

S.No: 1	Exp. Name: <i>Design a C program which sorts the strings using array of pointers</i>	Date: 2023-04-23
---------	---	------------------

Aim:

Design a C program that sorts the strings using array of pointers.

Sample input output

Sample input-output -1:

Enter the number of strings: 2

Enter string 1: Tantra

Enter string 2: Code

Before Sorting

Tantra

Code

After Sorting

Code

Tantra

Sample input-output -2:

Enter the number of strings: 3

Enter string 1: India

Enter string 2: USA

Enter string 3: Japan

Before Sorting

India

USA

Japan

After Sorting

India

Japan

USA

ID: 224G1A0535 Page No: 1

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

Source Code:

stringssort.c

```

#include<stdio.h>
#include<stdlib.h>
#include<string.h>
void main() {
    char*temp;
    int i, j, diff, num_strings;
    char * strArray[10];
    printf("Enter the number of strings: ");
    scanf("%d",&num_strings);
    for(i = 0 ;i < num_strings ;i++) {
        strArray[i] = (char *)malloc(sizeof(char)*20);
        printf("Enter string %d: ",i+1);
        scanf("%s",strArray[i]);
    }
    printf("Before Sorting\n");
    for(i = 0;i < num_strings ; i++) {
        printf("%s\n",strArray[i]);
    }
    for(i = 0;i < num_strings-1 ; i++) {
        for(j = 0; j < num_strings-1-i; j++) {
            diff = strcmp(strArray[j], strArray[j+1]);
            if(diff>0) {
                temp = strArray[j];
                strArray[j] = strArray[j+1];
                strArray[j+1] = temp;
            }
        }
    }
    printf("After Sorting\n");
    for(i=0;i<num_strings;i++) {
        printf("%s\n",strArray[i]);
    }
}

```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Enter the number of strings:
2
Enter string 1:
Tantra
Enter string 2:
Code
Before Sorting
Tantra
Code
After Sorting
Code
Tantra

Test Case - 2

User Output

Enter the number of strings:

3

Enter string 1:

Dhoni

Enter string 2:

Kohli

Enter string 3:

Rohit

Before Sorting

Dhoni

Kohli

Rohit

After Sorting

Dhoni

Kohli

Rohit

ID: 224G1A0535 Page No: 3

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

S.No: 2

Exp. Name: **Write a C program to Search a Key element using Linear search Technique**

Date: 2023-04-25

Aim:

Write a program to search a **key element** with in the given array of elements using [linear search](#) process.

At the time of execution, the program should print the message on the console as:

Enter value of n :

For example, if the user gives the **input** as:

Enter value of n : 3

Next, the program should print the messages one by one on the console as:

Enter element for a[0] :
Enter element for a[1] :
Enter element for a[2] :

if the user gives the **input** as:

Enter element for a[0] : 89
Enter element for a[1] : 33
Enter element for a[2] : 56

Next, the program should print the message on the console as:

Enter key element :

if the user gives the **input** as:

Enter key element : 56

then the program should **print** the result as:

The key element 56 is found at the position 2

Similarly if the key element is given as **25** for the above one dimensional array elements then the program should print the output as "**The key element 25 is not found in the array**".

Fill in the missing code so that it produces the desired result.

Source Code:

LinearSearch.c

ID: 224G1A0535 Page No: 4

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

```

#include<stdio.h>
void main()
{
    int a[10],i,n,j,flag = 0,pos;
    printf("Enter value of n : ");
    scanf("%d",&n);
    for(i=0;i<n;i++)
    {
        printf("Enter element for a[%d] : ",i);
        scanf("%d",&a[i]);
    }
    printf("Enter key element : ");
    scanf("%d",&j);
    for(i=0;i<n;i++)
    {
        if(j==a[i])
        {
            flag = 1;
            pos = i;
            break ;
        }
    }
    if(flag==1)
    {
        printf("The key element %d is found at the position %d\n",j,i);
    }
    else
    {
        printf("The key element %d is not found in the array\n",j);
    }
}

```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Enter value of n :
4
Enter element for a[0] :
1
Enter element for a[1] :
22
Enter element for a[2] :
33
Enter element for a[3] :
44
Enter key element :
22
The key element 22 is found at the position 1

Test Case - 2

User Output

Enter value of n :

7

Enter element for a[0] :

101

Enter element for a[1] :

102

Enter element for a[2] :

103

Enter element for a[3] :

104

Enter element for a[4] :

105

Enter element for a[5] :

106

Enter element for a[6] :

107

Enter key element :

110

The key element 110 is not found in the array

S.No: 3

Exp. Name: **Write a C program to Search a Key element using Binary search Technique**

Date: 2023-04-25

Aim:

Write a program to **search** a key element in the given array of elements using [binary search](#).

At the time of execution, the program should print the message on the console as:

Enter value of n :

For example, if the user gives the **input** as:

Enter value of n : 3

Next, the program should print the messages one by one on the console as:

Enter element for a[0] :
Enter element for a[1] :
Enter element for a[2] :

if the user gives the **input** as:

Enter element for a[0] : 89
Enter element for a[1] : 33
Enter element for a[2] : 56

Next, the program should print the message on the console as:

Enter key element :

if the user gives the **input** as:

Enter key element : 56

then the program should **print** the result as:

After sorting the elements in the array are
Value of a[0] = 33
Value of a[1] = 56
Value of a[2] = 89
The key element 56 is found at the position 1

Similarly if the key element is given as **25** for the above one dimensional array elements then the program should print the output as "**The Key element 25 is not found in the array**".

Fill in the missing code so that it produces the desired result.

Source Code:

BinarySearch.c

ID: 224G1A0535 Page No: 7

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

```

#include<stdio.h>
void main()
{
    int a[5],i,j,temp,k,n,flag = 0;
    printf("Enter value of n : ");
    scanf("%d",&n);
    for(i=0;i<n;i++)
    {
        printf("Enter element for a[%d] : ",i);
        scanf("%d",&a[i]);
    }
    for(i=0;i<n-1;i++)
    {
        for(j=i+1;j<n;j++)
        {
            if(a[j]<a[i])
            {
                temp = a[i];
                a[i] = a[j];
                a[j] = temp;
            }
        }
    }
    printf("Enter key element : ");
    scanf("%d",&k);
    printf("After sorting the elements in the array are\n");
    for(i=0;i<n;i++)
    {
        printf("Value of a[%d] = %d\n",i,a[i]);
    }
    for(i=0;i<n;i++)
    {
        if(k == a[i])
        {
            flag++;
            break ;
        }
    }
    if(flag == 1)
    {
        printf("The key element %d is found at the position %d\n",k,i);
    }
    else
    {
        printf("The Key element %d is not found in the array\n",k);
    }
}
}

```

Execution Results - All test cases have succeeded!

Test Case - 1	
User Output	
Enter value of n :	

3
Enter element for a[0] :
25
Enter element for a[1] :
15
Enter element for a[2] :
23
Enter key element :
45
After sorting the elements in the array are
Value of a[0] = 15
Value of a[1] = 23
Value of a[2] = 25
The Key element 45 is not found in the array

ID: 224G1A0535 Page No: 9

Test Case - 2
User Output
Enter value of n :
2
Enter element for a[0] :
80
Enter element for a[1] :
39
Enter key element :
50
After sorting the elements in the array are
Value of a[0] = 39
Value of a[1] = 80
The Key element 50 is not found in the array

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

S.No: 4

Exp. Name: **Write a C program to implement Fibonacci Search technique**

Date: 2023-04-27

Aim:

Write a C program to implement **Fibonacci search** technique

Source Code:

FibonacciSearch.c

```
#include<stdio.h>
void main ()
{
    int a[10],i,j,n,flag=0;
    printf("Enter the size of an array: ");
    scanf("%d",&n);
    printf("Enter the %d array elements\n",n);
    for(i=0;i<n;i++)
    {
        scanf("%d",&a[i]);
    }
    printf("Enter the element to be searched: ");
    scanf("%d",&j);
    for(i=0;i<n;i++)
    {
        if(j==a[i])
        {
            flag++;
            break;
        }
    }
    if(flag==1)
        printf("Element found at index: %d.\n",i);
    else
        printf("Element not found.\n");
}
```

ID: 224G1A0535 Page No: 10

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

Execution Results - All test cases have succeeded!

Test Case - 1

User Output

Enter the size of an array:

5

Enter the 5 array elements

3 4 5 6 7

Enter the element to be searched:

3

Element found at index: 0.

Test Case - 2

User Output

Enter the size of an array:

5

Enter the 5 array elements

3 4 5 6 7

Enter the element to be searched:

4

Element found at index: 1.

S.No: 5

Exp. Name: **Write a C program to Sort the elements using Insertion Sort Technique**

Date: 2023-04-27

Aim:

Write a program to **sort** the given elements using [insertion sort technique](#).

At the time of execution, the program should print the message on the console as:

Enter value of n :

For example, if the user gives the **input** as:

Enter value of n : 3

Next, the program should print the messages one by one on the console as:

Enter element for a[0] :
Enter element for a[1] :
Enter element for a[2] :

if the user gives the **input** as:

Enter element for a[0] : 22
Enter element for a[1] : 33
Enter element for a[2] : 12

then the program should **print** the result as:

Before sorting the elements in the array are
Value of a[0] = 22
Value of a[1] = 33
Value of a[2] = 12
After sorting the elements in the array are
Value of a[0] = 12
Value of a[1] = 22
Value of a[2] = 33

Fill in the missing code so that it produces the desired result.

Source Code:

InsertionSortDemo3.c

ID: 224G1A0535

Page No: 12

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

```

#include<stdio.h>
void sort(int [],int);
void main()
{
    int a[20],n,i;
    printf("Enter value of n : ");
    scanf("%d",&n);
    for(i=0;i<n;i++)
    {
        printf("Enter element for a[%d] : ",i);
        scanf("%d",&a[i]);
    }
    printf("Before sorting the elements in the array are\n");
    for(i=0;i<n;i++)
    {
        printf("Value of a[%d] = %d\n",i,a[i]);
    }
    sort(a,n);
    printf("After sorting the elements in the array are\n");
    for(i=0;i<n;i++)
    {
        printf("Value of a[%d] = %d\n",i,a[i]);
    }
}
void sort (int a[],int n)
{
    int i,j,k;
    for(i=0;i<n;i++)
    {
        k=a[i];
        j=i-1;
        while(j>=0&&a[j]>k)
        {
            a[j+1]=a[j];
            j=j-1;
        }
        a[j+1]=k;
    }
}

```

ID: 224G1A0535 | Page No: 13

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Enter value of n :
6
Enter element for a[0] :
5
Enter element for a[1] :
9
Enter element for a[2] :

```
Enter element for a[3] :  
5  
Enter element for a[4] :  
1  
Enter element for a[5] :  
3  
Before sorting the elements in the array are  
Value of a[0] = 5  
Value of a[1] = 9  
Value of a[2] = 2  
Value of a[3] = 5  
Value of a[4] = 1  
Value of a[5] = 3  
After sorting the elements in the array are  
Value of a[0] = 1  
Value of a[1] = 2  
Value of a[2] = 3  
Value of a[3] = 5  
Value of a[4] = 5  
Value of a[5] = 9
```

ID: 224G1A0535 Page No: 14

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

Test Case - 2

User Output

```
Enter value of n :  
3  
Enter element for a[0] :  
5  
Enter element for a[1] :  
9  
Enter element for a[2] :  
4  
Before sorting the elements in the array are  
Value of a[0] = 5  
Value of a[1] = 9  
Value of a[2] = 4  
After sorting the elements in the array are  
Value of a[0] = 4  
Value of a[1] = 5  
Value of a[2] = 9
```

S.No: 6	Exp. Name: <i>Write a C program to Sort the elements using Selection Sort - Smallest element method Technique</i>	Date: 2023-04-27
---------	--	------------------

Aim:

Write a program to **sort** the given array elements using **selection sort smallest element** method.

At the time of execution, the program should print the message on the console as:

Enter value of n :

For example, if the user gives the **input** as:

Enter value of n : 3

Next, the program should print the messages one by one on the console as:

Enter element for a[0] :
Enter element for a[1] :
Enter element for a[2] :

if the user gives the **input** as:

Enter element for a[0] : 22
Enter element for a[1] : 33
Enter element for a[2] : 12

then the program should **print** the result as:

Before sorting the elements in the array are
Value of a[0] = 22
Value of a[1] = 33
Value of a[2] = 12
After sorting the elements in the array are
Value of a[0] = 12
Value of a[1] = 22
Value of a[2] = 33

Fill in the missing code so that it produces the desired result.

Source Code:

SelectionSortDemo6.c

```

#include<stdio.h>
void main()
{
    int a[20],i,j,n,max,temp=0;
    printf("Enter value of n : ");
    scanf("%d",&n);
    for(i=0;i<n;i++)
    {
        printf("Enter element for a[%d] : ",i);
        scanf("%d",&a[i]);
    }
    printf("Before sorting the elements in the array are\n");
    for(i=0;i<n;i++)
    {
        printf("Value of a[%d] = %d\n",i,a[i]);
    }
    for(i=n-1;i>0;i--)
    {
        max=1;
        for(j=i;j>=0;j--)
        {
            if(a[j]>=a[max])
            {
                max=j;
            }
        }
        temp=a[i];
        a[i]=a[max];
        a[max]=temp;
    }
    printf("After sorting the elements in the array are\n");
    for(i=0;i<n;i++)
    {
        printf("Value of a[%d] = %d\n",i,a[i]);
    }
}

```

ID: 224G1A0535 | Page No: 16

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Enter value of n :
4
Enter element for a[0] :
78
Enter element for a[1] :
43
Enter element for a[2] :
99
Enter element for a[3] :
27

Before sorting the elements in the array are

Value of a[0] = 78

Value of a[1] = 43

Value of a[2] = 99

Value of a[3] = 27

After sorting the elements in the array are

Value of a[0] = 27

Value of a[1] = 43

Value of a[2] = 78

Value of a[3] = 99

S.No: 7

Exp. Name: **Write a C program to sort given elements using shell sort technique.**

Date: 2023-04-29

Aim:

Write a program to **sort** (**ascending order**) the given elements using **shell sort** technique.

At the time of execution, the program should print the message on the console as:

Enter array size :

For example, if the user gives the **input** as:

Enter array size : 5

Next, the program should print the following message on the console as:

Enter 5 elements :

if the user gives the **input** as:

Enter 5 elements : 34 67 12 45 22

then the program should **print** the result as:

Before sorting the elements are : 34 67 12 45 22

After sorting the elements are : 12 22 34 45 67

Note: Do use the **printf()** function with a **newline** character (**\n**).

Source Code:

ShellSort2.c

```

#include<stdio.h>
#include<conio.h>
int main()
{
    int size;
    int *arr,i;
    printf("Enter array size : ");
    scanf("%d",&size);
    arr=(int*)malloc(size*sizeof(int));
    printf("Enter %d elements : ",size);
    for(i=0;i<size;i++)
    {
        scanf("%d",&arr[i]);
    }
    printf("Before sorting the elements are : ");
    printArray(arr,size);
    ShellSort(arr,size);
    printf("After sorting the elements are : ");
    printArray(arr,size);
    return 0;
}
int ShellSort(int arr[],int n)
{
    int gap,i,j,temp;
    for(gap=n/2;gap>0;gap/=2)
    {
        for(i=gap;i<n;i++)
        {
            temp=arr[i];
            for(j=i;j>=gap&&arr[j-gap]>temp;j-=gap)
                arr[j]=arr[j-gap];
            arr[j]=temp;
        }
    }
}
void printArray(int arr[],int n)
{
    for(int i=0;i<n;i++)
    {
        printf("%d ",arr[i]);
    }
    printf("\n");
}

```

ID: 224G1A0535 | Page No: 19

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Enter array size :
5
Enter 5 elements :
12 32 43 56 78

Before sorting the elements are : 12 32 43 56 78

After sorting the elements are : 12 32 43 56 78

S.No: 8

Exp. Name: **Write a C program to Sort the elements using Bubble Sort Technique**

Date: 2023-04-29

Aim:

Write a program to **sort** the given elements using [bubble sort technique](#).

At the time of execution, the program should print the message on the console as:

Enter value of n :

For example, if the user gives the **input** as:

Enter value of n : 3

Next, the program should print the messages one by one on the console as:

Enter element for a[0] :
Enter element for a[1] :
Enter element for a[2] :

if the user gives the **input** as:

Enter element for a[0] : 22
Enter element for a[1] : 33
Enter element for a[2] : 12

then the program should **print** the result as:

Before sorting the elements in the array are
Value of a[0] = 22
Value of a[1] = 33
Value of a[2] = 12
After sorting the elements in the array are
Value of a[0] = 12
Value of a[1] = 22
Value of a[2] = 33

Fill in the missing code so that it produces the desired result.

Source Code:

BubbleSortDemo3.c

ID: 224G1A0535 | Page No: 21

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

```

#include<stdio.h>
void main()
{
    int a[20],i,n,j,temp;
    printf("Enter value of n : ");
    scanf("%d",&n);
    for(i=0;i<n;i++)
    {
        printf("Enter element for a[%d] : ",i);
        scanf("%d",&a[i]);
    }
    printf("Before sorting the elements in the array are\n");
    for(i=0;i<n;i++)
    {
        printf("Value of a[%d] = %d",i,a[i]);
        printf("\n");
    }
    for(i=0;i<n;i++)
    {
        for(j=i+1;j<n;j++)
        {
            if(a[j]<a[i])
            {
                temp=a[i];
                a[i]=a[j];
                a[j]=temp;
            }
        }
    }
    printf("After sorting the elements in the array are\n");
    for(i=0;i<n;i++)
    {
        printf("Value of a[%d] = %d",i,a[i]);
        printf("\n");
    }
}

```

Execution Results - All test cases have succeeded!

Test Case - 1	
User Output	
Enter value of n :	
3	
Enter element for a[0] :	
34	
Enter element for a[1] :	
25	
Enter element for a[2] :	
28	
Before sorting the elements in the array are	
Value of a[0] = 34	

Value of a[1] = 25

Value of a[2] = 28

After sorting the elements in the array are

Value of a[0] = 25

Value of a[1] = 28

Value of a[2] = 34

Test Case - 2

User Output

Enter value of n :

5

Enter element for a[0] :

1

Enter element for a[1] :

6

Enter element for a[2] :

3

Enter element for a[3] :

8

Enter element for a[4] :

4

Before sorting the elements in the array are

Value of a[0] = 1

Value of a[1] = 6

Value of a[2] = 3

Value of a[3] = 8

Value of a[4] = 4

After sorting the elements in the array are

Value of a[0] = 1

Value of a[1] = 3

Value of a[2] = 4

Value of a[3] = 6

Value of a[4] = 8

S.No: 9	Exp. Name: <i>Write a program to sort Ascending order the given elements using quick sort technique.</i>	Date: 2023-05-02
---------	---	------------------

Aim:

Write a program to **sort** (**Ascending order**) the given elements using **quick sort** technique.

Note: Pick the first element as pivot. You will not be awarded marks if you do not follow this instruction.

At the time of execution, the program should print the message on the console as:

Enter array size :

For example, if the user gives the **input** as:

Enter array size : 5

Next, the program should print the following message on the console as:

Enter 5 elements :

if the user gives the **input** as:

Enter 5 elements : 34 67 12 45 22

then the program should **print** the result as:

Before sorting the elements are : 34 67 12 45 22
After sorting the elements are : 12 22 34 45 67

Note: Do use the **printf()** function with a **newline** character (**\n**).

Source Code:

QuickSortMain.c

```

#include<stdio.h>
void sort(int [],int ,int );
void main()
{
    int arr[50],i,n;
    printf("Enter array size : ");
    scanf("%d",&n);
    printf("Enter %d elements : ",n);
    for(i=0;i<n;i++)
    {
        scanf("%d",&arr[i]);
    }
    printf("Before sorting the elements are : ");
    for(i=0;i<n;i++)
    {
        printf("%d ",arr[i]);
    }
    sort(arr,0,n-1);
    printf("\nAfter sorting the elements are : ");
    for(i=0;i<n;i++)
    {
        printf("%d ",arr[i]);
    }
    printf("\n");
}
void sort(int a[20],int low,int high)
{
    int left,right,pivolt,temp;
    left=low;
    right=high;
    pivolt=a[(low+high)/2];
    do
    {
        while(a[left]<pivolt)
            left++;
        while(a[right]>pivolt)
            right--;
        if(left<=right)
        {
            temp=a[left];
            a[left]=a[right];
            a[right]=temp;
            right--;
            left++;
        }
    }
    while(left<=right);
    if(low<right)
        sort(a,low,right);
    if(left<high)
        sort(a,left,high);
}

```

Execution Results - All test cases have succeeded!

Test Case - 1

User Output

Enter array size :

5

Enter 5 elements :

34 67 12 45 22

Before sorting the elements are : 34 67 12 45 22

After sorting the elements are : 12 22 34 45 67

Page No: 26

ID: 224G1A0535

Test Case - 2

User Output

Enter array size :

8

Enter 8 elements :

77 55 22 44 99 33 11 66

Before sorting the elements are : 77 55 22 44 99 33 11 66

After sorting the elements are : 11 22 33 44 55 66 77 99

CSE-A

Srinivasa Ramanujan Institute of Technology
2022-2026-CSE-A

Test Case - 3

User Output

Enter array size :

5

Enter 5 elements :

-32 -45 -67 -46 -14

Before sorting the elements are : -32 -45 -67 -46 -14

After sorting the elements are : -67 -46 -45 -32 -14

S.No: 10

Exp. Name: ***Write a C program to sort the given elements using Heap sort***

Date: 2023-05-02

Aim:

Write a program to sort (ascending order) the given elements using heap sort technique.

Note: Do use the printf() function with a newline character (\n).

Source Code:

HeapSortMain.c

ID: 224G1A0535 | Page No: 27

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

```

#include<stdio.h>
void main()
{
    int arr[15],i,n;
    printf("Enter array size : ");
    scanf("%d",&n);
    printf("Enter %d elements : ",n);
    for(i=0;i<n;i++)
    {
        scanf("%d",&arr[i]);
    }
    printf("Before sorting the elements are : ");
    display(arr,n);
    heapsort(arr,n);
    printf("After sorting the elements are : ");
    display(arr,n);
}
void display(int arr[15],int n)
{
    int i;
    for(i=0;i<n;i++)
        printf("%d ",arr[i]);
    printf("\n");
}
void heapify(int arr[],int n,int i)
{
    int largest=i;
    int l=2*i+1;
    int r=2*i+2;
    int temp;
    if(l<n&&arr[l]>arr[largest])
        largest=l;
    if(r<n&&arr[r]>arr[largest])
        largest=r;
    if(largest!=i)
    {
        temp=arr[i];
        arr[i]=arr[largest];
        arr[largest]=temp;
        heapify(arr,n,largest);
    }
}
void heapsort(int arr[],int n)
{
    int i,temp;
    for(i=n/2-1;i>=0;i--)
    {
        heapify(arr,n,i);
    }
    for(i=n-1;i>=0;i--)
    {
        temp=arr[0];
        arr[0]=arr[i];
        arr[i]=temp;
        heapify(arr,i,0);
    }
}

```

```
    }  
}
```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Enter array size :
5
Enter 5 elements :
23 54 22 44 12
Before sorting the elements are : 23 54 22 44 12
After sorting the elements are : 12 22 23 44 54

Test Case - 2
User Output
Enter array size :
6
Enter 6 elements :
12 65 23 98 35 98
Before sorting the elements are : 12 65 23 98 35 98
After sorting the elements are : 12 23 35 65 98 98

Test Case - 3
User Output
Enter array size :
4
Enter 4 elements :
-23 -45 -12 -36
Before sorting the elements are : -23 -45 -12 -36
After sorting the elements are : -45 -36 -23 -12

Test Case - 4
User Output
Enter array size :
6
Enter 6 elements :
1 -3 8 -4 -2 5
Before sorting the elements are : 1 -3 8 -4 -2 5
After sorting the elements are : -4 -3 -2 1 5 8

S.No: 11

Exp. Name: **Write a C program to Sort given elements using Merge sort**

Date: 2023-06-15

Aim:

Write a program to **sort** (**Ascending order**) the given elements using **merge sort** technique.

At the time of execution, the program should print the message on the console as:

Enter array size :

For example, if the user gives the **input** as:

Enter array size : 5

Next, the program should print the following message on the console as:

Enter 5 elements :

if the user gives the **input** as:

Enter 5 elements : 34 67 12 45 22

then the program should **print** the result as:

Before sorting the elements are : 34 67 12 45 22
After sorting the elements are : 12 22 34 45 67

Note: Do use the **printf()** function with a **newline** character (**\n**).

Source Code:

MergeSortMain.c

ID: 224G1A0535 Page No: 30

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

```

#include <stdio.h>
void main()
{
    int arr[15],i,n;
    printf("Enter array size : ");
    scanf("%d",&n);
    printf("Enter %d elements : ",n);
    for(i=0;i<n;i++)
    {
        scanf("%d",&arr[i]);
    }
    printf("Before sorting the elements are : ");
    display(arr,n);
    splitAndMerge(arr,0,n-1);
    printf("After sorting the elements are : ");
    display(arr,n);
}
void display(int arr[15],int n)
{
    int i;
    for(i=0;i<n;i++)
    printf("%d ",arr[i]);
    printf("\n");
}
void merge(int arr[15],int low,int mid,int high)
{
    int i=low,h=low,j=mid+1,k,temp[15];
    while(h<=mid && j<=high)
    {
        if(arr[h]<=arr[j])
        {
            temp[i] = arr[h];
            h++;
        }
        else
        {
            temp[i]=arr[j];
            j++;
        }
        i++;
    }
    if(h>mid)
    {
        for(k=j;k<=high;k++)
        {
            temp[i] = arr[k];
            i++;
        }
    }
    else
    {
        for(k=h;k<=mid;k++)
        {
            temp[i] = arr[k];
            i++;
        }
    }
}

```

```

for(k=low;k<=high;k++)
{
    arr[k] = temp[k];
}
}
void splitAndMerge(int arr[15], int low, int high)
{
    if(low<high)
    {
        int mid=(low+high)/2;
        splitAndMerge(arr, low, mid);
        splitAndMerge(arr,mid+1,high);
        merge(arr,low,mid,high);
    }
}

```

Execution Results - All test cases have succeeded!

Test Case - 1

User Output

Enter array size :

5

Enter 5 elements :

34 67 12 45 22

Before sorting the elements are : 34 67 12 45 22

After sorting the elements are : 12 22 34 45 67

Test Case - 2

User Output

Enter array size :

8

Enter 8 elements :

77 55 22 44 99 33 11 66

Before sorting the elements are : 77 55 22 44 99 33 11 66

After sorting the elements are : 11 22 33 44 55 66 77 99

Test Case - 3

User Output

Enter array size :

5

Enter 5 elements :

-32 -45 -67 -46 -14

Before sorting the elements are : -32 -45 -67 -46 -14

After sorting the elements are : -67 -46 -45 -32 -14

S.No: 12

Exp. Name: **Write a C program to sort given elements using Radix sort**

Date: 2023-05-09

Aim:

Write a program to **sort** (**ascending order**) the given elements using **radix sort** technique.

At the time of execution, the program should print the message on the console as:

Enter array size :

For example, if the user gives the **input** as:

Enter array size : 5

Next, the program should print the following message on the console as:

Enter 5 elements :

if the user gives the **input** as:

Enter 5 elements : 34 67 12 45 22

then the program should **print** the result as:

Before sorting the elements are : 34 67 12 45 22
After sorting the elements are : 12 22 34 45 67

Note: Do use the **printf()** function with a **newline** character (**\n**).

Source Code:

RadixSortMain2.c

ID: 224G1A0535 | Page No: 33

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

```

#include <stdio.h>
#include <conio.h>
int main() {
    int size;
    int *arr, i;
    printf("Enter array size : ");
    scanf("%d",&size);
    arr = (int*) malloc(size * sizeof(int));
    printf("Enter %d elements : ",size);
    for (i = 0; i < size; i++) {
        scanf("%d", &arr[i]);
    }
    printf("Before sorting the elements are : ");
    printArray(arr,size);
    RadixSort(arr,size);
    printf("After sorting the elements are : ");
    printArray(arr,size);
    return 0;
}
int largest(int a[], int n) {
    int i, k = a[0];
    for(i=1;i<n;i++) {
        if(a[i]>k) {
            k = a[i];
        }
    }
    return k;
}
void printArray(int a[], int n) {
    int i, k = a[0];
    for(i=0;i<n;i++) {
        printf("%d ",a[i]);
    }
    printf("\n");
}
void RadixSort(int a[], int n) {
    int bucket[10][10],bucket_count[10],i,j,k,rem,NOP=0,divi=1,large,pass;
    large=largest(a,n);
    while(large>0) {
        NOP++;
        large/=10;
    }
    for(pass=0;pass<NOP;pass++) {
        for(i=0;i<10;i++) {
            bucket_count[i] = 0;
        }
        for(i=0;i<n;i++) {
            rem = (a[i]/divi)%10;
            bucket[rem][bucket_count[rem]] = a[i];
            bucket_count[rem]++;
        }
        i=0;
        for(k=0;k<10;k++) {
            for(j=0;j<bucket_count[k];j++) {
                a[i] = bucket[k][j];
                i++;
            }
        }
    }
}

```

```
        }
        divi*=10;
    }
}
```

Execution Results - All test cases have succeeded!

Test Case - 1	
User Output	
Enter array size :	
5	
Enter 5 elements :	
23	
43	
54	
12	
65	
Before sorting the elements are : 23 43 54 12 65	
After sorting the elements are : 12 23 43 54 65	

Test Case - 2	
User Output	
Enter array size :	
7	
Enter 7 elements :	
23	
54	
136	
85	
24	
65	
76	
Before sorting the elements are : 23 54 136 85 24 65 76	
After sorting the elements are : 23 24 54 65 76 85 136	

S.No: 13

Exp. Name: **C program to performs all operations
on singly linked list**

Date: 2023-06-13

Aim:

Write a program that uses functions to perform the following **operations on singly linked list**

- i) Creation
- ii) Insertion
- iii) Deletion
- iv) Traversal

Source Code:

```
singlelinkedlistalloperations.c
```

ID: 224G1A0535 | Page No: 36

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

```

#include<stdio.h>
#include<stdlib.h>
void menu()
{
    printf("Options\n");
    printf("1 : Insert elements into the linked list\n");
    printf("2 : Delete elements from the linked list\n");
    printf("3 : Display the elements in the linked list\n");
    printf("4 : Count the elements in the linked list\n");
    printf("5 : Exit()\n");

}

struct node
{
    int data;
    struct node *next;
};

typedef struct node node;
struct node *head=NULL;
node* createnode(int data)
{
    node* temp=(node*)malloc(sizeof(node));
    temp->data=data;
    temp->next=NULL;
    return temp;
}
void insert(int data)
{
    node* newnode=createnode(data);
    node* temp;
    if(head==NULL)
    {
        head=createnode(data);
    }
    else
    {
        temp=head;
        while(temp->next!=NULL)
        {
            temp=temp->next;
        }
        temp->next=newnode;
    }
}
void delete(int position)
{
    int i;
    node* temp;
    if(head==NULL)
    {
        printf("List is empty");
    }
    else
    {
        temp=head;
    }
}

```

```

        temp=temp->next;
    }
    temp->next=temp->next->next;
    printf("Deleted successfully\n");
}
void display()
{
    node* temp;
    temp=head;
    if(head==NULL)
    {
        printf("List is empty\n");
    }
    while(temp!=NULL)
    {
        printf("%d ",temp->data);
        temp=temp->next;
    }
    printf("\n");
}
void count()
{
    int c=0;
    node * temp;
    if(head==NULL)
    {
        printf("List is Empty\n");
    }
    else
    {
        temp=head;
        while(temp!=NULL)
        {
            c++;
            temp=temp->next;
        }
    }
    printf("No of elements in the linked list are : %d\n",c);
}
void main()
{
    int choice,data,position,c;
    printf("Singly Linked List Example - All Operations\n");
    menu();
    printf("Enter your option : ");
    scanf("%d",&choice);
    while(choice!=5)
    {
        switch(choice)
        {
            case 1:
            {
                printf("Enter elements for inserting into linked list : ");
                scanf("%d",&data);

```

```

    }
    case 2:
    {
        printf("Enter position of the element for deleteing the
element : ");
        scanf("%d",&position);
        delete(position);
        break;
    }
    case 3:
    {
        printf("The elements in the linked list are : ");
        display();
        break;
    }
    case 4:
    {
        count();
        break;
    }
    case 5:
    {
        exit(0);
    }
    default:
    {
        printf("Enter options from 1 to 5\n");
        exit(0);
    }
}
menu();
printf("Enter your option : ");
scanf("%d",&choice);
}
}

```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Singly Linked List Example - All Operations
Options
1 : Insert elements into the linked list
2 : Delete elements from the linked list
3 : Display the elements in the linked list
4 : Count the elements in the linked list
5 : Exit()
Enter your option :
1
Enter elements for inserting into linked list :

Options
1 : Insert elements into the linked list
2 : Delete elements from the linked list
3 : Display the elements in the linked list
4 : Count the elements in the linked list
5 : Exit()
Enter your option :
1
Enter elements for inserting into linked list :
222
Options
1 : Insert elements into the linked list
2 : Delete elements from the linked list
3 : Display the elements in the linked list
4 : Count the elements in the linked list
5 : Exit()
Enter your option :
1
Enter elements for inserting into linked list :
333
Options
1 : Insert elements into the linked list
2 : Delete elements from the linked list
3 : Display the elements in the linked list
4 : Count the elements in the linked list
5 : Exit()
Enter your option :
1
Enter elements for inserting into linked list :
444
Options
1 : Insert elements into the linked list
2 : Delete elements from the linked list
3 : Display the elements in the linked list
4 : Count the elements in the linked list
5 : Exit()
Enter your option :
3
The elements in the linked list are : 111 222 333 444
Options
1 : Insert elements into the linked list
2 : Delete elements from the linked list
3 : Display the elements in the linked list
4 : Count the elements in the linked list
5 : Exit()
Enter your option :
2
Enter position of the element for deleteing the element :
2
Deleted successfully

```

1 : Insert elements into the linked list
2 : Delete elements from the linked list
3 : Display the elements in the linked list
4 : Count the elements in the linked list
5 : Exit()
Enter your option :
3
The elements in the linked list are : 111 333 444
Options
1 : Insert elements into the linked list
2 : Delete elements from the linked list
3 : Display the elements in the linked list
4 : Count the elements in the linked list
5 : Exit()
Enter your option :
4
No of elements in the linked list are : 3
Options
1 : Insert elements into the linked list
2 : Delete elements from the linked list
3 : Display the elements in the linked list
4 : Count the elements in the linked list
5 : Exit()
Enter your option :
5

```

ID: 224G1A0535 Page No: 41

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

Test Case - 2

User Output

```

Singly Linked List Example - All Operations
Options
1 : Insert elements into the linked list
2 : Delete elements from the linked list
3 : Display the elements in the linked list
4 : Count the elements in the linked list
5 : Exit()
Enter your option :
1
Enter elements for inserting into linked list :
001
Options
1 : Insert elements into the linked list
2 : Delete elements from the linked list
3 : Display the elements in the linked list
4 : Count the elements in the linked list
5 : Exit()
Enter your option :
1
Enter elements for inserting into linked list :
010

```

```
1 : Insert elements into the linked list
2 : Delete elements from the linked list
3 : Display the elements in the linked list
4 : Count the elements in the linked list
5 : Exit()
Enter your option :
1
Enter elements for inserting into linked list :
100
Options
1 : Insert elements into the linked list
2 : Delete elements from the linked list
3 : Display the elements in the linked list
4 : Count the elements in the linked list
5 : Exit()
Enter your option :
1
Enter elements for inserting into linked list :
101
Options
1 : Insert elements into the linked list
2 : Delete elements from the linked list
3 : Display the elements in the linked list
4 : Count the elements in the linked list
5 : Exit()
Enter your option :
3
The elements in the linked list are : 1 10 100 101
Options
1 : Insert elements into the linked list
2 : Delete elements from the linked list
3 : Display the elements in the linked list
4 : Count the elements in the linked list
5 : Exit()
Enter your option :
2
Enter position of the element for deleteing the element :
3
Deleted successfully
Options
1 : Insert elements into the linked list
2 : Delete elements from the linked list
3 : Display the elements in the linked list
4 : Count the elements in the linked list
5 : Exit()
Enter your option :
3
The elements in the linked list are : 1 10 101
Options
1 : Insert elements into the linked list
```

```
3 : Display the elements in the linked list
4 : Count the elements in the linked list
5 : Exit()
Enter your option :
4
No of elements in the linked list are : 3
Options
1 : Insert elements into the linked list
2 : Delete elements from the linked list
3 : Display the elements in the linked list
4 : Count the elements in the linked list
5 : Exit()
Enter your option :
5
```

ID: 224G1A0535 Page No: 43

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

S.No: 14

Exp. Name: **C program which performs all operations on double linked list.**

Date: 2023-06-15

Aim:

Write a C program that uses functions to perform the following **operations on double linked list**

- i) Creation
- ii) Insertion
- iii) Deletion
- iv) Traversal

Source Code:

AllOperationsDLL.c

ID: 224G1A0535 | Page No: 44

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

```

#include<stdio.h>
#include<stdlib.h>
#include<conio.h>
struct dnode
{
    int data;
    struct dnode*prev;
    struct dnode*next;
};
struct dnode *start=NULL;
void insert(int);
void remov(int);
void display();

int main()
{
    int n,ch;
    do
    {
        printf("Operations on doubly linked list\n");
        printf("1. Insert \n2.Remove\n3. Display\n0. Exit");
        printf("\nEnter Choice 0-4? : ");
        scanf("%d",&ch);
        switch(ch)
        {
            case 0:
                exit(0);
            case 1:
                printf("Enter number: ");
                scanf("%d",&n);
                insert(n);
                break;
            case 2:
                printf("Enter number to delete: ");
                scanf("%d",&n);
                remov(n);
                break;
            case 3:
                display();
                break;
        }
    }while(ch!=0);
}
void insert(int num)
{
    struct dnode*nptr,*temp=start;
    nptr=malloc(sizeof(struct dnode));
    nptr->data=num;
    nptr->next=NULL;
    nptr->prev=NULL;
    if(start==NULL)
    {
        start=nptr;
    }
    else

```

```

        temp=temp->next;
        nptr->prev=temp;
        temp->next=nptr;
    }
}
void remov(int num)
{
    struct dnode*temp=start;
    while(temp!=NULL)
    {
        if(temp->data==num)
        {
            if(temp==start)
            {
                start=start->next;
                start->prev=NULL;
            }
            else
            {
                if(temp->next==NULL)
                temp->prev->next=NULL;
                else
                {
                    temp->prev->next=temp->next;
                    temp->next->prev=temp->prev;
                }
                free(temp);
            }
            return;
        }
        temp=temp->next;
    }
    printf("%d not found.\n",num);
}
void display()
{
    struct dnode *temp=start;
    while(temp!=NULL)
    {
        printf("%d\t",temp->data);
        temp=temp->next;
    }
    printf("\n");
}

```

Execution Results - All test cases have succeeded!

Test Case - 1	
User Output	
Operations on doubly linked list	
1.Insert	
2.Remove	

```
0.Exit
Enter Choice 0-4?:
1
Enter number:
15
Operations on doubly linked list
1.Insert
2.Remove
3.Display
0.Exit
Enter Choice 0-4?:
1
Enter number:
16
Operations on doubly linked list
1.Insert
2.Remove
3.Display
0.Exit
Enter Choice 0-4?:
1
Enter number:
17
Operations on doubly linked list
1.Insert
2.Remove
3.Display
0.Exit
Enter Choice 0-4?:
1
Enter number:
18
Operations on doubly linked list
1.Insert
2.Remove
3.Display
0.Exit
Enter Choice 0-4?:
3
15 16 17 18
Operations on doubly linked list
1.Insert
2.Remove
3.Display
0.Exit
Enter Choice 0-4?:
2
Enter number to delete:
19
19 not found
```

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

ID: 224G1A0535 Page No: 47

1.Insert
2.Remove
3.Display
0.Exit
Enter Choice 0-4?:
3
15 16 17 18
Operations on doubly linked list
1.Insert
2.Remove
3.Display
0.Exit
Enter Choice 0-4?:
2
Enter number to delete:
16
Operations on doubly linked list
1.Insert
2.Remove
3.Display
0.Exit
Enter Choice 0-4?:
0

ID: 224G1A0535 Page No: 48

2022-2026-CSE-A

S.No: 15

Exp. Name: **C program to which performs all operations on Circular linked list.**

Date: 2023-05-16

Aim:

Write a program that uses functions to perform the following **operations on Circular linked list**
i)Creation ii)insertion iii)deletion iv) Traversal

Source Code:

AlloperationsinCLL.c

ID: 224G1A0535 | Page No: 49

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

```

#include<stdio.h>
#include<stdlib.h>
struct node{
    int data;
    struct node *next;
};
void insert();
void deletion();
void find();
void print();
struct node *head = NULL;
int main()
{
    int choice;
    printf("CIRCULAR LINKED LIST IMPLEMENTATION OF LIST ADT\n");
    while(1)
    {
        printf("1.INSERT ");
        printf("2.DELETE ");
        printf("3.FIND ");
        printf("4.PRINT ");
        printf("5.QUIT\n");
        printf("Enter the choice: ");
        scanf("%d", &choice);
        switch(choice)
        {
            case 1:insert();break;
            case 2:deletion();break;
            case 3:find();break;
            case 4:print();break;
            case 5:exit(0);
        }
    }
}
void insert()
{
    int x,n;
    struct node *newnode,*temp = head, *prev;
    newnode = (struct node*)malloc(sizeof(struct node));
    printf("Enter the element to be inserted: ");
    scanf("%d", &x);
    printf("Enter the position of the element: ");
    scanf("%d", &n);
    newnode->data = x;
    newnode->next = NULL;
    if(head == NULL)
    {
        head = newnode;
        newnode->next = newnode;
    }
    else if(n == 1)
    { temp = head;
    newnode->next = temp;
    while(temp->next != head)
        temp = temp->next;
}
}

```

```

    }
else
{   for(int i =1; i < n-1; i++)
    {   temp = temp->next;   }
    newnode->next = temp->next;
    temp->next = newnode;
}
}

void deletion()
{
    struct node *temp = head, *prev, *temp1 = head;
    int key, count = 0;
    printf("Enter the element to be deleted: ");
    scanf("%d", &key);
    if(temp->data == key){
        prev = temp -> next;
        while(temp->next != head)
        {   temp = temp->next;   }
        temp->next = prev;
        free(head);
        head = prev;
        printf("Element deleted\n");
    }
    else{
        while(temp->next != head)
        {   if(temp->data == key){
            count += 1;
            break;
        }
        prev = temp;
        temp = temp->next;
    }
    if(temp->data == key)
    {   prev->next = temp->next;
        free(temp);
        printf("Element deleted\n");
    }
    else{
        printf("Element does not exist...!\n");
    }
}
}

void find()
{
    struct node*temp = head;
    int key, count = 0;
    printf("Enter the element to be searched: ");
    scanf("%d",&key);
    while(temp->next != head)
    {
        if(temp->data == key)
        {   count = 1;
            break;   }
        temp = temp->next;
    }
}

```

```

        printf("Element exist...!\n");
    else
    {   if(temp->data == key)
        printf("Element exist...!\n");
    else
        printf("Element does not exist...!\n");
    }
}
void print(){
    struct node*temp = head;
    printf("The list element are: ");
    while(temp->next != head)
    {
        printf("%d -> ",temp->data);
        temp = temp->next;
    }
    printf("%d -> ", temp->data) ;
    printf("\n");
}

```

ID: 224G1A0535 | Page No: 52

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
CIRCULAR LINKED LIST IMPLEMENTATION OF LIST ADT
1.INSERT 2.DELETE 3.FIND 4.PRINT 5.QUIT
Enter the choice:
1
Enter the element to be inserted:
12
Enter the position of the element:
1
1.INSERT 2.DELETE 3.FIND 4.PRINT 5.QUIT
Enter the choice:
1
Enter the element to be inserted:
14
Enter the position of the element:
2
1.INSERT 2.DELETE 3.FIND 4.PRINT 5.QUIT
Enter the choice:
1
Enter the element to be inserted:
15
Enter the position of the element:
3
1.INSERT 2.DELETE 3.FIND 4.PRINT 5.QUIT
Enter the choice:
4

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

```
The list element are: 12 -> 14 -> 15 ->
```

```
1.INSERT 2.DELETE 3.FIND 4.PRINT 5.QUIT
```

```
Enter the choice:
```

```
2
```

```
Enter the element to be deleted:
```

```
14
```

```
Element deleted
```

```
1.INSERT 2.DELETE 3.FIND 4.PRINT 5.QUIT
```

```
Enter the choice:
```

```
4
```

```
The list element are: 12 -> 15 ->
```

```
1.INSERT 2.DELETE 3.FIND 4.PRINT 5.QUIT
```

```
Enter the choice:
```

```
3
```

```
Enter the element to be searched:
```

```
12
```

```
Element exist...!
```

```
1.INSERT 2.DELETE 3.FIND 4.PRINT 5.QUIT
```

```
Enter the choice:
```

```
5
```

Test Case - 2

User Output

```
CIRCULAR LINKED LIST IMPLEMENTATION OF LIST ADT
```

```
1.INSERT 2.DELETE 3.FIND 4.PRINT 5.QUIT
```

```
Enter the choice:
```

```
1
```

```
Enter the element to be inserted:
```

```
54
```

```
Enter the position of the element:
```

```
1
```

```
1.INSERT 2.DELETE 3.FIND 4.PRINT 5.QUIT
```

```
Enter the choice:
```

```
2
```

```
Enter the element to be deleted:
```

```
1
```

```
Element does not exist...!
```

```
1.INSERT 2.DELETE 3.FIND 4.PRINT 5.QUIT
```

```
Enter the choice:
```

```
4
```

```
The list element are: 54 ->
```

```
1.INSERT 2.DELETE 3.FIND 4.PRINT 5.QUIT
```

```
Enter the choice:
```

```
1
```

```
Enter the element to be inserted:
```

```
65
```

```
Enter the position of the element:
```

```
2
```

```
Enter the choice:  
4  
The list element are: 54 -> 65 ->  
1.INSERT 2.DELETE 3.FIND 4.PRINT 5.QUIT  
Enter the choice:  
5
```

S.No: 16

Exp. Name: ***Implementation of Circular Queue
using Dynamic Array***

Date: 2023-06-15

Aim:

Write a program to implement [circular queue](#) using **dynamic array**.

ID: 224G1A0535 | Page No: 55

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

Sample Input and Output:

```
Enter the maximum size of the circular queue : 3
1.Enqueue 2.Dequeue 3.Display 4.Exit
Enter your option : 2
Circular queue is underflow.
1.Enqueue 2.Dequeue 3.Display 4.Exit
Enter your option : 3
Circular queue is empty.
1.Enqueue 2.Dequeue 3.Display 4.Exit
Enter your option : 1
Enter element : 111
Successfully inserted.
1.Enqueue 2.Dequeue 3.Display 4.Exit
Enter your option : 1
Enter element : 222
Successfully inserted.
1.Enqueue 2.Dequeue 3.Display 4.Exit
Enter your option : 1
Enter element : 333
Successfully inserted.
1.Enqueue 2.Dequeue 3.Display 4.Exit
Enter your option : 1
Enter element : 444
Circular queue is overflow.
1.Enqueue 2.Dequeue 3.Display 4.Exit
Enter your option : 3
Elements in the circular queue : 111 222 333
1.Enqueue 2.Dequeue 3.Display 4.Exit
Enter your option : 2
Deleted element = 111
1.Enqueue 2.Dequeue 3.Display 4.Exit
Enter your option : 1
Enter element : 444
Successfully inserted.
1.Enqueue 2.Dequeue 3.Display 4.Exit
Enter your option : 3
Elements in the circular queue : 222 333 444
1.Enqueue 2.Dequeue 3.Display 4.Exit
Enter your option : 2
Deleted element = 222
1.Enqueue 2.Dequeue 3.Display 4.Exit
Enter your option : 2
Deleted element = 333
1.Enqueue 2.Dequeue 3.Display 4.Exit
Enter your option : 2
Deleted element = 444
1.Enqueue 2.Dequeue 3.Display 4.Exit
Enter your option : 3
Circular queue is empty.
1.Enqueue 2.Dequeue 3.Display 4.Exit
Enter your option : 4
```

ID: 224G1A0535 | Page No: 56

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

Source Code:

```
CQueueUsingDynamicArray.c
```

```

#include<stdio.h>
#include<stdlib.h>
int *cqueue;
int front, rear;
int maxSize;
void initCircularQueue()
{
    cqueue = (int *)malloc(maxSize * sizeof(int));
    front = -1;
    rear = -1;
}
void dequeue()
{
    if(front == -1)
    {
        printf("Circular queue is underflow.\n");
    }
    else
    {
        printf("Deleted element = %d\n", *(cqueue + front));
        if(rear == front)
        {
            rear = front = -1;
        }
        else if(front == maxSize - 1)
        {
            front = 0;
        }
        else
        {
            front++;
        }
    }
}
void enqueue(int x)
{
    if (((rear == maxSize - 1)&& (front == 0)) || (rear + 1 == front))
    {
        printf("Circular queue is overflow.\n");
    }
    else
    {
        if(rear == maxSize - 1)
        {
            rear = -1;
        }
        else if (front == -1)
        {
            front = 0;
        }
        rear++;
        cqueue[rear] = x;
        printf("Successfully inserted.\n");
    }
}

```

```

int i;
if(front == -1 && rear == -1)
{
    printf("Circular queue is empty.\n");
}
else
{
    printf("Elements in the circular queue : ");
    if(front <= rear)
    {
        for (i = front; i <= rear; i++)
        {
            printf("%d ", *(cqueue + i));
        }
    }
    else
    {
        for (i = front; i <= maxSize - 1; i++)
        {
            printf("%d ", *(cqueue + i));
        }
    }
    for(i = 0; i <= rear; i++)
    {
        printf("%d ", *(cqueue +i));
    }
}
printf("\n");
}

int main()
{
    int op, x;
    printf("Enter the maximum size of the circular queue : ");
    scanf("%d", &maxSize);
    initCircularQueue();
    while(1)
    {
        printf("1.Enqueue 2.Dequeue 3.Display 4.Exit\n");
        printf("Enter your option : ");
        scanf("%d",&op);
        switch(op)
        {
        case 1:
            printf("Enter element : ");
            scanf("%d",&x);
            enqueue(x);
            break;
        case 2:
            dequeue();
            break;
        case 3:
            display();
            break;
        case 4:
            exit(0);
        }
    }
}

```

}

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Enter the maximum size of the circular queue :
3
1.Enqueue 2.Dequeue 3.Display 4.Exit
Enter your option :
2
Circular queue is underflow.
1.Enqueue 2.Dequeue 3.Display 4.Exit
Enter your option :
3
Circular queue is empty.
1.Enqueue 2.Dequeue 3.Display 4.Exit
Enter your option :
1
Enter element :
111
Successfully inserted.
1.Enqueue 2.Dequeue 3.Display 4.Exit
Enter your option :
1
Enter element :
222
Successfully inserted.
1.Enqueue 2.Dequeue 3.Display 4.Exit
Enter your option :
1
Enter element :
333
Successfully inserted.
1.Enqueue 2.Dequeue 3.Display 4.Exit
Enter your option :
1
Enter element :
444
Circular queue is overflow.
1.Enqueue 2.Dequeue 3.Display 4.Exit
Enter your option :
3
Elements in the circular queue : 111 222 333
1.Enqueue 2.Dequeue 3.Display 4.Exit
Enter your option :

```
Deleted element = 111
1.Enqueue 2.Dequeue 3.Display 4.Exit
Enter your option :
1
Enter element :
444
Successfully inserted.
1.Enqueue 2.Dequeue 3.Display 4.Exit
Enter your option :
3
Elements in the circular queue : 222 333 444
1.Enqueue 2.Dequeue 3.Display 4.Exit
Enter your option :
2
Deleted element = 222
1.Enqueue 2.Dequeue 3.Display 4.Exit
Enter your option :
2
Deleted element = 333
1.Enqueue 2.Dequeue 3.Display 4.Exit
Enter your option :
3
Deleted element = 444
1.Enqueue 2.Dequeue 3.Display 4.Exit
Enter your option :
4
Circular queue is empty.
```

ID: 224G1A0535 Page No: 60

2022-2026-CSE-A

S.No: 17	Exp. Name: <i>Write a C program to implement different Operations on Stack using Array representation</i>	Date: 2023-06-16
----------	--	------------------

Aim:

Write a program to implement [stack](#) using [arrays](#).

Sample Input and Output:

```

1.Push 2.Pop 3.Display 4.IsEmpty 5.Peek 6.Exit
Enter your option : 4
Stack is empty.

1.Push 2.Pop 3.Display 4.IsEmpty 5.Peek 6.Exit
Enter your option : 2
Stack is underflow.

1.Push 2.Pop 3.Display 4.IsEmpty 5.Peek 6.Exit
Enter your option : 3
Stack is empty.

1.Push 2.Pop 3.Display 4.IsEmpty 5.Peek 6.Exit
Enter your option : 5
Stack is underflow.

1.Push 2.Pop 3.Display 4.IsEmpty 5.Peek 6.Exit
Enter your option : 1
Enter element : 25
Successfully pushed.

1.Push 2.Pop 3.Display 4.IsEmpty 5.Peek 6.Exit
Enter your option : 1
Enter element : 26
Successfully pushed.

1.Push 2.Pop 3.Display 4.IsEmpty 5.Peek 6.Exit
Enter your option : 3
Elements of the stack are : 26 25

1.Push 2.Pop 3.Display 4.IsEmpty 5.Peek 6.Exit
Enter your option : 2
Popped value = 26

1.Push 2.Pop 3.Display 4.IsEmpty 5.Peek 6.Exit
Enter your option : 4
Stack is not empty.

1.Push 2.Pop 3.Display 4.IsEmpty 5.Peek 6.Exit
Enter your option : 5
Peek value = 25

1.Push 2.Pop 3.Display 4.IsEmpty 5.Peek 6.Exit
Enter your option : 6

```

ID: 224G1A0535 | Page No: 61

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

```

#include<stdio.h>
#include<stdlib.h>
#define STACK_MAX_SIZE 10
int arr[STACK_MAX_SIZE];
int top = -1;
void push(int element)
{
    if(top == STACK_MAX_SIZE - 1)
    {
        printf("Stack is overflow.\n");
    }
    else
    {
        top = top + 1;
        arr[top] = element;
        printf("Successfully pushed.\n");
    }
}
void display()
{
    if (top < 0)
    {
        printf("Stack is empty.\n");
    }
    else
    {
        printf("Elements of the stack are : " );
        for(int i = top; i >= 0; i--)
        {
            printf("%d ", arr[i]);
        }
        printf("\n");
    }
}
void pop()
{
    int x;
    if(top < 0)
    {
        printf("Stack is underflow.\n");
    }
    else
    {
        x = arr[top];
        top = top - 1;
        printf("Popped value = %d\n",x);
    }
}
void peek()
{
    int x;
    if(top < 0)
    {
        printf("Stack is underflow.\n");
    }
}

```

```

        x = arr[top];
        printf("Peek value = %d\n",x);
    }
}
void isEmpty()
{
    if (top < 0)
    {
        printf("Stack is empty.\n");
    }
    else
    {
        printf("Stack is not empty.\n");
    }
}
int main()
{
    int op, x;
    while(1)
    {
        printf("1.Push 2.Pop 3.Display 4.Is Empty 5.Peek 6.Exit\n");
        printf("Enter your option : ");
        scanf("%d", &op);
        switch(op)
        {
            case 1:
                printf("Enter element : ");
                scanf("%d",&x);
                push(x);
                break;
            case 2:
                pop();
                break;
            case 3:
                display();
                break;
            case 4:
                isEmpty();
                break;
            case 5:
                peek();
                break;
            case 6:
                exit(0);
        }
    }
}

```

ID: 224G1A0535 | Page No: 63

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
1.Push 2.Pop 3.Display 4.Is Empty 5.Peek 6.Exit

```
Enter your option :  
1  
Enter element :  
10  
Successfully pushed.  
1.Push 2.Pop 3.Display 4.IsEmpty 5.Peek 6.Exit  
Enter your option :  
1  
Enter element :  
20  
Successfully pushed.  
1.Push 2.Pop 3.Display 4.IsEmpty 5.Peek 6.Exit  
Enter your option :  
1  
Enter element :  
30  
Successfully pushed.  
1.Push 2.Pop 3.Display 4.IsEmpty 5.Peek 6.Exit  
Enter your option :  
3  
Elements of the stack are : 30 20 10  
1.Push 2.Pop 3.Display 4.IsEmpty 5.Peek 6.Exit  
Enter your option :  
5  
Peek value = 30  
1.Push 2.Pop 3.Display 4.IsEmpty 5.Peek 6.Exit  
Enter your option :  
2  
Popped value = 30  
1.Push 2.Pop 3.Display 4.IsEmpty 5.Peek 6.Exit  
Enter your option :  
2  
Popped value = 20  
1.Push 2.Pop 3.Display 4.IsEmpty 5.Peek 6.Exit  
Enter your option :  
3  
Elements of the stack are : 10  
1.Push 2.Pop 3.Display 4.IsEmpty 5.Peek 6.Exit  
Enter your option :  
5  
Peek value = 10  
1.Push 2.Pop 3.Display 4.IsEmpty 5.Peek 6.Exit  
Enter your option :  
4  
Stack is not empty.  
1.Push 2.Pop 3.Display 4.IsEmpty 5.Peek 6.Exit  
Enter your option :  
2  
Popped value = 10  
1.Push 2.Pop 3.Display 4.IsEmpty 5.Peek 6.Exit
```

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

ID: 224G1A0535 Page No: 64

3

Stack is empty.

1.Push 2.Pop 3.Display 4.IsEmpty 5.Peek 6.Exit

Enter your option :

4

Stack is empty.

1.Push 2.Pop 3.Display 4.IsEmpty 5.Peek 6.Exit

Enter your option :

6

S.No: 18

Exp. Name: ***Write a C program to implement different Operations on Stack using Linked Lists***

Date: 2023-06-18

Aim:

Write a program to implement stack using **linked lists**.

```
Sample Input and Output:  
1.Push 2.Pop 3.Display 4.IsEmpty 5.Peek 6.Exit  
Enter your option : 1  
Enter element : 33  
Successfully pushed.  
1.Push 2.Pop 3.Display 4.IsEmpty 5.Peek 6.Exit  
Enter your option : 1  
Enter element : 22  
Successfully pushed.  
1.Push 2.Pop 3.Display 4.IsEmpty 5.Peek 6.Exit  
Enter your option : 1  
Enter element : 55  
Successfully pushed.  
1.Push 2.Pop 3.Display 4.IsEmpty 5.Peek 6.Exit  
Enter your option : 1  
Enter element : 66  
Successfully pushed.  
1.Push 2.Pop 3.Display 4.IsEmpty 5.Peek 6.Exit  
Enter your option : 3  
Elements of the stack are : 66 55 22 33  
1.Push 2.Pop 3.Display 4.IsEmpty 5.Peek 6.Exit  
Enter your option : 2  
Popped value = 66  
1.Push 2.Pop 3.Display 4.IsEmpty 5.Peek 6.Exit  
Enter your option : 2  
Popped value = 55  
1.Push 2.Pop 3.Display 4.IsEmpty 5.Peek 6.Exit  
Enter your option : 3  
Elements of the stack are : 22 33  
1.Push 2.Pop 3.Display 4.IsEmpty 5.Peek 6.Exit  
Enter your option : 5  
Peek value = 22  
1.Push 2.Pop 3.Display 4.IsEmpty 5.Peek 6.Exit  
Enter your option : 4  
Stack is not empty.  
1.Push 2.Pop 3.Display 4.IsEmpty 5.Peek 6.Exit  
Enter your option : 6
```

Source Code:

```
StackUsingLLList.c
```

ID: 224G1A0535 Page No: 66

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

```

#include <stdio.h>
#include <stdlib.h>
struct stack
{
    int data;
    struct stack *next;
};
typedef struct stack *stk;
stk top = NULL;
stk push(int x)
{
    stk temp;
    temp = (stk)malloc(sizeof(struct stack));
    if(temp == NULL)
    {
        printf("Stack is overflow.\n");
    }
    else
    {
        temp->data = x;
        temp->next = top;
        top = temp;
        printf("Successfully pushed.\n");
    }
}
void display()
{
    stk temp = top;
    if(temp == NULL)
    {
        printf("Stack is empty.\n");
    }
    else
    {
        printf("Elements of the stack are : ");
        while(temp != NULL)
        {
            printf("%d ",temp->data);
            temp=temp->next;
        }
        printf("\n");
    }
}
stk pop()
{
    stk temp;
    if(top == NULL)
    {
        printf("Stack is underflow.\n");
    }
    else
    {
        temp=top;
        top=top->next;
        printf("Popped value = %d\n",temp->data);
        free(temp);
    }
}

```

```

void peek()
{
    stk temp;
    if(top == NULL)
    {
        printf("Stack is underflow.\n");
    }
    else
    {
        temp=top;
        printf("Peek value = %d\n",temp->data);
    }
}
void isEmpty()
{
    if(top==NULL)
    {
        printf("Stack is empty.\n");
    }
    else
    {
        printf("Stack is not empty.\n");
    }
}
int main()
{
int op,x;
while(1)
{
    printf("1.Push 2.Pop 3.Display 4.IsEmpty 5.Peek 6.Exit\n");
    printf("Enter your option : ");
    scanf("%d", &op);
    switch(op)
    {
        case 1:
        printf("Enter element : ");
        scanf("%d", &x);
        push(x);
        break;
        case 2:
        pop();
        break;
        case 3:
        display();
        break;
        case 4:
        isEmpty();
        break;
        case 5:
        peek();
        break;
        case 6:
        exit(0);
    }
}
}

```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
1.Push 2.Pop 3.Display 4.IsEmpty 5.Peek 6.Exit
Enter your option :
1
Enter element :
33
Successfully pushed.
1.Push 2.Pop 3.Display 4.IsEmpty 5.Peek 6.Exit
Enter your option :
1
Enter element :
22
Successfully pushed.
1.Push 2.Pop 3.Display 4.IsEmpty 5.Peek 6.Exit
Enter your option :
1
Enter element :
55
Successfully pushed.
1.Push 2.Pop 3.Display 4.IsEmpty 5.Peek 6.Exit
Enter your option :
1
Enter element :
66
Successfully pushed.
1.Push 2.Pop 3.Display 4.IsEmpty 5.Peek 6.Exit
Enter your option :
3
Elements of the stack are : 66 55 22 33
1.Push 2.Pop 3.Display 4.IsEmpty 5.Peek 6.Exit
Enter your option :
2
Popped value = 66
1.Push 2.Pop 3.Display 4.IsEmpty 5.Peek 6.Exit
Enter your option :
2
Popped value = 55
1.Push 2.Pop 3.Display 4.IsEmpty 5.Peek 6.Exit
Enter your option :
3
Elements of the stack are : 22 33
1.Push 2.Pop 3.Display 4.IsEmpty 5.Peek 6.Exit
Enter your option :
5
Peek value = 22
1.Push 2.Pop 3.Display 4.IsEmpty 5.Peek 6.Exit

ID: 224G1A0535 | Page No: 69

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

4

Stack is not empty.

1.Push 2.Pop 3.Display 4.IsEmpty 5.Peek 6.Exit

Enter your option :

6

Test Case - 2

User Output

1.Push 2.Pop 3.Display 4.IsEmpty 5.Peek 6.Exit

Enter your option :

2

Stack is underflow.

1.Push 2.Pop 3.Display 4.IsEmpty 5.Peek 6.Exit

Enter your option :

3

Stack is empty.

1.Push 2.Pop 3.Display 4.IsEmpty 5.Peek 6.Exit

Enter your option :

5

Stack is underflow.

1.Push 2.Pop 3.Display 4.IsEmpty 5.Peek 6.Exit

Enter your option :

4

Stack is empty.

1.Push 2.Pop 3.Display 4.IsEmpty 5.Peek 6.Exit

Enter your option :

1

Enter element :

23

Successfully pushed.

1.Push 2.Pop 3.Display 4.IsEmpty 5.Peek 6.Exit

Enter your option :

1

Enter element :

24

Successfully pushed.

1.Push 2.Pop 3.Display 4.IsEmpty 5.Peek 6.Exit

Enter your option :

3

Elements of the stack are : 24 23

1.Push 2.Pop 3.Display 4.IsEmpty 5.Peek 6.Exit

Enter your option :

5

Peek value = 24

1.Push 2.Pop 3.Display 4.IsEmpty 5.Peek 6.Exit

Enter your option :

2

Popped value = 24

Enter your option :
2
Popped value = 23
1.Push 2.Pop 3.Display 4.IsEmpty 5.Peek 6.Exit
Enter your option :
2
Stack is underflow.
1.Push 2.Pop 3.Display 4.IsEmpty 5.Peek 6.Exit
Enter your option :
4
Stack is empty.
1.Push 2.Pop 3.Display 4.IsEmpty 5.Peek 6.Exit
Enter your option :
6

ID: 224G1A0535 Page No: 71

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

S.No: 19	Exp. Name: <i>Write a C program to implement different Operations on Queue using Array representation</i>	Date: 2023-06-16
----------	--	------------------

Aim:

Write a program to implement queue using **arrays**.

```
Sample Input and Output:
1.Enqueue 2.Dequeue 3.Display 4.Is Empty 5.Size 6.Exit
Enter your option : 1
Enter element : 23
Successfully inserted.
1.Enqueue 2.Dequeue 3.Display 4.Is Empty 5.Size 6.Exit
Enter your option : 1
Enter element : 56
Successfully inserted.
1.Enqueue 2.Dequeue 3.Display 4.Is Empty 5.Size 6.Exit
Enter your option : 3
Elements in the queue : 23 56
1.Enqueue 2.Dequeue 3.Display 4.Is Empty 5.Size 6.Exit
Enter your option : 4
Queue is not empty.
1.Enqueue 2.Dequeue 3.Display 4.Is Empty 5.Size 6.Exit
Enter your option : 5
Queue size : 2
1.Enqueue 2.Dequeue 3.Display 4.Is Empty 5.Size 6.Exit
Enter your option : 2
Deleted element = 23
1.Enqueue 2.Dequeue 3.Display 4.Is Empty 5.Size 6.Exit
Enter your option : 2
Deleted element = 56
1.Enqueue 2.Dequeue 3.Display 4.Is Empty 5.Size 6.Exit
Enter your option : 4
Queue is empty.
1.Enqueue 2.Dequeue 3.Display 4.Is Empty 5.Size 6.Exit
Enter your option : 6
```

Source Code:

```
QUsingArray.c
```

```

#include<conio.h>
#include<stdio.h>
#define MAX 10
int queue[MAX];
int front = -1, rear = -1;
void enqueue(int x)
{
    if (rear == MAX - 1)
    {
        printf("Queue is overflow.\n");
    }
    else
    {
        rear++;
        queue[rear] = x;
        printf("Successfully inserted.\n");
    }
    if (front == -1)
    {
        front++;
    }
}
void dequeue()
{
    if (front == -1)
    {
        printf("Queue is underflow.\n");
    }
    else
    {
        printf("Deleted element = %d\n",queue[front]);
        if (rear == front)
        {
            rear = front = -1;
        }
        else
        {
            front++;
        }
    }
}
void display()
{
    if (front == -1 && rear == -1)
    {
        printf("Queue is empty.\n");
    }
    else
    {
        printf("Elements in the queue : ");
        for (int i = front; i <= rear; i++)
        {
            printf("%d ",queue[i]);
        }
        printf("\n");
    }
}

```

```

void size()
{
    if(front == -1 && rear == -1)
        printf("Queue size : 0\n");
    else
        printf("Queue size : %d\n",rear-front+1);
}
void isEmpty()
{
    if(front == -1 && rear == -1)
        printf("Queue is empty.\n");
    else
        printf("Queue is not empty.\n");
}
int main()
{
    int op, x;
    while(1)
    {
        printf("1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit\n");
        printf("Enter your option : ");
        scanf("%d",&op);
        switch(op)
        {
            case 1:
                printf("Enter element : ");
                scanf("%d",&x);
                enqueue(x);
                break;
            case 2:
                dequeue();
                break;
            case 3:
                display();
                break;
            case 4:
                isEmpty();
                break;
            case 5:
                size();
                break;
            case 6:
                exit(0);
        }
    }
}

```

ID: 224G1A0535 | Page No: 74

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit
Enter your option :

```
2
Queue is underflow.
1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit
Enter your option :
3
Queue is empty.
1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit
Enter your option :
4
Queue is empty.
1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit
Enter your option :
5
Queue size : 0
1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit
Enter your option :
1
Enter element :
14
Successfully inserted.
1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit
Enter your option :
1
Enter element :
78
Successfully inserted.
1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit
Enter your option :
1
Enter element :
53
Successfully inserted.
1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit
Enter your option :
3
Elements in the queue : 14 78 53
1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit
Enter your option :
5
Queue size : 3
1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit
Enter your option :
6
```

Test Case - 2
User Output
1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit Enter your option :

```

Enter element :
25
Successfully inserted.
1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit
Enter your option :
2
Deleted element = 25
1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit
Enter your option :
2
Queue is underflow.
1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit
Enter your option :
3
Queue is empty.
1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit
Enter your option :
1
Enter element :
65
Successfully inserted.
1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit
Enter your option :
3
Elements in the queue : 65
1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit
Enter your option :
4
Queue is not empty.
1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit
Enter your option :
2
Deleted element = 65
1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit
Enter your option :
4
Queue is empty.
1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit
Enter your option :
5
Queue size : 0
1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit
Enter your option :
1
Enter element :
63
Successfully inserted.
1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit
Enter your option :
5

```

1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit

Enter your option :

6

S.No: 20	Exp. Name: <i>Write a C program to implement different Operations on Queue using Dynamic Array</i>	Date: 2023-06-18
----------	---	------------------

Aim:

Write a program to implement queue using **dynamic array**.

In this queue implementation has

1. a pointer 'queue' to a dynamically allocated array (used to hold the contents of the queue)
2. an integer 'maxSize' that holds the size of this array (i.e the maximum number of data that can be held in this array)
3. an integer 'front' which stores the array index of the first element in the queue
4. an integer 'rear' which stores the array index of the last element in the queue.

Sample Input and Output:

```

Enter the maximum size of the queue : 3
1.Enqueue 2.Dequeue 3.Display 4.Exit
Enter your option : 2
Queue is underflow.
1.Enqueue 2.Dequeue 3.Display 4.Exit
Enter your option : 3
Queue is empty.
1.Enqueue 2.Dequeue 3.Display 4.Exit
Enter your option : 1
Enter element : 15
Successfully inserted.
1.Enqueue 2.Dequeue 3.Display 4.Exit
Enter your option : 1
Enter element : 16
Successfully inserted.
1.Enqueue 2.Dequeue 3.Display 4.Exit
Enter your option : 1
Enter element : 17
Successfully inserted.
1.Enqueue 2.Dequeue 3.Display 4.Exit
Enter your option : 1
Enter element : 18
Queue is overflow.
1.Enqueue 2.Dequeue 3.Display 4.Exit
Enter your option : 3
Elements in the queue : 15 16 17
1.Enqueue 2.Dequeue 3.Display 4.Exit
Enter your option : 2
Deleted element = 15
1.Enqueue 2.Dequeue 3.Display 4.Exit
Enter your option : 2
Deleted element = 16
1.Enqueue 2.Dequeue 3.Display 4.Exit
Enter your option : 3
Elements in the queue : 17
1.Enqueue 2.Dequeue 3.Display 4.Exit
Enter your option : 2
Deleted element = 17
1.Enqueue 2.Dequeue 3.Display 4.Exit
Enter your option : 3
Queue is empty.
1.Enqueue 2.Dequeue 3.Display 4.Exit
Enter your option : 2
Queue is underflow.
1.Enqueue 2.Dequeue 3.Display 4.Exit
Enter your option : 4

```

Source Code:

8

ID: 224G1A0535

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

Page No: 78


```

#include <conio.h>
#include <stdio.h>
int *queue;
int front, rear;
int maxSize;
void initQueue()
{
    queue = (int *)malloc(maxSize*sizeof(int));
    front = -1;
    rear = -1;
}
void enqueue(int x)
{
    if (rear == maxSize - 1)
    {
        printf("Queue is overflow.\n");
    }
    else
    {
        rear++;
        queue[rear] = x;
        printf("Successfully inserted.\n");
    }
    if (front == -1)
    {
        front++;
    }
}
void dequeue()
{
    if (front == -1)
    {
        printf("Queue is underflow.\n");
    }
    else
    {
        printf("Deleted element = %d\n", *(queue+front));
        if (rear == front)
        {
            rear = front = -1;
        }
        else
        {
            front++;
        }
    }
}
void display()
{
    if (front == -1 && rear == -1)
    {
        printf("Queue is empty.\n");
    }
    else
    {

```

```

    {
        printf("%d ", *(queue+i));
    }
    printf("\n");
}
int main()
{
    int op, x;
    printf("Enter the maximum size of the queue : ");
    scanf("%d", &maxSize);
    initQueue();
    while(1)
    {
        printf("1.Enqueue 2.Dequeue 3.Display 4.Exit\n");
        printf("Enter your option : ");
        scanf("%d",&op);
        switch(op)
        {
            case 1:
                printf("Enter element : ");
                scanf("%d",&x);
                enqueue(x);
                break;
            case 2:
                dequeue();
                break;
            case 3:
                display();
                break;
            case 4:
                exit(0);
        }
    }
}

```

ID: 224G1A0535 | Page No: 81

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Enter the maximum size of the queue :
3
1.Enqueue 2.Dequeue 3.Display 4.Exit
Enter your option :
2
Queue is underflow.
1.Enqueue 2.Dequeue 3.Display 4.Exit
Enter your option :
3
Queue is empty.
1.Enqueue 2.Dequeue 3.Display 4.Exit

1
Enter element :
15
Successfully inserted.
1.Enqueue 2.Dequeue 3.Display 4.Exit
Enter your option :
1
Enter element :
16
Successfully inserted.
1.Enqueue 2.Dequeue 3.Display 4.Exit
Enter your option :
1
Enter element :
17
Successfully inserted.
1.Enqueue 2.Dequeue 3.Display 4.Exit
Enter your option :
1
Enter element :
18
Queue is overflow.
1.Enqueue 2.Dequeue 3.Display 4.Exit
Enter your option :
3
Elements in the queue : 15 16 17
1.Enqueue 2.Dequeue 3.Display 4.Exit
Enter your option :
2
Deleted element = 15
1.Enqueue 2.Dequeue 3.Display 4.Exit
Enter your option :
2
Deleted element = 16
1.Enqueue 2.Dequeue 3.Display 4.Exit
Enter your option :
3
Elements in the queue : 17
1.Enqueue 2.Dequeue 3.Display 4.Exit
Enter your option :
2
Deleted element = 17
1.Enqueue 2.Dequeue 3.Display 4.Exit
Enter your option :
3
Queue is empty.
1.Enqueue 2.Dequeue 3.Display 4.Exit
Enter your option :
2
Queue is underflow.

Enter your option :

4

Test Case - 2

User Output

Enter the maximum size of the queue :

2

1.Enqueue 2.Dequeue 3.Display 4.Exit

Enter your option :

1

Enter element :

34

Successfully inserted.

1.Enqueue 2.Dequeue 3.Display 4.Exit

Enter your option :

1

Enter element :

56

Successfully inserted.

1.Enqueue 2.Dequeue 3.Display 4.Exit

Enter your option :

1

Enter element :

45

Queue is overflow.

1.Enqueue 2.Dequeue 3.Display 4.Exit

Enter your option :

3

Elements in the queue : 34 56

1.Enqueue 2.Dequeue 3.Display 4.Exit

Enter your option :

2

Deleted element = 34

1.Enqueue 2.Dequeue 3.Display 4.Exit

Enter your option :

2

Deleted element = 56

1.Enqueue 2.Dequeue 3.Display 4.Exit

Enter your option :

2

Queue is underflow.

1.Enqueue 2.Dequeue 3.Display 4.Exit

Enter your option :

2

Queue is underflow.

1.Enqueue 2.Dequeue 3.Display 4.Exit

Enter your option :

3

1.Enqueue	2.Dequeue	3.Display	4.Exit
Enter your option :			
1			
Enter element :			
56			
Successfully inserted.			
1.Enqueue	2.Dequeue	3.Display	4.Exit
Enter your option :			
3			
Elements in the queue : 56			
1.Enqueue	2.Dequeue	3.Display	4.Exit
Enter your option :			
4			

ID: 224G1A0535 Page No: 84

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

S.No: 21

Exp. Name: ***Write a C program to implement different Operations on Queue using Linked Lists***

Date: 2023-06-18

Aim:

Write a program to implement queue using **linked lists**.

Sample Input and Output:

```
1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit
Enter your option : 1
Enter element : 57
Successfully inserted.

1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit
Enter your option : 1
Enter element : 87
Successfully inserted.

1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit
Enter your option : 5
Queue size : 2

1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit
Enter your option : 3
Elements in the queue : 57 87

1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit
Enter your option : 2
Deleted value = 57

1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit
Enter your option : 2
Deleted value = 87

1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit
Enter your option : 3
Queue is empty.

1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit
Enter your option : 5
Queue size : 0

1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit
Enter your option : 6
```

Source Code:

QUsingLL.c

ID: 224G1A0535 Page No: 85

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

```

#include<conio.h>
#include<stdio.h>
struct queue
{
    int data;
    struct queue*next;
};
typedef struct queue *Q;
Q front = NULL, rear = NULL;
void enqueue(int element)
{
    Q temp = NULL;
    temp = (Q)malloc(sizeof(struct queue));
    if(temp == NULL)
    {
        printf("Queue is overflow.\n");
    }
    else
    {
        temp -> data = element;
        temp -> next = NULL;
        if(front == NULL)
        {
            front = temp;
        }
        else
        {
            rear -> next = temp;
        }
        rear = temp;
        printf("Successfully inserted.\n");
    }
}
void dequeue()
{
    Q temp = NULL;
    if(front == NULL)
    {
        printf("Queue is underflow.\n");
    }
    else
    {
        temp = front;
        if (front == rear)
        {
            front = rear = NULL;
        }
        else
        {
            front = front -> next;
        }
        printf("Deleted value = %d\n", temp -> data);
        free(temp);
    }
}

```

```

if(front == NULL)
{
    printf("Queue is empty.\n");
}
else
{
    Q temp = front;
    printf("Elements in the queue : ");
    while(temp != NULL)
    {
        printf("%d ", temp -> data);
        temp = temp -> next;
    }
    printf("\n");
}
void size()
{
    int count =0;
    if(front == NULL)
    {
        printf("Queue size : 0\n");
    }
    else
    {
        Q temp = front;
        while(temp != NULL)
        {
            temp = temp -> next;
            count = count + 1;
        }
        printf("Queue size : %d\n",count);
    }
}
void isEmpty()
{
    if(front == NULL )
    {
        printf("Queue is empty.\n");
    }
    else
    {
        printf("Queue is not empty.\n");
    }
}
int main()
{
    int op, x;
    while(1)
    {
        printf("1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit\n");
        printf("Enter your option : ");
        scanf("%d",&op);
        switch(op)
        {

```

```

        scanf("%d",&x);
        enqueue(x);
        break;
    case 2:
        dequeue();
        break;
    case 3:
        display();
        break;
    case 4:
        isEmpty();
        break;
    case 5:
        size();
        break;
    case 6: exit(0);
}
}
}

```

ID: 224G1A0535 | Page No: 88

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit
Enter your option :
2
Queue is underflow.
1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit
Enter your option :
3
Queue is empty.
1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit
Enter your option :
4
Queue is empty.
1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit
Enter your option :
5
Queue size : 0
1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit
Enter your option :
1
Enter element :
44
Successfully inserted.
1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit
Enter your option :
1

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

55

Successfully inserted.

1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit

Enter your option :

1

Enter element :

66

Successfully inserted.

1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit

Enter your option :

1

Enter element :

67

Successfully inserted.

1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit

Enter your option :

3

Elements in the queue : 44 55 66 67

1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit

Enter your option :

2

Deleted value = 44

1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit

Enter your option :

2

Deleted value = 55

1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit

Enter your option :

5

Queue size : 2

1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit

Enter your option :

4

Queue is not empty.

1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit

Enter your option :

6

Test Case - 2

User Output

1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit

Enter your option :

1

Enter element :

23

Successfully inserted.

1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit

Enter your option :

```
Enter element :  
234  
Successfully inserted.  
1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit  
Enter your option :  
1  
Enter element :  
45  
Successfully inserted.  
1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit  
Enter your option :  
1  
Enter element :  
456  
Successfully inserted.  
1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit  
Enter your option :  
2  
Deleted value = 23  
1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit  
Enter your option :  
3  
Elements in the queue : 234 45 456  
1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit  
Enter your option :  
2  
Deleted value = 234  
1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit  
Enter your option :  
3  
Elements in the queue : 45 456  
1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit  
Enter your option :  
4  
Queue is not empty.  
1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit  
Enter your option :  
5  
Queue size : 2  
1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit  
Enter your option :  
3  
Elements in the queue : 45 456  
1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit  
Enter your option :  
6
```

S.No: 22

Exp. Name: ***Reversing the links of a linked list***

Date: 2023-06-18

Aim:

Write a C program to reverse the links (not just displaying) of a linked list.

Note: Add node at the beginning.

Source Code:

reverseLinkedList.c

ID: 224G1A0535 Page No: 91

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

```

#include <stdio.h>
#include <stdlib.h>

struct Node
{
    int data;
    struct Node* next;
};

static void reverse(struct Node** head_ref)
{
    struct Node* prev = NULL;
    struct Node* current = *head_ref;
    struct Node* next = NULL;
    while (current != NULL)
    {
        next = current->next;
        current->next = prev;
        prev = current;
        current = next;
    }
    *head_ref = prev;
}

void push(struct Node** head_ref, int new_data)
{
    struct Node* new_node = (struct Node*) malloc(sizeof(struct Node));
    new_node->data = new_data;
    new_node->next = (*head_ref);
    (*head_ref) = new_node;
}

void printList(struct Node* head)
{
    struct Node* temp = head;
    while (temp != NULL)
    {
        printf("%d", temp->data);
        if (temp->next != NULL)
        {
            printf("->");
        }
        temp = temp->next;
    }
}

int main()
{
    struct Node* head = NULL;
    int i, count = 0, num = 0;
    printf("How many numbers you want to enter:");
    scanf(" %d", &count);
    for (i = 0; i < count; i++)
    {
        printf("Enter number %d:", i+1);
        num = ...
        push(&head, num);
    }
}

```

```

    }
    printf("Given linked list:");
    printList(head);
    reverse(&head);
    printf("\nReversed linked list:");
    printList(head);
}

```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
How many numbers you want to enter:
4
Enter number 1:
6
Enter number 2:
1
Enter number 3:
8
Enter number 4:
5
Given linked list:5->8->1->6
Reversed linked list:6->1->8->5

Test Case - 2
User Output
How many numbers you want to enter:
2
Enter number 1:
5
Enter number 2:
9
Given linked list:9->5
Reversed linked list:5->9

S.No: 23

Exp. Name: **Program to insert into BST and traversal using In-order, Pre-order and Post-order**

Date: 2023-06-18

Aim:

Write a program to create a binary search tree of integers and perform the following operations using linked list.

- 5. Insert a node
- 6. In-order traversal
- 7. Pre-order traversal
- 8. Post-order traversal

Source Code:

BinarySearchTree.c

ID: 224G1A0535 | Page No: 94

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

```

#include<stdio.h>
#include<stdlib.h>
struct node
{
    int data;
    struct node*left,*right;
};
typedef struct node *BSTNODE;
BSTNODE newNodeInBST(int item)
{
    BSTNODE temp = (BSTNODE)malloc(sizeof(struct node));
    temp->data = item;
    temp->left = temp->right = NULL;
    return temp;
}
void inorderInBST(BSTNODE root)
{
    if(root != NULL)
    {
        inorderInBST(root->left);
        printf("%d ", root->data);
        inorderInBST(root->right);
    }
}
void preorderInBST(BSTNODE root)
{
    if(root != NULL)
    {
        printf("%d ",root->data);
        preorderInBST(root->left);
        preorderInBST(root->right);
    }
}
void postorderInBST(BSTNODE root)
{
    if (root != NULL)
    {
        postorderInBST(root->left);
        postorderInBST(root->right);
        printf("%d ", root->data);
    }
}
BSTNODE insertNodeInBST(BSTNODE node, int ele)
{
    if (node == NULL)
    {
        printf("Successfully inserted.\n");
        return newNodeInBST(ele);
    }
    if (ele < node->data)
        node->left = insertNodeInBST(node->left,ele);
    else if (ele > node->data)
        node->right = insertNodeInBST(node->right,ele);
    else
        printf("Element already exists in BST.\n");
}

```

```

void main()
{
    int x, op;
    BSTNODE root = NULL;
    while(1)
    {
        printf("1.Insert 2.Inorder Traversal 3.Preorder Traversal 4.Postorder
Traversals 5.Exit\n");
        printf("Enter your option : ");
        scanf("%d", &op);
        switch(op)
        {
            case 1:
                printf("Enter an element to be inserted : ");
                scanf("%d", &x);
                root = insertNodeInBST(root,x);
                break;
            case 2:
                if(root == NULL)
                {
                    printf("Binary Search Tree is empty.\n");
                }
                else
                {
                    printf("Elements of the BST (in-order traversal): ");
                    inorderInBST(root);
                    printf("\n");
                }
                break;
            case 3:
                if(root == NULL)
                {
                    printf("Binary Search Tree is empty.\n");
                }
                else
                {
                    printf("Elements of the BST (pre-order traversal): ");
                    preorderInBST(root);
                    printf("\n");
                }
                break;
            case 4:
                if(root == NULL)
                {
                    printf("Binary Search Tree is empty.\n");
                }
                else
                {
                    printf("Elements of the BST (post-order traversal): ");
                    postorderInBST(root);
                    printf("\n");
                }
                break;
            case 5:
                exit(0);
        }
    }
}

```

}

Execution Results - All test cases have succeeded!

Test Case - 1	
User Output	
1.Insert 2.Inorder Traversal 3.Preorder Traversal 4.Postorder Traversal 5.Exit	
Enter your option :	
1	
Enter an element to be inserted :	
100	
Successfully inserted.	
1.Insert 2.Inorder Traversal 3.Preorder Traversal 4.Postorder Traversal 5.Exit	
Enter your option :	
1	
Enter an element to be inserted :	
20	
Successfully inserted.	
1.Insert 2.Inorder Traversal 3.Preorder Traversal 4.Postorder Traversal 5.Exit	
Enter your option :	
1	
Enter an element to be inserted :	
200	
Successfully inserted.	
1.Insert 2.Inorder Traversal 3.Preorder Traversal 4.Postorder Traversal 5.Exit	
Enter your option :	
1	
Enter an element to be inserted :	
10	
Successfully inserted.	
1.Insert 2.Inorder Traversal 3.Preorder Traversal 4.Postorder Traversal 5.Exit	
Enter your option :	
1	
Enter an element to be inserted :	
30	
Successfully inserted.	
1.Insert 2.Inorder Traversal 3.Preorder Traversal 4.Postorder Traversal 5.Exit	
Enter your option :	
1	
Enter an element to be inserted :	
150	
Successfully inserted.	
1.Insert 2.Inorder Traversal 3.Preorder Traversal 4.Postorder Traversal 5.Exit	
Enter your option :	
1	
Enter an element to be inserted :	

300

Successfully inserted.

1.Insert 2.Inorder Traversal 3.Preorder Traversal 4.Postorder Traversal 5.Exit

Enter your option :

2

Elements of the BST (in-order traversal): 10 20 30 100 150 200 300

1.Insert 2.Inorder Traversal 3.Preorder Traversal 4.Postorder Traversal 5.Exit

Enter your option :

3

Elements of the BST (pre-order traversal): 100 20 10 30 200 150 300

1.Insert 2.Inorder Traversal 3.Preorder Traversal 4.Postorder Traversal 5.Exit

Enter your option :

4

Elements of the BST (post-order traversal): 10 30 20 150 300 200 100

1.Insert 2.Inorder Traversal 3.Preorder Traversal 4.Postorder Traversal 5.Exit

Enter your option :

5

Page No: 98

ID: 224G1A0535

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

Test Case - 2

User Output

1.Insert 2.Inorder Traversal 3.Preorder Traversal 4.Postorder Traversal 5.Exit

Enter your option :

1

Enter an element to be inserted :

25

Successfully inserted.

1.Insert 2.Inorder Traversal 3.Preorder Traversal 4.Postorder Traversal 5.Exit

Enter your option :

1

Enter an element to be inserted :

63

Successfully inserted.

1.Insert 2.Inorder Traversal 3.Preorder Traversal 4.Postorder Traversal 5.Exit

Enter your option :

1

Enter an element to be inserted :

89

Successfully inserted.

1.Insert 2.Inorder Traversal 3.Preorder Traversal 4.Postorder Traversal 5.Exit

Enter your option :

1

Enter an element to be inserted :

45

Successfully inserted.

1.Insert 2.Inorder Traversal 3.Preorder Traversal 4.Postorder Traversal 5.Exit

Enter your option :

1

Enter an element to be inserted :

Successfully inserted.
1.Insert 2.Inorder Traversal 3.Preorder Traversal 4.Postorder Traversal 5.Exit
Enter your option :
1
Enter an element to be inserted :
28
Successfully inserted.
1.Insert 2.Inorder Traversal 3.Preorder Traversal 4.Postorder Traversal 5.Exit
Enter your option :
4
Elements of the BST (post-order traversal): 28 45 65 89 63 25
1.Insert 2.Inorder Traversal 3.Preorder Traversal 4.Postorder Traversal 5.Exit
Enter your option :
3
Elements of the BST (pre-order traversal): 25 63 45 28 89 65
1.Insert 2.Inorder Traversal 3.Preorder Traversal 4.Postorder Traversal 5.Exit
Enter your option :
2
Elements of the BST (in-order traversal): 25 28 45 63 65 89
1.Insert 2.Inorder Traversal 3.Preorder Traversal 4.Postorder Traversal 5.Exit
Enter your option :
5

S.No: 24

Exp. Name: **Write a Program to Search an element using Binary Search and Recursion**

Date: 2023-06-18

Aim:

Write a program to **search** the given element from a list of elements with **binary search** technique using **recursion**.

At the time of execution, the program should print the message on the console as:

Enter value of n :

For example, if the user gives the **input** as:

Enter value of n : 5

Next, the program should print the following messages one by one on the console as:

Enter 5 elements :

if the user gives the **input** as:

Enter 5 elements : 33 55 22 44 11

then the program should **print** the result as:

After sorting the elements are : 11 22 33 44 55

Next, the program should print the message on the console as:

Enter key element :

if the user gives the **input** as:

Enter key element : 11

then the program should **print** the result as:

The given key element 11 is found at position : 0

Similarly, if the key element is given as **18** for the above example then the program should print the output as:

The given key element 18 is not found

Note: Write the functions **read()**, **bubbleSort()**, **display()** and **binarySearch()** in [BinarySearch.c](#)

Source Code:

[BinarySearch.c](#)

Page No: 100

ID: 224G1A0535

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

```

#include <stdio.h>
void read(int a[20], int n)
{
    int i;
    printf("Enter %d elements : ", n);
    for (i = 0; i < n; i++)
    {
        scanf("%d", &a[i]);
    }
}
void bubbleSort(int a[20], int n)
{
    int i, j, temp;
    for (i = 0; i < n - 1; i++)
    {
        for (j = 0; j < n - i - 1; j++)
        {
            if (a[j] > a[j+1])
            {
                temp = a[j];
                a[j] = a[j+1];
                a[j+1] = temp;
            }
        }
    }
}
void display(int a[20], int n)
{
    int i;
    for (i = 0; i < n; i++)
    {
        printf("%d ", a[i]);
    }
    printf("\n");
}
int binarySearch(int a[20], int low, int high, int key)
{
    int mid;
    if(low <= high)
    {
        mid = (low + high) / 2;
        if (a[mid] == key)
            return mid;
        else if (key < a[mid])
            binarySearch(a, low, mid - 1, key);
        else if (key > a[mid])
            binarySearch(a, mid + 1, high, key);
    }
    else
    {
        return -1;
    }
}
void main()
{

```

```

scanf("%d", &n);
read(a, n);
bubbleSort(a, n);
printf("After sorting the elements are : ");
display(a, n);
printf("Enter key element : ");
scanf("%d", &key);
flag = binarySearch(a, 0, n - 1, key);
if (flag == -1)
{
    printf("The given key element %d is not found\n", key);
}
else
{
    printf("The given key element %d is found at position : %d\n", key, flag);
}

```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Enter value of n :
5
Enter 5 elements :
33 55 22 44 11
After sorting the elements are : 11 22 33 44 55
Enter key element :
11
The given key element 11 is found at position : 0

Test Case - 2
User Output
Enter value of n :
4
Enter 4 elements :
23 9 45 18
After sorting the elements are : 9 18 23 45
Enter key element :
24
The given key element 24 is not found

S.No: 25

Exp. Name: ***Graph traversals implementation - Breadth First Search***

Date: 2023-06-18

Aim:

Write a program to implement Breadth First Search of a graph.

Source Code:

GraphsBFS.c

Page No: 103

ID: 224G1A0535

2022-2026-CSE-A

```

#include <stdio.h>
#include <stdlib.h>
#define MAX 99
struct node
{
    struct node *next;
    int vertex;
};
typedef struct node * GNODE;
GNODE graph[20];
int visited[20];
int queue[MAX],front = -1,rear = -1;
int n;
void insertQueue(int vertex)
{
    if(rear == MAX-1)
        printf("Queue Overflow.\n");
    else
    {
        if(front == -1)
            front = 0;
        rear = rear+1;
        queue[rear] = vertex ;
    }
}
int isEmptyQueue()
{
    if(front == -1 || front > rear)
        return 1;
    else
        return 0;
}
int deleteQueue()
{
    int deleteItem;
    if(front == -1 || front > rear)
    {
        printf("Queue Underflow\n");
        exit(1);
    }
    deleteItem = queue[front];
    front = front+1;
    return deleteItem;
}
void BFS(int v)
{
    int w;
    insertQueue(v);
    while(!isEmptyQueue())
    {
        v = deleteQueue();
        printf("\n%d",v);
        visited[v]=1;
        GNODE g = graph[v];
        for(;g!=NULL;g=g->next)

```

```

        if(visited[w]==0)
        {
            insertQueue(w);
            visited[w]=1;
        }
    }
}

void main()
{
    int N, E, s, d, i, j, v;
    GNODE p, q;
    printf("Enter the number of vertices : ");
    scanf("%d",&N);
    printf("Enter the number of edges : ");
    scanf("%d",&E);
    for(i=1;i<=E;i++)
    {
        printf("Enter source : ");
        scanf("%d",&s);
        printf("Enter destination : ");
        scanf("%d",&d);
        q=(GNODE)malloc(sizeof(struct node));
        q->vertex=d;
        q->next=NULL;
        if(graph[s]==NULL)
        {
            graph[s]=q;
        }
        else
        {
            p=graph[s];
            while(p->next!=NULL)
            p=p->next;
            p->next=q;
        }
    }
    for(i=1;i<=n;i++)
    visited[i]=0;
    printf("Enter Start Vertex for BFS : ");
    scanf("%d", &v);
    printf("BFS of graph : ");
    BFS(v);
    printf("\n");
}
}

```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Enter the number of vertices :
5

5
Enter source :
1
Enter destination :
2
Enter source :
1
Enter destination :
4
Enter source :
4
Enter destination :
2
Enter source :
2
Enter destination :
3
Enter source :
4
Enter destination :
5
Enter Start Vertex for BFS :
1
BFS of graph :
1
2
4
3
5

Test Case - 2
User Output
Enter the number of vertices :
4
Enter the number of edges :
3
Enter source :
1
Enter destination :
2
Enter source :
2
Enter destination :
3
Enter source :
3
Enter destination :

Enter Start Vertex for BFS :

2

BFS of graph :

2

3

4

S.No: 26

Exp. Name: ***Graph traversals implementation - Depth First Search***

Date: 2023-06-22

Aim:

Write a program to implement Depth First Search for a graph.

Source Code:

GraphsDFS.c

Page No: 108

ID: 224G1A0535

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

```

#include<stdio.h>
#include<stdlib.h>
struct node
{
    struct node *next;
    int vertex;
};
typedef struct node * GNODE;
GNODE graph[20];
int visited[20];
int n;
void DFS(int i)
{
    GNODE p;
    printf("\n%d",i);
    p=graph[i];
    visited[i]=1;
    while(p!=NULL)
    {
        i=p->vertex;
        if(!visited[i])
            DFS(i);
        p=p->next;
    }
}
void main()
{
    int N,E,i,s,d,v;
    GNODE q,p;
    printf("Enter the number of vertices : ");
    scanf("%d",&N);
    printf("Enter the number of edges : ");
    scanf("%d",&E);
    for(i=1;i<=E;i++)
    {
        printf("Enter source : ");
        scanf("%d",&s);
        printf("Enter destination : ");
        scanf("%d",&d);
        q=(GNODE)malloc(sizeof(struct node));
        q->vertex=d;
        q->next=NULL;
        if(graph[s]==NULL)
            graph[s]=q;
        else
        {
            p=graph[s];
            while(p->next!=NULL)
                p=p->next;
            p->next=q;
        }
    }
    for(i=0;i<n;i++)
        visited[i]=0;
}

```

```
    printf("DFS of graph : ");
    DFS(v);
    printf("\n");
}
```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Enter the number of vertices :
6
Enter the number of edges :
7
Enter source :
1
Enter destination :
2
Enter source :
1
Enter destination :
4
Enter source :
4
Enter destination :
2
Enter source :
2
Enter destination :
3
Enter source :
4
Enter destination :
5
Enter source :
1
Enter destination :
3
Enter source :
3
Enter destination :
6
Enter Start Vertex for DFS :
1
DFS of graph :
1

3
6
4
5

Test Case - 2

User Output

Enter the number of vertices :

5

Enter the number of edges :

5

Enter source :

1

Enter destination :

2

Enter source :

1

Enter destination :

4

Enter source :

4

Enter destination :

2

Enter source :

2

Enter destination :

3

Enter source :

4

Enter destination :

5

Enter Start Vertex for DFS :

1

DFS of graph :

1

2

3

4

5

S.No: 27

Exp. Name: **Travelling Sales Person problem using
Dynamic programming**

Date: 2023-06-22

Aim:

Write a C program to implement **Travelling Sales Person** problem using **Dynamic programming**.

Source Code:

TSP.c

ID: 224G1A0535 Page No: 112

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

```

#include<stdio.h>
int ary[10][10], completed[10], n, cost = 0;
void takeInput()
{
    int i, j;
    printf("Number of villages: ");
    scanf("%d", &n);
    for (i = 0; i < n; i++)
    {
        for (j = 0; j < n; j++)
            scanf("%d", &ary[i][j]);
        completed[i] = 0;
    }
    printf("The cost list is:");
    for (i = 0; i < n; i++)
    {
        printf("\n");
        for (j = 0; j < n; j++)
            printf("\t%d", ary[i][j]);
    }
}
void mincost(int city)
{
    int i, ncity;
    completed[city] = 1;
    printf("%d-->", city + 1);
    ncity = least(city);
    if (ncity == 999)
    {
        ncity = 0;
        printf("%d", ncity + 1);
        cost += ary[city][ncity];
        return;
    }
    mincost(ncity);
}
int least(int c)
{
    int i, nc = 999;
    int min = 999, kmin;
    for (i = 0; i < n; i++)
    {
        if ((ary[c][i] != 0) && (completed[i] == 0))
            if (ary[c][i] + ary[i][c] < min)
            {
                min = ary[i][0] + ary[c][i];
                kmin = ary[c][i];
                nc = i;
            }
    }
    if (min != 999)
        cost += kmin;
        return nc;
}
int main()

```

```
printf("\nThe Path is:\n");
mincost(0);
printf("\nMinimum cost is %d", cost);
return 0;
}
```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Number of villages:
3
0 10 15
10 0 35
15 35 0
The cost list is:
0 10 15
10 0 35
15 35 0
The Path is:
1-->2-->3-->1
Minimum cost is 60

Aim:

Follow the instructions given below to write a program to **open** a file and to **print** its **contents** on the screen.

- Open a new file "**SampleText1.txt**" in write mode
- Write the content in the file
- Close the file
- Open the same file in read mode
- Read the content from file and print them on the screen
- Close the file

Source Code:

```
file1.c
```

```
#include<stdio.h>
void main() {
    FILE *fp;
    char ch;
    fp = fopen("SampleText1.txt", "w");
    printf("Enter the text with @ at end : ");
    while((ch = getchar()) != '@') {
        putc(ch, fp);
    }
    putc(ch, fp);
    fclose(fp);
    fp = fopen("SampleText1.txt", "r");
    printf("Given message is : ");
    while((ch = getc(fp)) != '@') {
        putchar(ch);
    }
    printf("\n");
    fclose(fp);
}
```

Execution Results - All test cases have succeeded!**Test Case - 1****User Output**

Enter the text with @ at end :
CodeTantra is a
Startup Company recognized by Government
of India@
Given message is : CodeTantra is a
Startup Company recognized by Government of India

Test Case - 2**User Output**

Enter the text with @ at end :

CodeTantra is
increasing development of Languages Year
by Year@
Given message is : CodeTantra is
increasing development of Languages Year
by Year

Aim:

Write a program to **copy** contents of one file into another file. Follow the instructions given below to write a program to copy the contents of one file to another file:

- Open a new file "**SampleTextFile1.txt**" in write mode
- Write the content onto the file
- Close the file
- Open an existing file "**SampleTextFile1.txt**" in read mode
- Open a new file "**SampleTextFile2.txt**" in write mode
- Copy the content from existing file to new file
- Close the files
- Open the copied file in read mode
- Read the text from file and print on the screen
- Close the file

Source Code:**CopyFile.c**

```
#include<stdio.h>
void main() {
    FILE *fp, *fp1, *fp2;
    char ch;
    fp = fopen("SampleTextFile1.txt", "w");
    printf("Enter the text with @ at end : ");
    while((ch=getc(stdin))!='@') {
        putc(ch, fp);
    }
    putc(ch, fp);
    fclose(fp);
    fp1 = fopen("SampleTextFile1.txt", "r");
    fp2 = fopen("SampleTextFile2.txt", "w");
    while ((ch=getc(fp1))!= '@') {
        putc(ch, fp2);
    }
    putc(ch, fp2);
    fclose(fp1);
    fclose(fp2);
    fp2 = fopen("SampleTextFile2.txt", "r");
    printf("Copied text is : ");
    while((ch=getc(fp2))!='@') {
        putchar(ch);
    }
    printf("\n");
    fclose(fp2);
}
```

Execution Results - All test cases have succeeded!**Test Case - 1****User Output**

Enter the text with @ at end :

CodeTantra started in the year 2014@

Copied text is : CodeTantra started in the year 2014

Test Case - 2

User Output

Enter the text with @ at end :

CodeTantra received

best Startup award from Hysea in 2016@

Copied text is : CodeTantra received

best Startup award from Hysea in 2016

Page No: 118

ID: 224G1A0535

2022-2026-CSE-A

S.No: 30

Exp. Name: **Write a C program to Merge two Files and stores their contents in another File**

Date: 2023-06-22

Aim:

Write a program to **merge** two files and stores their contents in another file.

- Open a new file "**SampleDataFile1.txt**" in write mode
- Write the content onto the file
- Close the file
- Open another new file "**SampleDataFile2.txt**" in write mode
- Write the content onto the file
- Close the file
- Open first existing file "**SampleDataFile1.txt**" in read mode
- Open a new file "**SampleDataFile3.txt**" in write mode
- Copy the content from first existing file to new file
- Close the first existing file
- Open another existing file "**SampleDataFile2.txt**" in read mode
- Copy its content from existing file to new file
- Close that existing file
- Close the merged file

Source Code:

Merge.c

ID: 224G1A0535 Page No: 119

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

```

#include <stdio.h>
void main()
{
    FILE *fp1, *fp2, *fp3;
    char ch;
    fp1 = fopen("SampleDataFile1.txt", "w");
    printf("Enter the text with @ at end for file-1 :\n");
    while ((ch = getchar()) != '@')
    {
        putc(ch, fp1);
    }
    putc(ch, fp1);
    fclose(fp1);
    fp2 = fopen("SampleDataFile2.txt", "w");
    printf("Enter the text with @ at end for file-2 :\n");
    while ((ch = getchar()) != '@')
    {
        putc(ch, fp2);
    }
    putc(ch, fp2);
    fclose(fp2);
    fp1 = fopen("SampleDataFile1.txt", "r");
    fp3 = fopen("SampleDataFile3.txt", "w");
    while ((ch = getc(fp1)) != '@')
    {
        putc(ch, fp3);
    }
    fclose(fp1);
    fp2 = fopen("SampleDataFile2.txt", "r");
    while ((ch = getc(fp2)) != '@')
    {
        putc(ch, fp3);
    }
    putc(ch, fp3);
    fclose(fp2);
    fclose(fp3);
    fp3 = fopen("SampleDataFile3.txt", "r");
    printf("Merged text is : ");
    while ((ch = getc(fp3)) != '@')
    {
        putchar(ch);
    }
    printf("\n");
    fclose(fp3);
}

```

Execution Results - All test cases have succeeded!

Test Case - 1

User Output

Enter the text with @ at end for file-1 :

CodeTantra developed an interactive tool

in the year 2014

CodeTantra got best Startup award in 2016@

Enter the text with @ at end for file-2 :

Now lot of Companies and Colleges using

CodeTantra Tool@

Merged text is : CodeTantra developed an interactive tool

in the year 2014

CodeTantra got best Startup award in 2016@

Now lot of Companies and Colleges using CodeTantra Tool

Aim:

Write a program to **delete** a **file**.

Note: Use the `remove(fileName)` function to delete an existing file.

Source Code:

Delete.c

```
#include<stdio.h>
void main() {
    FILE *fp;
    int status;
    char fileName[40], ch;
    printf("Enter a new file name : ");
    gets(fileName);
    fp = fopen("fileName.txt", "w");
    printf("Enter the text with @ at end : ");
    while((ch=getc(stdin))!= '@') {
        putc(ch,fp);
    }
    putc(ch,fp);
    fclose(fp);
    fp = fopen("fileName.txt", "r");
    printf("Given message is : ");
    while ((ch = fgetc(fp))!= '@') {
        putchar(ch);
    }
    printf("\n");
    fclose(fp);
    status = remove(fileName);
    if (status == 0)
        printf("unable to delete the file --");
    else {
        printf("%s file is deleted successfully\n",fileName);
    }
}
```

ID: 224G1A0535 Page No: 122

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Enter a new file name :
Text1.txt
Enter the text with @ at end :
This is CodeTantra@
Given message is : This is CodeTantra
Text1.txt file is deleted successfully

Test Case - 2

User Output

Enter a new file name :

Text2.txt

Enter the text with @ at end :

C developed by Dennis Ritchie@

Given message is : C developed by Dennis Ritchie

Text2.txt file is deleted successfully

Page No: 123

ID: 224G1A0535

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

S.No: 32

Exp. Name: **Write a C program to Copy last n characters from one File to another File**

Date: 2023-06-22

Aim:

Write a program to **copy** last **n** characters from **file-1** to **file-2**.

- open a new file "**TestDataFile1.txt**" in write mode
- write the content onto the file
- close the file
- open an existing file "**TestDataFile1.txt**" in read mode
- open a new file "**TestDataFile2.txt**" in write mode
- read the number of characters to copy
- set the cursor position by using fseek()
- copy the content from existing file to new file
- close the files
- open the copied file "**TestDataFile2.txt**" in read mode
- read the text from file and print on the screen
- close the file

Source Code:

Copy.c

ID: 224G1A0535 Page No: 124

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

```

#include <stdio.h>
void main()
{
    FILE *fp, *fp1, *fp2;
    int num, length;
    char ch;
    fp = fopen("TestDataFile1.txt", "w");
    printf("Enter the text with @ at end : ");
    while ((ch = getchar()) != '@')
    {
        putc(ch, fp);
    }
    putc(ch, fp);
    fclose(fp);
    fp1 = fopen("TestDataFile1.txt", "r");
    fp2 = fopen("TestDataFile2.txt", "w");
    printf("Enter number of characters to copy : ");
    scanf("%d", &num);
    fseek(fp1, 0L, SEEK_END);
    length = ftell(fp1);
    fseek(fp1, (length - num - 1), SEEK_SET);
    while ((ch = getc(fp1)) != '@')
    {
        putc(ch, fp2);
    }
    putc(ch, fp2);
    fclose(fp1);
    fclose(fp2);
    fp2 = fopen("TestDataFile2.txt", "r");
    printf("Copied text is : ");
    while ((ch = getc(fp2)) != '@')
    {
        putchar(ch);
    }
    printf("\n");
    fclose(fp2);
}

```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Enter the text with @ at end :
We should not give up
and we should not allow the problem to defeat us@
Enter number of characters to copy :
15
Copied text is : em to defeat us

Test Case - 2

User Output

Enter the text with @ at end :

You have to dream

before

Your dreams can come true@

Enter number of characters to copy :

20

Copied text is : dreams can come true

ID: 224G1A0535 Page No: 126

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

S.No: 33

Exp. Name: ***Write a C program to Reverse first n characters in a File***

Date: 2023-06-22

Aim:

Write a program to **reverse** the first **n** characters in a file.

- open a new file "**TestDataFile3.txt**" in read/write mode
- write the content onto the file
- read the number of characters to copy
- copy the specified number of characters into a string
- reverse the string
- overwrite the entire string into the file from the begining
- close the file
- open the copied file "**TestDataFile3.txt**" in read mode
- read the text from file and print on the screen
- close the file

Source Code:

Program1506.c

ID: 224G1A0535 Page No: 127

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

```

#include <stdio.h>
#include <string.h>
void stringReverse(char[]);
void main()
{
    FILE *fp;
    int num, i;
    char ch, data[100];
    fp = fopen("TestDataFile3.txt", "w+");
    printf("Enter the text with @ at end : ");
    while ((ch = getchar()) != '@')
    {
        putc(ch, fp);
    }
    putc(ch, fp);
    printf("Enter number of characters to copy : ");
    scanf("%d", &num);
    i = 0;
    rewind(fp);
    while (i < num)
    {
        data[i] = getc(fp);
        i++;
    }
    data[i] = '\0';
    rewind(fp);
    stringReverse(data);
    fputs(data, fp);
    fclose(fp);
    fp = fopen("TestDataFile3.txt", "r");
    printf("Result is : ");
    while ((ch = getc(fp)) != '@')
    {
        putchar(ch);
    }
    printf("\n");
    fclose(fp);
}
void stringReverse(char data[100])
{
    int i, j;
    char temp;
    i = j = 0;
    while (data[j] != '\0')
    {
        j++;
    }
    j--;
    while (i < j)
    {
        temp = data[i];
        data[i] = data[j];
        data[j] = temp;
        i++;
        j--;
    }
}

```

```
    }  
}
```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Enter the text with @ at end :
Teaching is a
very noble profession that shapes the
character, caliber and future of an individual@
Enter number of characters to copy :
18
Result is : yrev
a si gnihcaeT noble profession that shapes the
character, caliber and future of an individual

Test Case - 2
User Output
Enter the text with @ at end :
Small aim
is a crime; have great aim@
Enter number of characters to copy :
11
Result is : i
mia llamSs a crime; have great aim

ID: 224G1A0535 Page No: 129

2022-2026-CSE-A

Aim:

Write a program to **append** data to an existing file and **display** its contents.

- open a new file "**DemoTextFile1.txt**" in write mode
- write the content onto the file
- close the file
- open a new same file in append mode
- write the content onto the file
- close the file
- open the same file in read mode
- read the text from file and print them on the screen
- close the file

ID: 224G1A0535 | Page No: 130

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

Source Code:**appendDataToFile.c**

```
#include<stdio.h>
void main()
{
    FILE *fp;
    char ch;
    fp = fopen("DemoTextFile1.txt", "w");
    printf("Enter the text with @ at end : ");
    while ((ch = getchar()) != '@')
    {
        putc(ch, fp);
    }
    fclose(fp);
    fp = fopen("DemoTextFile1.txt", "a");
    printf("Enter the text to append to a file with @ at end : ");
    while ((ch = getchar()) != '@')
    {
        putc(ch, fp);
    }
    putc(ch, fp);
    fclose(fp);
    fp = fopen("DemoTextFile1.txt", "r");
    printf("File content after appending : ");
    while ((ch = getc(fp)) != '@')
    {
        putchar(ch);
    }
    printf("\n");
    fclose(fp);
}
```

Execution Results - All test cases have succeeded!**Test Case - 1****User Output**

Enter the text with @ at end :
I am studying@
Enter the text to append to a file with @ at end :
Life skills in University@
File content after appending : I am studying
Life skills in University

Test Case - 2
User Output
Enter the text with @ at end :
CodeTantra
developed@
Enter the text to append to a file with @ at end :
an interactive tool
to learn Programming@
File content after appending : CodeTantra
developed
an interactive tool
to learn Programming

Aim:

Write a program to **count** number of **characters**, **words** and **lines** of given text file.

- open a new file "**DemoTextFile2.txt**" in write mode
- write the content onto the file
- close the file
- open the same file in read mode
- read the text from file and find the characters, words and lines count
- print the counts of characters, words and lines
- close the file

Source Code:

countCharWordLines.c

```
#include<stdio.h>
void main()
{
    FILE *fp;
    char ch;
int charCount = 0, wordCount = 0, lineCount = 0;
fp = fopen("DemoTextFile2.txt", "w");
printf("Enter the text with @ at end : ");
while ((ch = getchar()) != '@')
{
    putc(ch, fp);
}
putc(ch, fp);
fclose(fp);
fp = fopen("DemoTextFile2.txt", "r");
do
{
    if ((ch == ' ') || (ch == '\n') || (ch == '@'))
        wordCount++;
    else
        charCount++;
    if (ch == '\n' || ch == '@')
        lineCount++;
} while ((ch = getc(fp)) != '@');
fclose(fp);
printf("Total characters : %d\n", charCount);
printf("Total words : %d\n", wordCount);
printf("Total lines : %d\n", lineCount);
}
```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Enter the text with @ at end :

Arise! Awake!
and stop not until
the goal is reached@
Total characters : 43
Total words : 10
Total lines : 3

Test Case - 2

User Output

Enter the text with @ at end :
All power is with in you
you can do anything
and everything@
Total characters : 48
Total words : 12
Total lines : 3

ID: 224G1A0535 Page No: 133

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

S.No: 36

Exp. Name: ***Linked list Female gender first***

Date: 2023-06-22

Aim:

Consider a linked list consisting of name of a person and gender as a node. Arrange the linked list using 'Ladies first' principle. You may create new linked lists if necessary.

Note: Add node at the beginning.

Source Code:

rearrangeList.c

ID: 224G1A0535 Page No: 134

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

```

#include <stdio.h>
#include <stdlib.h>
#include <string.h>

struct Node
{
    int data;
    char name[20];
    char gender;
    struct Node *next;
};

void segregateEvenOdd(struct Node **head_ref)
{
    struct Node *end = *head_ref;
    struct Node *prev = NULL;
    struct Node *curr = *head_ref;

    while (end->next != NULL)
        end = end->next;

    struct Node *new_end = end;

    while (curr->data %2 != 0 && curr != end)
    {
        new_end->next = curr;
        curr = curr->next;
        new_end->next->next = NULL;
        new_end = new_end->next;
    }

    if (curr->data%2 == 0)
    {
        *head_ref = curr;

        while (curr != end)
        {
            if ( (curr->data)%2 == 0 )
            {
                prev = curr;
                curr = curr->next;
            }
            else
            {
                prev->next = curr->next;

                curr->next = NULL;

                new_end->next = curr;

                new_end = curr;

                curr = prev->next;
            }
        }
    }
}

```

```

        prev = curr;
        if (new_end!=end && (end->data)%2 != 0)
        {
            prev->next = end->next;
            end->next = NULL;
            new_end->next = end;
        }
        return;
    }

void push(struct Node** head_ref, char new_name[20], char new_gender)
{
    struct Node* new_node = (struct Node*) malloc(sizeof(struct Node));
    strcpy(new_node->name, new_name);
    new_node->gender = new_gender;
    if (new_gender == 'F')
        new_node->data = 0;
    else if (new_gender == 'M')
        new_node->data = 1;
    new_node->next = (*head_ref);
    (*head_ref) = new_node;
}
void printList(struct Node *node)
{
    while (node!=NULL)
    {
        printf("%s (%c)", node->name, node->gender);
        node = node->next;
        if (node!=NULL)
            printf(" --> ");
    }
}

int main()
{
    struct Node* head = NULL;
    char name[20];
    char gender;
    int noOfInputs, i;
    int option;
    printf("Insert Data\n");
    do
    {
        printf("Enter Name: ");
        scanf(" %s", name);
        printf("Enter Gender: ");
        scanf(" %c", &gender);
        push(&head, name, gender);
        printf("1 : Insert into Linked List\n");
        printf("0 : Exit\n");
        printf("Enter your option: ");
        scanf(" %d", &option);
    } while(option == 1);
}

```

```

    segregateEvenOdd(&head);
    printf("\nModified Linked list \n");
    printList(head);
    printf("\n");
    return 0;
}

```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Insert Data
Enter Name:
Ganga
Enter Gender:
F
1 : Insert into Linked List
0 : Exit
Enter your option:
1
Enter Name:
Yamuna
Enter Gender:
F
1 : Insert into Linked List
0 : Exit
Enter your option:
1
Enter Name:
Raj
Enter Gender:
M
1 : Insert into Linked List
0 : Exit
Enter your option:
1
Enter Name:
Veer
Enter Gender:
M
1 : Insert into Linked List
0 : Exit

1
Enter Name:
Narmada
Enter Gender:
F
1 : Insert into Linked List
0 : Exit
Enter your option:
1
Enter Name:
Amar
Enter Gender:
M
1 : Insert into Linked List
0 : Exit
Enter your option:
0
Original Linked list
Amar (M) --> Narmada (F) --> Veer (M) --> Raj (M) --> Yamuna (F) --> Ganga (F)
Modified Linked list
Narmada (F) --> Yamuna (F) --> Ganga (F) --> Amar (M) --> Veer (M) --> Raj (M)

ID: 224G1A0535

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

Page No: 138

Test Case - 2	
User Output	
Insert Data	
Enter Name:	
Ganga	
Enter Gender:	
F	
1 : Insert into Linked List	
0 : Exit	
Enter your option:	
1	
Enter Name:	
Yamuna	
Enter Gender:	
F	
1 : Insert into Linked List	
0 : Exit	
Enter your option:	
1	
Enter Name:	
Narmada	
Enter Gender:	
F	
1 : Insert into Linked List	
0 : Exit	
Enter your option:	

Original Linked list
Narmada (F) --> Yamuna (F) --> Ganga (F)
Modified Linked list
Narmada (F) --> Yamuna (F) --> Ganga (F)

Test Case - 3	
User Output	
Insert Data	
Enter Name:	
Raj	
Enter Gender:	
M	
1 : Insert into Linked List	
0 : Exit	
Enter your option:	
1	
Enter Name:	
Veer	
Enter Gender:	
M	
1 : Insert into Linked List	
0 : Exit	
Enter your option:	
1	
Enter Name:	
Amar	
Enter Gender:	
M	
1 : Insert into Linked List	
0 : Exit	
Enter your option:	
0	
Original Linked list	
Amar (M) --> Veer (M) --> Raj (M)	
Modified Linked list	
Amar (M) --> Veer (M) --> Raj (M)	

Aim:

Write a C program to illustrate **Indexing of a file**.

Take an array of integers and find whether the given integer is present or not using **file indexing** method and print the output as shown in the sample output.

Source Code:

```
fileIndexing.c
```

```

#include <stdio.h>
#define MAX 25
struct indexfile
{
    int indexId;
    int kIndex;
};
int main()
{
    int numbers[MAX];
    struct indexfile index[MAX];
    int i, num, low, high, br = 4;
    int noOfStudents;
    printf("How many numbers do you want to enter:");
    scanf(" %d",&noOfStudents);
    printf("Enter %d numbers:", noOfStudents);
    for (i = 0; i < noOfStudents; i++)
    {
        scanf("%d", &numbers[i]);
    }

    for (i = 0; i < (noOfStudents/5); i++)
    {
        index[i].indexId = numbers[br];
        index[i].kIndex = br;
        br = br + 5;
    }
    printf("Enter a number to search:");
    scanf("%d",&num);
    for (i = 0; (i < noOfStudents/5) && (index[i].indexId <= num); i++)
    {
        if(i != 0)
            low = index[i - 1].kIndex;
        else
            low = 0;
        if(index[i].kIndex != 0 && index[i].kIndex <= noOfStudents)
            high = index[i].kIndex;
        else
            high = noOfStudents;
        for (i = low; i <= high; i++)
        {
            if (num == numbers[i])
            {
                printf("Number found at position:%d", i);
                return 0;
            }
        }
    }
    printf("\nNumber not found.");
    return 0;
}

```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
How many numbers do you want to enter:
5
Enter 5 numbers:
1 5 6 9 12
Enter a number to search:
6
Number found at position:2

Test Case - 2
User Output
How many numbers do you want to enter:
7
Enter 7 numbers:
2 3 6 9 12 20 25
Enter a number to search:
20
Number found at position:5

ID: 224G1A0535 Page No: 142

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

S.No: 38

Exp. Name: ***Write a C program to Convert an Infix expression into Postfix expression***

Date: 2023-06-22

Aim:

Write a program to convert an **infix** expression into **postfix** expression.

Source Code:

Infix2PostfixMain.c

ID: 224G1A0535

Page No: 143

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

```

#include<stdlib.h>
#include<string.h>
#include<stdio.h>
#include<ctype.h>
#define STACK_MAX_SIZE 20
#define STACK_MAX_SIZE 20
char stack [STACK_MAX_SIZE];
int top=-1;
int isEmpty() {
    if(top<0)
        return 1;
    else
        return 0;
}
void push(char x) {
    if(top == STACK_MAX_SIZE - 1) {
        printf("Stack is overflow.\n");
    } else {
        top = top + 1;
        stack[top] = x;
    }
}
char pop() {
    if(top < 0) {
        printf("Stack is underflow : unbalanced parenthesis\n");
        exit(0);
    }
    else
        return stack[top--];
}
int priority(char x) {
    if(x == '(')
        return 0;
    if(x == '+' || x == '-')
        return 1;
    if(x == '*' || x == '/' || x == '%')
        return 2;
}
void convertInfix(char * e) {
int x;
int k=0;
char * p = (char *)malloc(sizeof(char)*strlen(e));
while(*e != '\0') {
    if(isalnum(*e))
        p[k++]=*e;
    else if(*e == '(')
        push(*e);
    else if(*e == ')') {
        while(!isEmpty() && (x = pop()) != '(')
            p[k++]=x;
    }
    else if ((*e == '+' || *e == '-' || *e == '*' || *e == '/' || *e == '%'))
    {
        while(priority(stack[top]) >= priority(*e))
            p[k++]=pop();
    }
}
}

```

```

        else {
            printf("Invalid symbols in infix expression. Only
alphanumeric and { '+', '-', '*', '%%', '/' } are allowed.\n");

            exit(0);
        }
        e++;
    }
    while(top != -1) {
        x=pop();
        if(x == '(') {
            printf("Invalid infix expression :
unbalanced parenthesis.\n");
            exit(0);
        }
        p[k++] = x;
    }
    p[k++]= '\0';
    printf("Postfix expression :
%s\n",p);
}

int main() {
    char exp[20];
    char *e, x;
    printf("Enter the expression : ");
    scanf("%s",exp);
    e = exp;
    convertInfix(e);
}

```

ID: 224G1A0535 Page No: 145

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Enter the expression :
A+B*(C-D)
Postfix expression : ABCD-*+

Test Case - 2
User Output
Enter the expression :
A+B*C
Postfix expression : ABC*+

S.No: 39

Exp. Name: ***Infix to Prefix Conversion***

Date: 2023-06-22

Aim:

Write a C program to convert an Infix expression to Prefix expression.

Source Code:

infixToPrefix.c

ID: 224G1A0535

Page No: 146

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

```

#define SIZE 50
#include<string.h>
#include <ctype.h>
#include<stdio.h>
char *strrev(char *str)
{
    char c, *front, *back;
    if(!str || !*str)
    {
        return str;
    }
    for(front=str, back=str+strlen(str)-1; front < back; front++, back--)
    {
        c=*front;
        *front=*back;
        *back=c;
    }
    return str;
}
char s[SIZE];
int top = -1;
void push (char elem)
{
    s[++top] = elem;
}
char pop ()
{
    return (s[top--]);
}
int pr (char elem)
{
    switch (elem)
    {
        case '#':
            return 0;
        case ')':
            return 1;
        case '+':
        case '-':
            return 2;
        case '*':
        case '/':
            return 3;
    }
}
void main ()
{
    char infix[50], prefix[50], ch, elem;
    int i = 0, k = 0;
    printf ("Enter Infix Expression:");
    scanf ("%s", infix);
    push ('#');
    strrev (infix);
    while ((ch = infix[i++]) != '\0')
    {

```

```

        else if (isalnum (ch))
            prfx[k++] = ch;
        else if (ch == '(')
        {
            while (s[top] != ')')
            {
                prfx[k++] = pop ();
            }
            elem = pop ();
        }
        else
        {
            while (pr (s[top]) >= pr (ch))
            {
                prfx[k++] = pop ();
            }
            push (ch);
        }
    }
    while (s[top] != '#')
    {
        prfx[k++] = pop ();
    }
    prfx[k] = '\0';
    strrev (prfx);
    strrev (infx);
    printf ("Prefix Expression:%s\n", prfx);
}

```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Enter Infix Expression:
A+B
Prefix Expression:+AB

Test Case - 2
User Output
Enter Infix Expression:
A/B+C/D
Prefix Expression:+/AB/CD

S.No: 40

Exp. Name: ***Postfix to Infix Conversion***

Date: 2023-05-14

Aim:

Write a C program to convert a Postfix expression to Infix expression.

Source Code:

postfixToInfix.c

Page No: 149

ID: 224G1A0535

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

```

#include<stdio.h>
#include<conio.h>
#include<string.h>
#include<stdlib.h>
#define MAX 20
char str[MAX],stack[MAX];
int top=-1;
void push(char c) {
    stack[++top]=c;
}
char pop() {
    return stack[top--];
}
char *strrev(char *str){
    char c, *front, *back;

    if(!str || !*str)
        return str;
    for(front=str,back=str+strlen(str)-1;front < back;front++,back--){
        c=*front;*front=*back;*back=c;
    }
    return str;
}
void postfix() {
    int n,i,j=0;
    char a,b,op,x[20];
    printf("Enter a Postfix expression:");
    fflush(stdin);
    scanf("%s", str);
    strrev(str);
    n=strlen(str);
    for(i=0;i<MAX;i++) {
        stack[i]='\0';
    }
    printf("Infix expression:");
    for(i=0;i<n;i++) {
        if(str[i]=='+'||str[i]=='-'||str[i]=='*'||str[i]=='/') {
            push(str[i]);
        } else {
            x[j]=str[i]; j++;
            x[j]=pop(); j++;
        }
    }
    x[j]=str[top--];
    strrev(x);
    printf("%s\n",x);
}
void main() {
    postfix();
}

```

Execution Results - All test cases have succeeded!

Test Case - 1

User Output

Enter a Postfix expression:

AB+

Infix expression:A+B

Test Case - 2**User Output**

Enter a Postfix expression:

ABC*D+

Infix expression:A+B*C+D

Page No: 151

ID: 224G1A0535

Srinivasa Ramanujan Institute of Technology
2022-2026-CSE-A

Aim:

Write a C program to convert a Prefix expression to Infix expression.

Source Code:**prefixToInfix.c**

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
#include<stdlib.h>
#define MAX 20
char str[MAX],stack[MAX];
int top=-1;
void push(char c) {
    stack[++top]=c;
}
char pop() {
    return stack[top--];
}
void prefix() {
    int n,i;
    char a,b,op;
    printf("Enter a Prefix expression:");
//    fflush(stdin);
    scanf("%s", str);
    n=strlen(str);
    for(i=0;i<MAX;i++) {
        stack[i]='\0';
    }
    printf("Infix expression:");
    for(i=0;i<n;i++) {
        if(str[i]=='+'||str[i]=='-'||str[i]=='*'||str[i]=='/') {
            push(str[i]);
        } else {
            op=pop();
            a=str[i];
            if(op == '\0') {
                printf("%c",a);
            } else {
                printf("%c%c",a,op);
            }
        }
    }
    if(top >= 0) {
        printf("%c\n",str[top--]);
    } else {
        printf("\n");
    }
//printf("%c\n",str[top--]);
}
void main() {
    prefix();
}
```

ID: 224G1A0535 Page No: 152**2022-2026-CSE-A**

Srinivasa Ramanujan Institute of Technology

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Enter a Prefix expression:
+AB
Infix expression:A+B

Test Case - 2
User Output
Enter a Prefix expression:
+/AB/CD
Infix expression:A/B+C/D

ID: 224G1A0535 Page No: 153

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

S.No: 42

Exp. Name: ***Postfix to Prefix Conversion***

Date: 2023-05-14

Aim:

Write a C program to convert a Postfix expression to Prefix expression.

Source Code:

postfixToPrefix.c

ID: 224G1A0535 Page No: 154

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

```

#include<stdio.h>
#include<conio.h>
#include<string.h>
#include<stdlib.h>
# define MAX 20

char *strrev(char *str){
    char c, *front, *back;

    if(!str || !*str)
        return str;
    for(front=str, back=str+strlen(str)-1; front < back; front++, back--){
        c=*front; *front=*back; *back=c;
    }
    return str;
}

char str[MAX],stack[MAX];
int top=-1;
void push(char c) {
    stack[++top]=c;
}
char pop() {
    return stack[top--];
}
void post_pre() {
    int n,i,j=0; char c[20];
    char a,b,op;
    printf("Enter the postfix expression:");
    scanf("%s", str);
    n=strlen(str);
    for(i=0;i<MAX;i++)
        stack[i]='\0';
    printf("Prefix expression is:");
    for(i=n-1;i>=0;i--) {
        if(str[i]=='+'||str[i]=='-'||str[i]=='*'||str[i]=='/') {
            push(str[i]);
        } else {
            c[j++]=str[i];
            while((top!=-1)&&(stack[top]=='@')) {
                a=pop(); c[j++]=pop();
            }
            push('@');
        }
    }
    c[j]='\0';
    strrev(c);
    printf("%s\n",c);
}
void main() {
    post_pre();
}

```

Execution Results - All test cases have succeeded!

Test Case - 1

User Output

Enter the postfix expression:

AB+

Prefix expression is:+AB

Test Case - 2

User Output

Enter the postfix expression:

ABC*D+

Prefix expression is:++A*B*CD

Aim:

Write a C program to convert a Prefix expression to Postfix expression.

Source Code:

prefixToPostfix.c

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
#include<stdlib.h>
#define MAX 20
char str[MAX],stack[MAX];
int top=-1;
void push(char c) {
    stack[++top]=c;
}
char pop() {
    return stack[top--];
}
void pre_post() {
    int n,i,j=0; char c[20];
    char a,b,op;
    printf("Enter a Prefix expression:");
    scanf("%s", str);
    n=strlen(str);
    for(i=0;i<MAX;i++)
        stack[i]='\0';
    printf("Postfix expression is:");
    for(i=0;i<n;i++) {
        if(str[i]=='+'||str[i]=='-'||str[i]=='*'||str[i]=='/')
            push(str[i]);
        } else {
            c[j++]=str[i];
            while((top!=-1)&&(stack[top]=='@')) {
                a=pop(); c[j++]=pop();
            }
            push('@');
        }
    }
    c[j]='\0';
    printf("%s\n",c);
}
void main() {
    pre_post();
}
```

ID: 224G1A0535 Page No: 157

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

Execution Results - All test cases have succeeded!

Test Case - 1
User Output

Enter a Prefix expression:

+AB

Postfix expression is:AB+

Test Case - 2

User Output

Enter a Prefix expression:

+/AB/CD

Postfix expression is:AB/CD/+

Page No: 158

ID: 224G1A0535

2022-2026-CSE-A

