("\n Average =%f", avg);
getch();
}
```

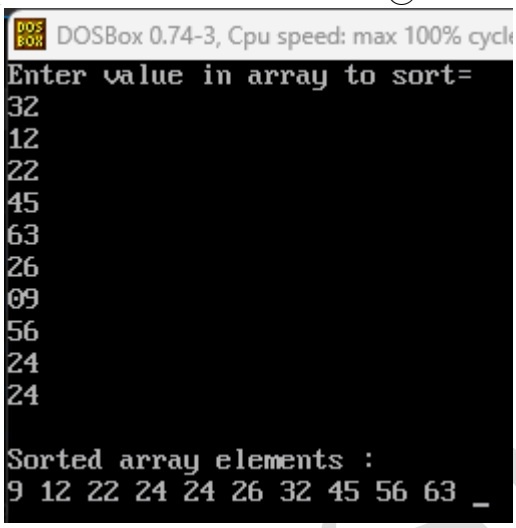


Q. Write a program to sort array element in ascending order. (6m)

Ans:

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

```
        for(i=0;i<10;i++)
        {
                for(j=0;j<10;j++)
                {
                        if(arr[i]<arr[j])
                        {
                                temp=arr[i];
                                arr[i]=arr[j];
                                arr[j]=temp;
                        }
                }
        }

        printf("\nSorted array elements :\n");
        for(i=0;i<10;i++)
        {
        printf("%d ",arr[i]);
        }
getch();
}
```



Q. Explain Character Input/Output:

Ans:

- In C programming, character input/output refers to the handling of single characters, typically represented by the char data type.
- String input/output, on the other hand, involves dealing with sequences of characters, which are represented as arrays of characters terminated by a null character ('\0').
- Strings in C are essentially arrays of characters.

- **getchar()**: Reads a single character from the standard input (usually the keyboard).
- **putchar()**: Writes a single character to the standard output (usually the console).

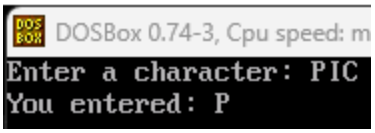
  Example:

  ```
  char ch;
  ch = getchar(); // Read a character
  putchar(ch);    // Write the character
  ```

Q. Write a C program to explain Character Input/Output.

```
#include <stdio.h>
#include <conio.h>

void main()
{
   char ch; // Declare a variable to store a character.
clrscr();
            // Prompt the user to enter a character
            printf("Enter a character: ");
```

```
    // Read a character from the standard input (keyboard)
    ch = getchar();

    // Display the entered character
    printf("You entered: ");

    // Write the character to the standard output (console)
    putchar(ch);

    // Newline for better output formatting
    printf("\n");

  getch();
}
```



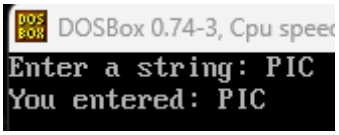
Q. Explain String Input/Output:

- **gets()**: Reads a string from the standard input. (Deprecated due to security vulnerabilities, should not be used.)
- **fgets()**: Reads a string from a specified input stream, usually the standard input.
- **puts()**: Writes a string to the standard output, followed by a newline character.
- **fputs()**: Writes a string to a specified output stream.
  Example:

```
char str[50];
fgets(str, sizeof(str), stdin); // Read a string
puts(str);                      // Write the string
```

Q. Write a C program to explain with comment String Input/Output.
Ans:

```
#include <stdio.h>
#include <conio.h>
void main()
{
  char str[50]; // Declare an array to store a string
clrscr();
  // Prompt the user to enter a string
  printf("Enter a string: ");

  // Read a string from the standard input (keyboard)
  fgets(str, 50, stdin);

  // Display the entered string
  printf("You entered: %s", str);

getch();
}
```



STRUCTURES:

Q. Explain structure in C with suitable example. 4m
Ans:
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;
```

Q. Explain initialization of structure. (2M)
Ans:
Initialization of structure

Syntax:

```
struct structure_name variable = { list of values};
```

Example:

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

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

Q. Write any two features of structure.

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.

Q. Write a program to declare a structure student having data member, roll_no; name and agg_marks. Accept data and display this information for one student.

Ans:

```
#include<stdio.h>
#include<conio.h>
        struct student
        {
        char name[20];
        float agg_marks;
        int roll_no;
        };

void main()
{
        struct student s;
clrscr();
        printf("\n Enter the name of a student:");
        scanf("%s",s.name);
        printf("\n Enter the aggregate marks:");
        scanf("%f",&s.agg_marks);
        printf("\n Enter the roll number:");
        scanf("%d",&s.roll_no);
        printf("\n\n Student data is:");
        printf("\n Name=%s", s.name);
        printf("\n Aggregate marks=%f", s.agg_marks);
        printf("\n Roll number=%d", s.roll_no);
getch();
}
```

Output:

```
Enter the name of a student: Arhaan
Enter the aggregate marks: 18.5
Enter the roll number: 12
Student data is:
Name= Arhaan
Aggregate marks= 18.5
Roll number= 12
```

Q. Write a program to declare a structure book having data members, title, author and price. Accept data and display information for one book. (Logic – 2 Marks, Syntax-2 Marks)

Ans:

```
#include<stdio.h>
#include<conio.h>
        struct book
        {
        char title[20];
        char author[20];
        int price;
        }b1;
void main()
{
clrscr();
        printf("\n Enter title of book:");
        scanf("%s",b1.title);
        printf("\n Enter author name of book:");
        scanf("%s",b1.author);
        printf("\n Enter price of book:");
        scanf("%d",&b1.price);
        printf("\n Details of book are:");
        printf("\n\t Title:%s \n Author:%s \n Price:%d", b1.title,b1.author,b1.price);
getch();
}
```

Output:

```
Enter title of book: See U in C
Enter author name of book: Ali Karim Sayed
Enter price of book: 1200

Details of book are:
Title: See U in C
Author: Ali Karim Sayed
Price: 1200
```

Q. 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();
}
```

Q. How structure members are access? Give example.

Ans:

- Structure members are accessed with the structure variable and (.) operator.
- In order to assign a value to a structure member, the member name must be linked with the structure variable using dot (.) operator also called period or member access operator.

Example – Consider book is a structure. Structure contains members as name and price. b1 is variable of a structure.

```
struct Book
{
char name[15];
int price;
} b1 ;

b1.price=200; //b1 is variable of Book type and price is member of Book
```

In the above statement b1 is a structure variable and price is a member of that structure.
Price is accessed using (.) operator preceded with variable name.

ARRAY OF STUCTURE:

Q. What is array of structure? List any two differences between array and array of structure. (Explanation of array of structure 2 Marks, Any 2 relevant differences 1 Mark each)
Ans:
Array of Structure is collection of structure. Structure is used to store the information of one particular object but if want to store many objects then array of structure is used.

Example -

```
struct student
{
int roll_no;
char name[10]; s
} s[5];
```

Following are differences:
Array stores similar data type elements whereas array of structure stores variables of structure where each structure variable contains different data type elements.

Example of array: int a[10];
Example Array of structure:

```
struct book
{
char name[10] ;
float price;
};

struct book b[100];
```

Q. Explain array of structure with example. (Explanation – 2M, example -2M)
Ans:
Array of Structure: -
Array of structures means collection of structure variables.
It can be used when we want to use many variables of the same structure.

For example:
If a structure for student data is defined and it has to be used for 10 different students, then array of structure can be declared as

```
struct student.
{
int rollno;
char name[20];
} s[10];
```

Here data in the form of rollno and name can be stored or accessed for 10 students.

For eg : s[0].rollno and s[0].name will be the data for first student.
s[1].rollno and s[1].name will be the data for second student and so on.

***Practical No. 20:**

Q. 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 Ravi
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

Q. Write a program to declare structure Stationery having data member, name, and quantity and cost. Accept and display this information for five items.
Ans:

```
#include<stdio.h>
#include<conio.h>
        struct Stationery
        {
        char name[20];
        int quantity, cost;
        }s[5];
void main()
{
        int i;
clrscr();
        printf("\n Enter information");
        for(i=0;i<5;i++)
        {
          scanf("%s %d %d", &s[i].name, &s[i].quantity, &s[i].cost);
        }
        printf("\n Display information");
        for(i=0;i<5;i++)
        {
           printf("%s %d %d", s[i].name, s[i].quantity, s[i].cost);
        }
getch();
}
```

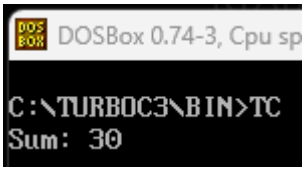
Q. What is Type def in C, explain with code example?
Ans

- typedef is used to create an alias or alternative name for existing data types.
- It allows programmers to define custom names for data types, making the code more readable and easier to maintain.
- This can be particularly useful when dealing with complex data types or when working with data types with long names.

//Type def Program:

```
#include <stdio.h>
#include <conio.h>

// Define a typedef for the data type 'int'
typedef int integer;
```

```
void main()
{
   // Declare variables using the typedef
   integer num1 = 10;
   integer num2 = 20;

   // Perform operations using the typedef
   integer sum = num1 + num2;

   // Display the result
   printf("Sum: %d\n", sum);

getch();
}
```



ENUM:

Q. With example describe enumerated data type.

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

Example

```
#include <stdio.h>
enum week {Sunday, Monday, Tuesday, Wednesday, Thursday, Friday, Saturday};
int main()
{
// creating today variable of enum week type
enum week today;
today = Wednesday;
printf("Day %d",today+1); //index position of Wednesday is 3, now plus 1 to it.
return 0;
}
```

Output:

Day