

S.No: 1	Exp. Name: <i>Design a C program which sorts the strings using array of pointers</i>	Date: 2023-05-01
---------	---	------------------

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
```

Source Code:

```
stringssort.c
```

ID: 224G1A0519 Page No: 1

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

```

#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++){
        printf("Enter string %d: ",i+1);
        strArray [i] = (char *)malloc(sizeof(char)*20);
        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

Page No. 4

ID: 224G1A0519

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

S.No: 2	Exp. Name: <i>Write a C program to Search a Key element using Linear search Technique</i>	Date: 2023-04-29
---------	--	------------------

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: 224G1A0519 Page No: 5

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

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

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

Page No. 7

ID: 224G1A0519

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

Srinivasa Ramanujan Institute of Technology

2022-2026-CSE-A

S.No: 3	Exp. Name: <i>Write a C program to Search a Key element using Binary search Technique</i>	Date: 2023-05-01
---------	--	------------------

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

```

#include<stdio.h>
void main()
{
    int a[5],i,j,n,temp,k,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: 224G1A0519 Page No: 11

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

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

S.No: 4	Exp. Name: <i>Write a C program to implement Fibonacci Search technique</i>	Date: 2023-04-29
---------	--	------------------

Aim:

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

Source Code:

FibonacciSearch.c

ID: 224G1A0519

Page No: 12

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

```
#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("Elements not found.\n");
}
```

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.

ID: 224G1A0519 Page No: 14

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

S.No: 5	Exp. Name: <i>Write a C program to Sort the elements using Insertion Sort Technique</i>	Date: 2023-04-29
---------	--	------------------

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: 224G1A0519 Page No: 15

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=1;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;
    }
}

```

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] :
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: 224G1A0519 Page No: 17

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

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: 2023-04-29
---------	--	------------------

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]);
    }
}
```

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

ID: 224G1A0519 Page No: 21

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

S.No: 7	Exp. Name: Write a C program to sort given elements using shell sort technique.	Date: 2023-05-02
---------	--	------------------

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

ID: 224G1A0519 Page No: 22

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);
    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");
}

```

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: <i>Write a C program to Sort the elements using Bubble Sort Technique</i>	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: 224G1A0519 Page No: 25

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

```
#include<stdio.h>
void main()
{
    int a[20],i,j,n,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\n",i,a[i]);
    }
    for(i=0;i<n-1;i++)
    {
        for(j=i+1;j<n;j++)
        {
            if(a[i]>a[j])
            {
                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\n",i,a[i]);
    }
}
```

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

ID: 224G1A0519 Page No: 27

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

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: 2023-05-09
---------	---	------------------

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[20],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

ID: 224G1A0519 Page No: 30

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

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

Srinivasa Ramanujan Institute of Technology

S.No: 10	Exp. Name: <i>Write a C program to sort the given elements using Heap sort</i>	Date: 2023-05-09
----------	---	------------------

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: 224G1A0519 Page No: 31

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

ID: 224G1A0519 Page No: 33

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

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: 2023-05-09
----------	---	------------------

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: 224G1A0519 Page No: 34

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++)
    {

```

```

}
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: 2023-05-16
----------	---	------------------

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: 224G1A0519 Page No: 37

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;
    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++)
        {

```

```

        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: <i>C program to performs all operations on singly linked list</i>	Date: 2023-05-18
----------	---	------------------

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: 224G1A0519 Page No: 41

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

```

#include<stdio.h>
#include<stdlib.h>
struct node {
    int data;
    struct node *next;
} *head = NULL, *tail = NULL;
void insert();
void Delete();
void display();
void count();
typedef struct node *NODE;
NODE temp, newNode, ptr, ptr2;
int value;
void main() {
    int option = 0;
    printf("Singly Linked List Example - All Operations\n");
    while(1) {
        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");
        printf("Enter your option : ");
        scanf("%d",&option);
        if(option<=5) {
            switch(option) {
                case 1:
                    insert();
                    break;
                case 2:
                    Delete();
                    break;
                case 3:
                    display();
                    break;
                case 4:
                    count();
                    break;
                case 5:
                    exit(0);
            }
        }
        else {
            printf("Enter options from 1 to 5\n");
            break;
        }
    }
}
void insert() {
    printf("Enter elements for inserting into linked list : ");
    scanf("%d",&value);
    newNode = (NODE) malloc(sizeof(struct node));
    newNode->data = value;
}

```

```

        head = newNode;
        tail = newNode;
    }
    else {
        tail->next = newNode;
        tail = newNode;
    }
}
void Delete() {
    int i = 1, j = 1, pos, spot, cnt = 0;
    temp = head, ptr2 = head;
    while(ptr2!=NULL) {
        cnt++;
        ptr2 = ptr2->next;
    }
    printf("Enter position of the element for deleteing the element : ");
    scanf("%d",&spot);
    while(i<=cnt) {
        if(i == spot){
            pos = spot;break;
        }
        i++;
    }
    if(pos != spot)
    printf("Invalid Position.\n");
    else {
        if(pos == 1){
            head = head->next;
            free(temp);
        }
        else{
            while(j<pos) {
                ptr = temp;
                temp = temp->next;
                j++;
            }
            if(temp->next == NULL) {
                ptr->next = NULL;
                free(temp);
            }
            else {
                ptr->next = temp->next;
                free(temp);
            }
        }
        printf("Deleted successfully\n");
    }
}
void display() {
    temp = head;
    printf("The elements in the linked list are : ");
    while(temp != NULL) {
        printf("%d ",temp->data);
        temp = temp->next;
    }
}

```

```

void count() {
    int count = 0;
    temp = head;
    while(temp != NULL) {
        count++;
        temp = temp->next;
    }
    printf("No of elements in the linked list are : %d\n",count);
}

```

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 :
111
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

```
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
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 : 111 333 444
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 :
```

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

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 :

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

```
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
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
```

S.No: 14	Exp. Name: C program which performs all operations on double linked list.	Date: 2023-05-19
----------	--	------------------

Aim:

Write a C program that uses functions to perform the following **operations on double linked list**
 i) Creationii) Insertioniii) Deletioniv) Traversal

Source Code:

AllOperationsDLL.c

ID: 224G1A0519 Page No: 48

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

```

#include<stdio.h>
#include<stdlib.h>
void insert();
void rem();
void display();
struct node
{
    int data;
    struct node*next;
    struct node*prev;
}
*head=NULL, *tail=NULL;
typedef struct node *NODE;
void main()
{
    int option=0;
    while(1)
    {
        printf("Operations on doubly linked list\n");
        printf("1. Insert \n");
        printf("2.Remove\n");
        printf("3. Display\n");
        printf("0. Exit\n");
        printf("Enter Choice 0-4? : ");
        scanf("%d",&option);
        switch(option)
        {
            case 1:
                insert();
                break;
            case 2:
                rem();
                break;
            case 3:
                display();
                break;
            case 0:
                exit(0);
        }
    }
}
void insert()
{
    NODE temp ,newNODE;
    int value;
    newNODE=(NODE)malloc(sizeof(struct node));
    printf("Enter number: ");
    scanf("%d",&value);
    newNODE->data=value;
    if(head==NULL)
    {
        newNODE->next=NULL;
        newNODE->prev=NULL;
        head=newNODE;
        tail=newNODE;
    }
}

```

```

    {
        tail->next=newNODE;
        newNODE->prev=tail;
        newNODE->next=NULL;
        tail=newNODE;
    }
}
void rem()
{
    int devalue,item;
    NODE temp,ptr;
    printf("Enter number to delete: ");
    scanf("%d",&item);
    ptr=head;
    while(ptr!=NULL)
    {
        if(ptr->data==item)
        {
            devalue=item;
            break;
        }
        ptr=ptr->next;
    }
    if(devalue!=item)
    printf("%d not found.\n" ,item);
    else
    {
        if(devalue==head->data)
        {
            temp=head;
            head=head->next;
            head->prev=NULL;
            free(temp);
        }
        else
        {
            if(devalue==tail->data)
            {
                temp=tail;
                tail=tail->prev;
                tail->next=NULL;
                free(temp);
            }
            else
            {
                temp=head;
                while(temp->data!=devalue)
                {
                    temp=temp->next;
                }
                temp->prev->next=temp->next;
                temp->next->prev=temp->prev;
                free(temp);
            }
        }
    }
}

```

```

void display()
{
    NODE temp;
    temp=head;
    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
3.Display
0.Exit
Enter Choice 0-4?:
1
Enter number:
15
Operations on doubly linked list
1.Insert
2.Remove

```
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
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
```

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: 224G1A0519 Page No: 53

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

S.No: 15	Exp. Name: C program to which performs all operations on Circular linked list.	Date: 2023-05-19
----------	---	------------------

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: 224G1A0519 Page No: 54

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)

```

```

        newnode->next = temp;
        while(temp->next != head)
            temp->next = newnode;
            head = newnode;
    }
    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");
        }
    }
}

```

```
{  
    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;  
    }  
    if(count == 1)  
    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");  
}
```

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
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

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

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: <i>Implementation of Circular Queue using Dynamic Array</i>	Date: 2023-05-19
----------	---	------------------

Aim:

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

ID: 224G1A0519 Page No: 60

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
```

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");
    }
}

```

```

void display()
{
    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;
        }
    }
}

```

```
        }  
    }  
}
```

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

```
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
```

S.No: 17	Exp. Name: <i>Write a C program to implement different Operations on Stack using Array representation</i>	Date: 2023-05-19
----------	--	------------------

Aim:

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

Sample Input and Output:

```

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

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

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

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

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

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

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

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

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

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

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

```

ID: 224G1A0519 Page No: 66

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 isEmpty()
{
    if(top < 0)
    {
        printf("Stack is empty.\n");
    }
    else

```

```

    }
}

void peek()
{
    int x;
    if(top < 0)
    {
        printf("Stack is underflow.\n");
    }
    else
    {
        x = arr[top];
        printf("Peek value = %d\n",x);
    }
}
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 :

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 :

```
Popped value = 10
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 :
4
Stack is empty.
1.Push 2.Pop 3.Display 4.IsEmpty 5.Peek 6.Exit
Enter your option :
6
```

ID: 224G1A0519 Page No: 70

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

S.No: 18	Exp. Name: <i>Write a C program to implement different Operations on Stack using Linked Lists</i>	Date: 2023-05-22
----------	--	------------------

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:

```
StackUsingLList.c
```

ID: 224G1A0519 Page No: 71

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

```

#include<stdio.h>
#include<stdlib.h>
void display();
void peek();
void isEmpty();
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);
        }
    }
}
struct stack {
    int data;
    struct stack *next;
};
typedef struct stack *stk;
stk top = NULL;
void 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() {
    if(top==NULL)
        printf("Stack is empty.\n");
}

```

```

        stk temp=top;
        while(temp!=NULL) {
            printf("%d ",temp->data);
            temp=temp->next;
        }
        printf("\n");
    }
}

void pop() {
    if(top==NULL)
        printf("Stack is underflow.\n");
    else {
        printf("Popped value = ");
        stk temp = top;
        top = top->next;
        printf("%d\n",temp->data);
        free(temp);
    }
}

void peek() {
    if(top==NULL)
        printf("Stack is underflow.\n");
    else {
        printf("Peek value = ");
        stk temp = top;
        printf("%d\n",temp->data);
    }
}

void isEmpty() {
    if(top==NULL)
        printf("Stack is empty.\n");
    else
        printf("Stack is not empty.\n");
}

```

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

Enter your option :

4

Stack is not empty.

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

Enter your option :

6

ID: 224G1A0519 Page No: 74

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

Test Case - 2

User Output

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

Enter your option :

2

Stack is underflow.

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

S.No: 19	Exp. Name: <i>Write a C program to implement different Operations on Queue using Array representation</i>	Date: 2023-06-12
----------	--	------------------

Aim:

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

Sample Input and 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 : 1
Enter element : 56
Successfully inserted.

1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit
Enter your option : 3
Elements in the queue : 23 56

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 : 2
Deleted element = 23

1.Enqueue 2.Dequeue 3.Display 4.IsEmpty 5.Size 6.Exit
Enter your option : 2
Deleted element = 56

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 : 6

```

Source Code:

```
QUsingArray.c
```

ID: 224G1A0519 Page No: 76

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

```

#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: 224G1A0519 Page No:78

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
```

ID: 224G1A0519 Page No: 79

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

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: 2023-06-12
----------	---	------------------

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:


```

#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);
        }
    }
}

```

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.

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

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

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 :

```
Queue is empty.  
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: 224G1A0519 Page No: 88

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

S.No: 21	Exp. Name: <i>Write a C program to implement different Operations on Queue using Linked Lists</i>	Date: 2023-06-15
----------	--	------------------

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: 224G1A0519 Page No: 89

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

```

#include<stdio.h>
#include<conio.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.Is Empty 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);
}
}
}

```

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

```

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

```

Page No:93

ID: 224G1A0519

2022-2026-CSE-A

Srinivasa Ramanujan Institute of Technology

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: 2023-06-15

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: 224G1A0519 Page No: 95

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);
        scanf("%d",&num);
        push(&head,num);
    }
    printf("Given linked list:");
    printList(head);
}

```

```
    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: <i>Program to insert into BST and traversal using In-order, Pre-order and Post-order</i>	Date: 2023-06-15
----------	--	------------------

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: 224G1A0519 Page No: 98

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 :	