Batch: C1_2	**Roll No.:16010123032**
Experiment / assignment / tutorial No. 4	
Grade: AA / AB / BB / BC / CC / CD /DD	
Signature of the Staff In-charge with date	

TITLE: Write a program in C to demonstrate use of arrays

AIM: Program to sort the 1D array in the ascending or descending order and then accept the element from user and insert in the same array at its correct place by keeping array sorted
Write a program to find the Transpose of a Matrix.

Expected OUTCOME of Experiment:
Apply basic concepts of C programming for problem solving.(CO1 and CO2).

Books/ Journals/ Websites referred:

1. Programming in C, second edition, Pradeep Dey and Manas Ghosh, Oxford University Press.
2. Programming in ANSI C, fifth edition, E Balagurusamy, Tata McGraw Hill.
3. Introduction to programming and problem solving , G. Michael Schneider ,Wiley India edition.
4. **http://cse.iitkgp.ac.in/~rkumar/pds-vlab/**

Problem Definition:

1. The program takes a 1D array and sorts it in the specified manner. The user enters an element and the same has to be inserted at the correct place in the sorted array.

2. Write a program to find the Transpose of a Matrix.
   - Entered matrix:

```
1   4  0

-5  2   7
```

- Transpose of the matrix:

```
1    -5
4     2

0     7
```

Algorithm:

1)

Print name and roll number.

Declare an integer variable n.

Prompt the user to enter the size of the array (n).

Declare an integer array a of size n+1.

Declare variables i and j for looping.

Prompt the user to enter elements for the array using a loop.

Implement a nested loop for sorting the array in ascending order using the Bubble Sort algorithm.

Print the sorted array.

Prompt the user to enter an element.

Initialize a variable position.

Implement a loop to find the correct position for the new element in the sorted array.

If the element is less than or equal to the first element, set position to 0 and break.

If the element is greater than or equal to the current element and less than or equal to the next element, set position to the next index and break.

If i reaches the second-to-last index, set position to n+1 and break.

Print the correct position of the element.

Shift elements to make space for the new element at the correct position.

Insert the element at its correct position.

Print the array after inserting the element in its correct place.

End.

2)

Print name and roll number.

Declare integer variables rows and cols.

Prompt the user to enter the number of rows.

Prompt the user to enter the number of columns.

Declare a 2D array 'matrix' with dimensions rows x cols.

Declare variables i and j for looping.

Prompt the user to enter elements for the matrix using nested loops.

Print the entered matrix in the form of rows x cols.

Declare a 2D array 'transpose' with dimensions cols x rows.

Transpose the matrix:

- Iterate over each position in the transpose matrix.
- Assign the value of the corresponding position in the original matrix to the transpose matrix.

Print the transpose matrix in the form of cols x rows.

End.

Implementation details:

```
1)#include <stdio.h>

int main() {
   printf("Name: Aksh Maheshwari \nRoll no: 16010123032\n");
   int n;
   printf("Enter the size of the array:");
   scanf("%d",&n);
   int a[n+1];
   int i=0;
   int j=0;
   printf("Enter the elements for the array\n");
   for(i=0;i<n;i++)
   {
      scanf("%d",&a[i]);
   }

   for(i=0;i<n-1;i++)
   {
      for(j=i+1;j<n;j++)
      {
         if(a[i]>a[j])
            {
            int temp=a[i];
            a[i]=a[j];
            a[j]=temp;
         }
      }
   }
   printf("The sorted array is :");
   for(i=0;i<n;i++)
   {
      printf("%d",a[i]);
      printf(" ");
   }
   printf("\n");
```

```
    int element;
    printf("Enter the element:");
    scanf("%d",&element);

    int position;
    for(i=0;i<n-1;i++)
    {
        if(element <= a[0])
        {
            position=0;
            break;
        }
        else if(element>=a[i] && element<=a[i+1])
        {
            position=i+1;
            break;
        }
        else if(i==n-2)
        {
            position=n+1;
            break;
        }
    }

    printf("the correct position of this element is %d",position);
    printf("\n");

     for (i = n; i >= position; i--)
    {
        a[i] = a[i - 1];
    }
    a[position] = element;

    printf("The array after inserting the element in its correct place is :");
    for(i=0;i<n+1;i++){
        printf("%d",a[i]);
        printf(" ");
    }

    return 0;
}
```

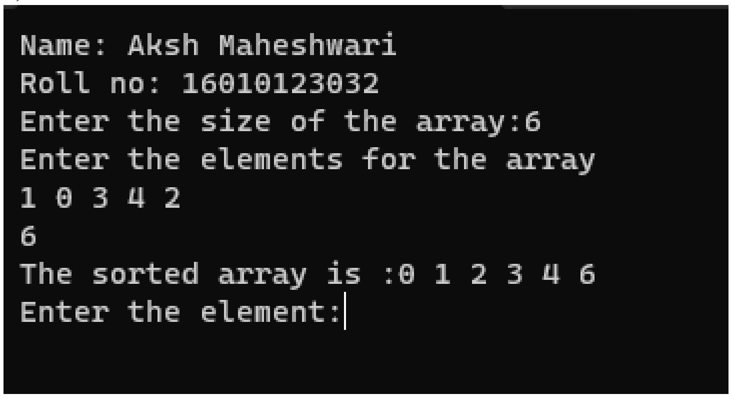
2)

```
#include<stdio.h>
int main() {
   printf("Name: Aksh Maheshwari \n Roll no: 16010123032\n");

   int rows, cols;

   printf("Enter the number of rows: ");
   scanf("%d", &rows);

   printf("Enter the number of columns: ");
   scanf("%d", &cols);

   int matrix[rows][cols];
   int i, j;

   printf("Enter the elements of the matrix:\n");
   for (i = 0; i < rows; i++) {
      for (j = 0; j < cols; j++) {
         printf("Enter element at position (%d, %d): ", i + 1, j + 1);
         scanf("%d", &matrix[i][j]);
      }
   }

   printf("Entered matrix:\n");
   for (i = 0; i < rows; i++) {
      for (j = 0; j < cols; j++)
         printf("%d ", matrix[i][j]);
      printf("\n");
   }

   int transpose[cols][rows];
   for (i = 0; i < cols; i++)
      for (j = 0; j < rows; j++)
         transpose[i][j] = matrix[j][i];

   printf("Transpose of the matrix:\n");
   for (i = 0; i < cols; i++) {
      for (j = 0; j < rows; j++)
         printf("%d ", transpose[i][j]);
      printf("\n");
   }

   return 0;
```

}

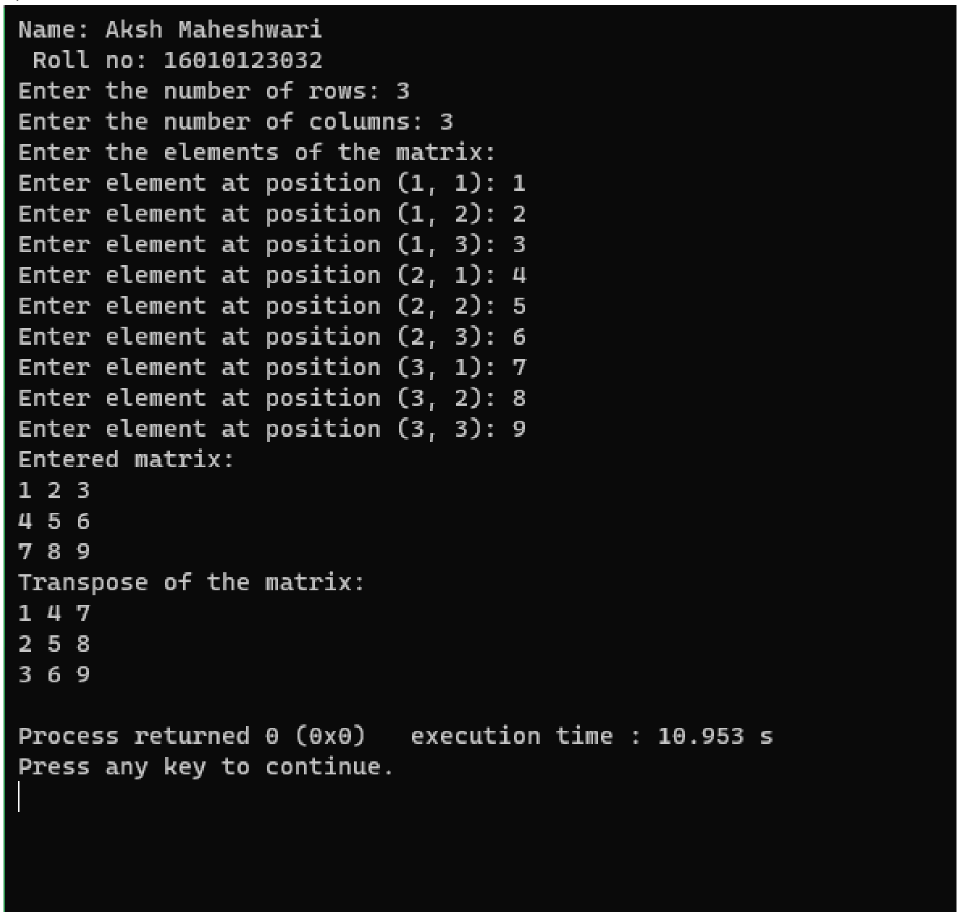
Output(s):

1)



```
Name: Aksh Maheshwari
Roll no: 16010123032
Enter the size of the array:6
Enter the elements for the array
1 0 3 4 2
6
The sorted array is :0 1 2 3 4 6
Enter the element:
```

2)



```
Name: Aksh Maheshwari
 Roll no: 16010123032
Enter the number of rows: 3
Enter the number of columns: 3
Enter the elements of the matrix:
Enter element at position (1, 1): 1
Enter element at position (1, 2): 2
Enter element at position (1, 3): 3
Enter element at position (2, 1): 4
Enter element at position (2, 2): 5
Enter element at position (2, 3): 6
Enter element at position (3, 1): 7
Enter element at position (3, 2): 8
Enter element at position (3, 3): 9
Entered matrix:
1 2 3
4 5 6
7 8 9
Transpose of the matrix:
1 4 7
2 5 8
3 6 9

Process returned 0 (0x0)   execution time : 10.953 s
Press any key to continue.
```

Conclusion:
In conclusion, arrays in C serve as fundamental and versatile data structures, facilitating efficient storage, retrieval, and manipulation of multiple values within a program,enhancing its overall functionality and structure.

Post Lab Questions

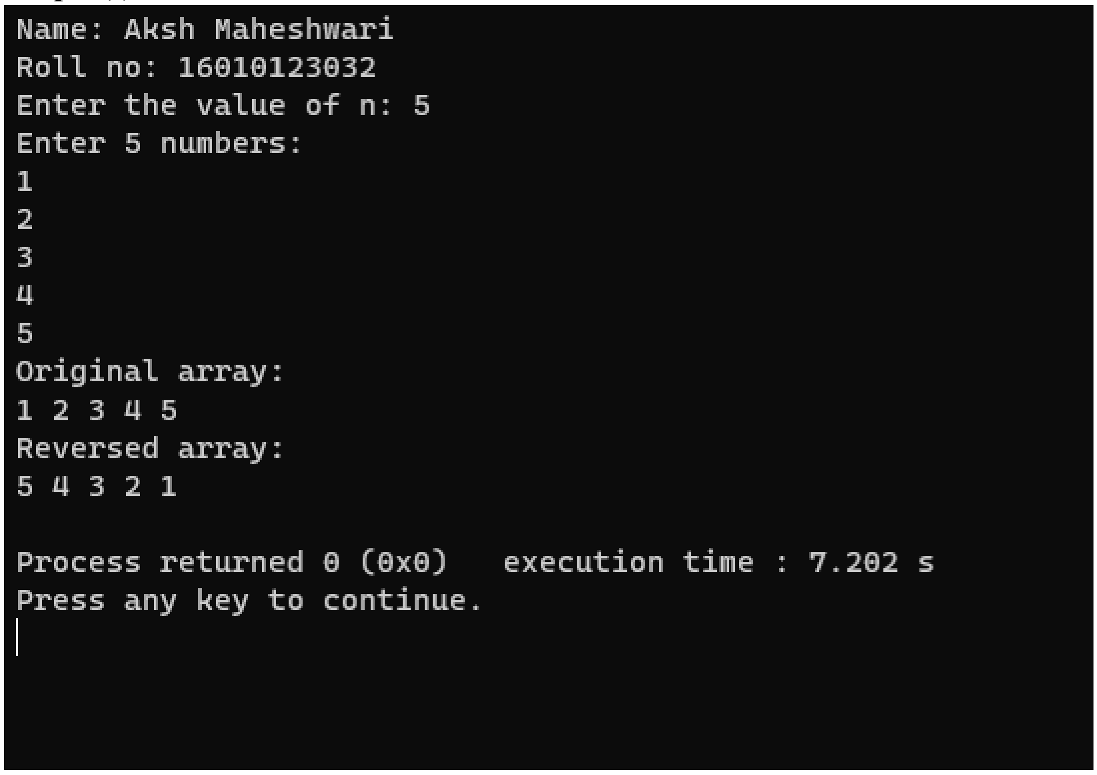
1. Write a program to enter n numbers, store them in an array and rearrange the array in the reverse order.
   Algorithm:

```
Start

Print name and roll number.

Declare an integer variable n.

Prompt the user to enter the value of n.

Declare an integer array of numbers of size n.

Declare variables i and j for looping.

Prompt the user to enter n numbers using a loop.

Print the original array.

Implement a loop to reverse the array using two separate indices, i and j.

Swap the elements at positions i and j until i is less than j.

Print the reversed array.

End
```

   Implementation details:

```
#include <stdio.h>

int main() {
   printf("Name: Aksh Maheshwari  \nRoll no: 16010123032\n");
   printf("Enter the value of n: ");
   int n;
   scanf("%d", &n);

   int numbers[n];
   int i,j;
   printf("Enter %d numbers:\n", n);
   for (i = 0; i < n; i++) {
      scanf("%d", &numbers[i]);
   }

   printf("Original array:\n");
   for (i = 0; i < n; i++) {
      printf("%d ", numbers[i]);
   }
   printf("\n");

   for (i = 0, j = n - 1; i < j; i++, j--) {
      int temp = numbers[i];
      numbers[i] = numbers[j];
      numbers[j] = temp;
   }

   printf("Reversed array:\n");
   for (i = 0; i < n; i++) {
      printf("%d ", numbers[i]);
   }
   printf("\n");

   return 0;
}
```

Output(s):



```
Name: Aksh Maheshwari
Roll no: 16010123032
Enter the value of n: 5
Enter 5 numbers:
1
2
3
4
5
Original array:
1 2 3 4 5
Reversed array:
5 4 3 2 1

Process returned 0 (0x0)   execution time : 7.202 s
Press any key to continue.
```

2. Write a program which performs the following tasks:
a) Initialize an integer array of 10 elements in main( )
b) Pass the entire array to a function modify( )
c) In modify( ) multiply each element of array by 3
d) Return the control to main( ) and print the new array elements in main( )

Algorithm:

Start

Print name and roll number.

Declare a function modify that takes an integer array arr and its size size.

Declare a loop variable j.

In the modify function, iterate through the array and multiply each element by 3.

Declare an integer array of size 10 in main.

Declare a loop variable i.

Prompt the user to enter 10 elements for the array using a loop.

Call the modify function, passing the array and its size.

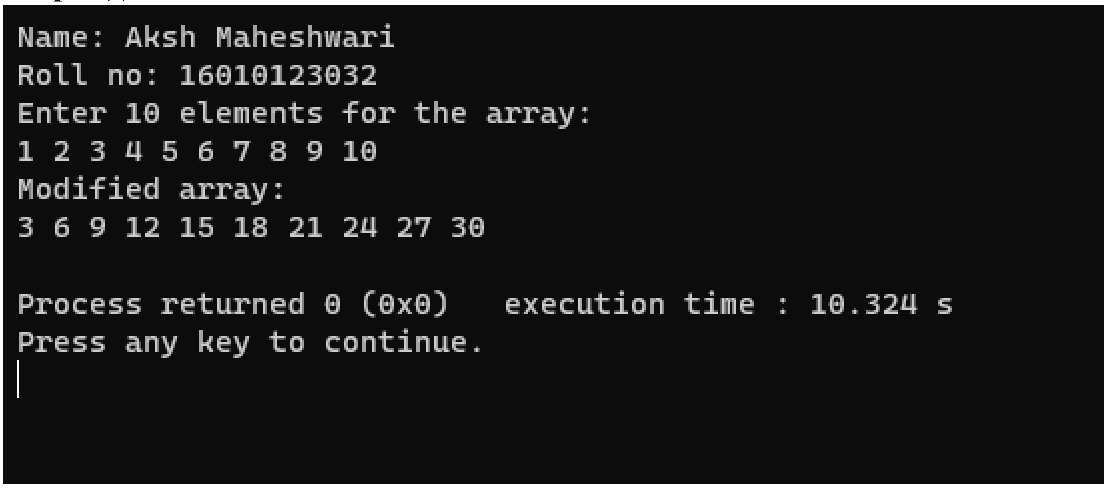
Print the modified array.

End

Implementation details:

```
#include <stdio.h>

void modify(int arr[], int size) {
   int j;
   for (j = 0; j< size; j++) {
      arr[j] *= 3;
   }
}

int main() {
   printf("Name: Aksh Maheshwari  \nRoll no: 16010123032\n");
   int array[10];
   int i;
   printf("Enter 10 elements for the array:\n");
   for (i = 0; i < 10; i++) {
      scanf("%d", &array[i]);
   }

   modify(array, 10);

   printf("Modified array:\n");
   for (i = 0; i < 10; i++) {
      printf("%d ", array[i]);
   }
   printf("\n");

   return 0;
}
```

Output(s):



```
Name: Aksh Maheshwari
Roll no: 16010123032
Enter 10 elements for the array:
1 2 3 4 5 6 7 8 9 10
Modified array:
3 6 9 12 15 18 21 24 27 30

Process returned 0 (0x0)   execution time : 10.324 s
Press any key to continue.
```

Date:30/01/2024 **Signature of faculty in-charge**