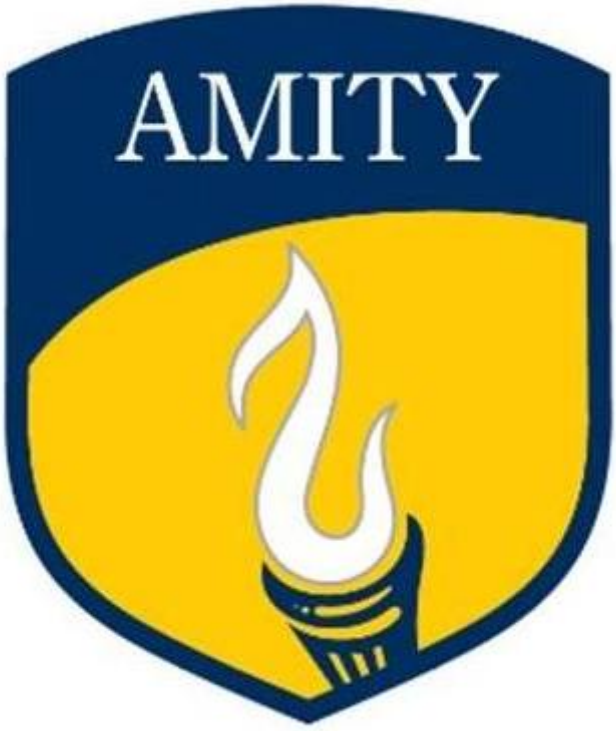
ADVANCED DATA STRUCTURE & ALGORITHMS LAB

LAB ASSIGNMENT FILE



SUBMITTED TO:

MR AKSHAT AGRAWAL
AMITY SCHOOL OF ENGINEERING & TECHNOLOGY

SUBMITTED BY:

HARSH - A501144824003
M.TECH(AI)
M.TECH 2024-2026

INDEX

S. NO.	NAME OF EXPERIMENT	DATE OF SUBMISSION	TEACHER'S SIGNATURE
1)	TO EXECUTE PUSH AND POP IN STACK	10-09-2024	
2)	INSERTION AND DELETION IN QUEUE	17-09-2024	
3)	INSERTION AND DELETION IN CIRCULAR QUEUE	24-09-2024	
4)	INSERTION AND DELETION IN DOUBLY ENDED QUEUE	01-10-2024	
5)	INSERTION IN LINKED LIST	08-10-2024	
6)	DELETION IN LINKED LIST	15-10-2024	
7)	BINARY SEARCH IN ARRAY	22-10-2024	
8)	SORT ARRAY USING BUBBLE SORT	05-11-2024	
9)	LINARY SEARCH IN ARRAY	12-11-2024	
10)	MERGE SOFT IN ARRAY SORTING	19-11-2024	

EXPERIMENT 1

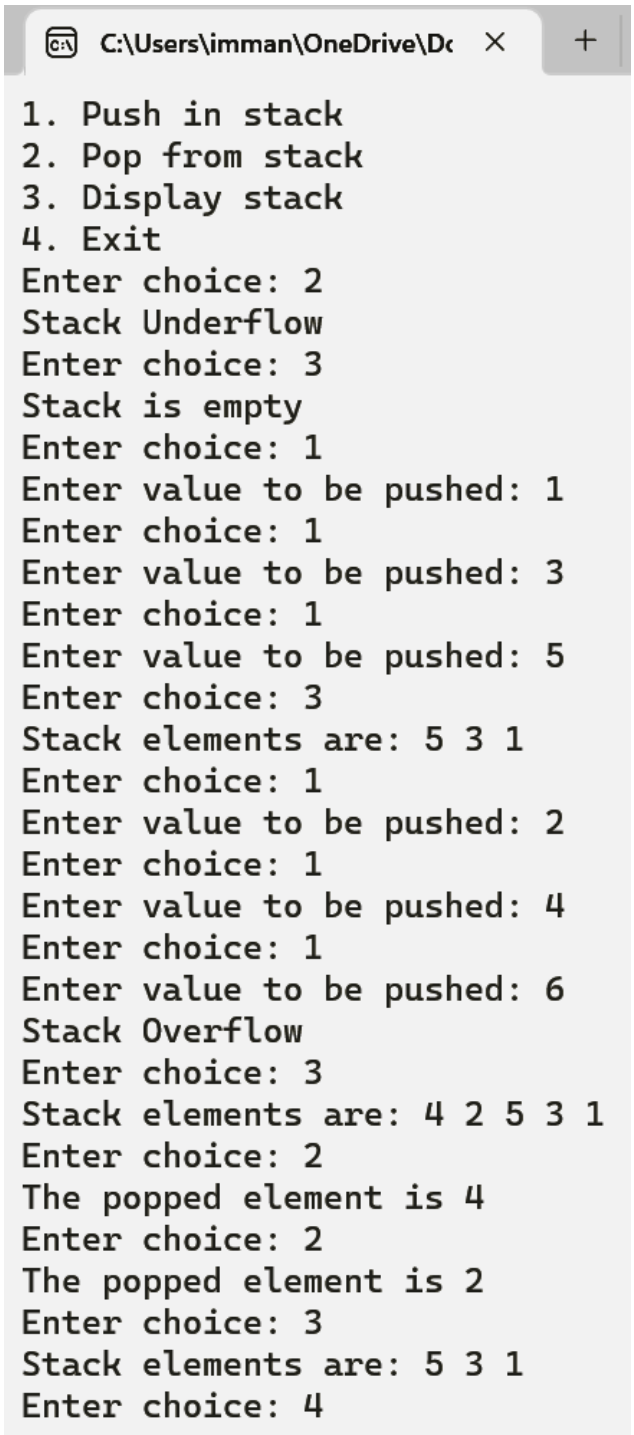
AIM: TO EXECUTE PUSH AND POP OPERATIONS IN STACK

SOURCE CODE:

```
#include <iostream>
using namespace std;
int stack[5],n=5,top=-1;
void push(int num)
{
    if(top>=n-1)
    cout<<"Stack Overflow"<<endl;
    else
    {
        top++;
        stack[top]=num;
    }
}
void pop()
{
    if(top<=-1)
    cout<<"Stack Underflow"<<endl;
    else
    {
        cout<<"The popped element is "<<stack[top]<<endl;
        top--;
    }
}
void display()
{
    if(top>=0)
    {
        cout<<"Stack elements are: ";
        for(int i=top;i>=0;i--)
            cout<<stack[i]<<" ";
        cout<<endl;
    }
    else
```

```
34          cout<<"Stack is empty"<<endl;
35      }
36      int main()
37      {
38          int ch,num;
39          cout<<"1. Push in stack"<<endl;
40          cout<<"2. Pop from stack"<<endl;
41          cout<<"3. Display stack"<<endl;
42          cout<<"4. Exit"<<endl;
43          while (ch!=4)
44          {
45              cout<<"Enter choice: ";
46              cin>>ch;
47              switch(ch)
48              {
49                  case 1:
50                  {
51                      cout<<"Enter value to be pushed: ";
52                      cin>>num;
53                      push(num);
54                      break;
55                  }
56                  case 2:
57                  {
58                      pop();
59                      break;
60                  }
61                  case 3:
62                  {
63                      display();
64                      break;
65                  }
66                  case 4:
67                  {
68                      cout<<"Goodbye!!!";
69                      break;
70                  }
71                  default:
72                  {
73                      cout<<"Invalid Choice"<<endl;
74                  }
75              }
76          }
77          return 0;
78      }
```

OUTPUT:


C:\Users\imman\OneDrive\Dc
1. Push in stack
2. Pop from stack
3. Display stack
4. Exit
Enter choice: 2
Stack Underflow
Enter choice: 3
Stack is empty
Enter choice: 1
Enter value to be pushed: 1
Enter choice: 1
Enter value to be pushed: 3
Enter choice: 1
Enter value to be pushed: 5
Enter choice: 3
Stack elements are: 5 3 1
Enter choice: 1
Enter value to be pushed: 2
Enter choice: 1
Enter value to be pushed: 4
Enter choice: 1
Enter value to be pushed: 6
Stack Overflow
Enter choice: 3
Stack elements are: 4 2 5 3 1
Enter choice: 2
The popped element is 4
Enter choice: 2
The popped element is 2
Enter choice: 3
Stack elements are: 5 3 1
Enter choice: 4
Goodbye!!!
Process exited after 22.75 seconds with return value 0
Press any key to continue . . .

EXPERIMENT 2

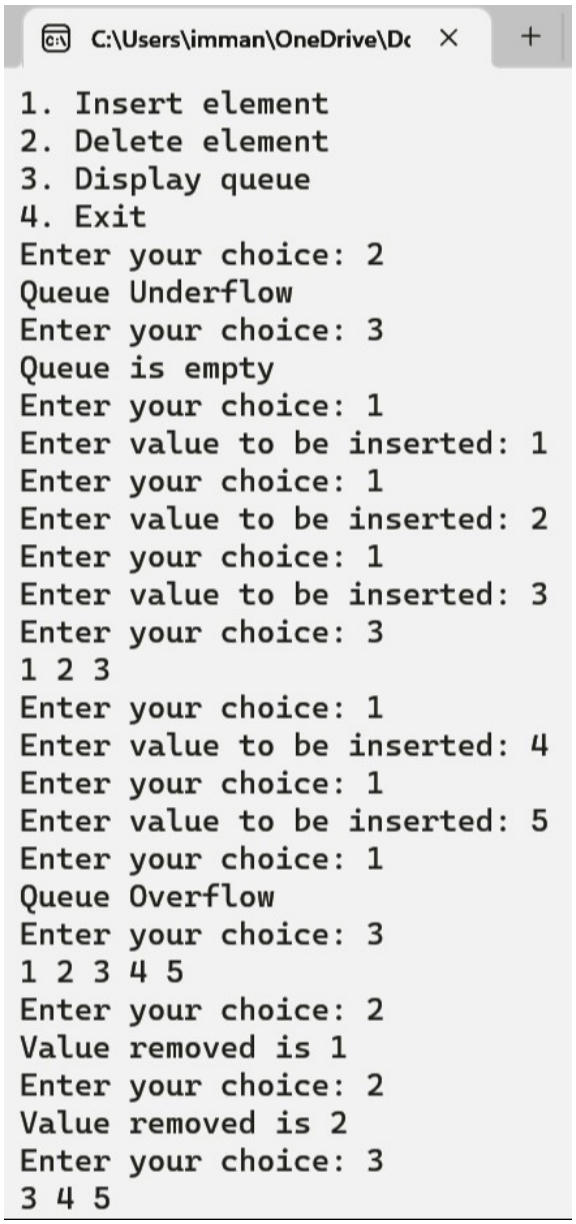
AIM: TO INSERT AND DELETE ELEMENTS IN A QUEUE

SOURCE CODE:

```
#include<iostream>
#define n 5
using namespace std;
int queue[n],front=-1,rear=-1;
void insert()
{
    int num;
    if (rear==n-1)
        cout<<"Queue Overflow"<<endl;
    else
    {
        if (front==-1)
            front=0;
        cout<<"Enter value to be inserted: ";
        cin>>num;
        rear+=1;
        queue[rear]=num;
    }
}
void remove()
{
    if (front==-1||front>rear)
    {
        cout<<"Queue Underflow"<<endl;
    }
    else
    {
        cout<<"Value removed is "<<queue[front]<<endl;
        front+=1;
    }
}
void display()
```

```
{
    int i;
    if (front==-1)
        cout<<"Queue is empty"<<endl;
    else
    {
        for (i=front;i<=rear;i++)
            cout<<queue[i]<<" ";
        cout<<endl;
    }
}
int main()
{
    int ch;
    cout<<"1. Insert element\n2. Delete element\n3. Display queue\n4. Exit\n";
    while (ch!=4)
    {
        cout<<"Enter your choice: ";
        cin>>ch;
        switch(ch)
        {
            case 1:
                insert();
                break;
            case 2:
                remove();
                break;
            case 3:
                display();
                break;
            case 4:
                cout<<"Goodbye!!!";
                break;
            default:
                cout<"Wrong input";
        }
    }
    return 0;
}
```

OUTPUT:




```
Enter your choice: 4
Goodbye!!!
--------------------------------
Process exited after 27.66 seconds with return value 0
Press any key to continue . . .
```

EXPERIMENT 3

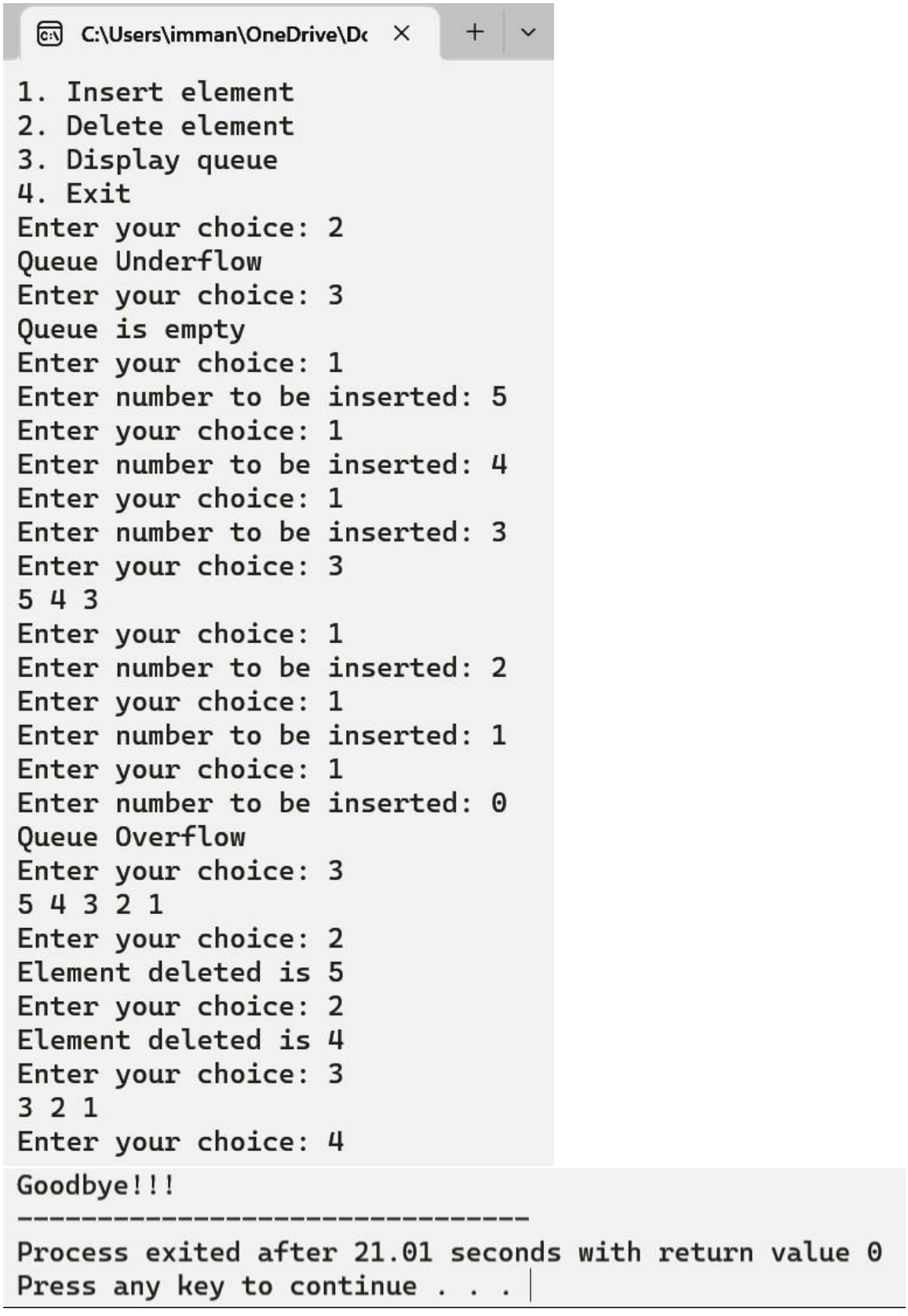
AIM: TO EXECUTE INSERTION AND DELETION OF VALUES IN CIRCULAR QUEUE

SOURCE CODE:

```
#include<iostream>
#define n 5
using namespace std;
int queue[n],front=-1,rear=-1;
void insert(int num)
{
    if((front==0&&rear==n-1)||(front==rear+1))
        cout<<"Queue Overflow"<<endl;
    else if (front==-1)
    {
        front=0;
        rear=0;
        queue[rear]=num;
    }
    else
    {
        if(rear==n-1)
            rear=0;
        else
            rear+=1;
        queue[rear]=num;
    }
}
void remove()
{
    if (front==-1)
        cout<<"Queue Underflow"<<endl;
    else
        cout<<"Element deleted is "<<queue[front]<<endl;
        if(front==rear)
        {
            front=-1;
            rear=-1;
```

```
        }
        else
        {
            if(front==n-1)
                front=0;
            else
                front+=1;
        }
}
void display()
{
    int f=front,r=rear;
    if(front==-1)
        cout<<"Queue is empty"<<endl;
    else if(f<=r)
    {
        while(f<=r)
        {
            cout<<queue[f]<<" ";
            f++;
        }
        cout<<endl;
    }
    else
    {
        while(f<=n-1)
        {
            cout<<queue[f]<<" ";
            f++;
        }
        f=0;
        while(f<=r)
        {
            cout<<queue[f]<<" ";
            f++;
        }
        cout<<endl;
    }
}
int main()
{
    int ch,num;
    cout<<"1. Insert element\n2. Delete element\n3. Display queue\n4. Exit\n";
    while (ch!=4)
    {
        cout<<"Enter your choice: ";
        cin>>ch;
        switch(ch)
        {
            case 1:
                cout<<"Enter number to be inserted: ";
                cin>>num;
                insert(num);
                break;
            case 2:
                remove();
                break;
            case 3:
                display();
                break;
            case 4:
                cout<<"Goodbye!!!";
                break;
            default:
                 cout<<"Wrong input"<<endl;
        }
    }
    return 0;
}
```

OUTPUT:


C:\Users\imman\OneDrive\Dc
1. Insert element
2. Delete element
3. Display queue
4. Exit
Enter your choice: 2
Queue Underflow
Enter your choice: 3
Queue is empty
Enter your choice: 1
Enter number to be inserted: 5
Enter your choice: 1
Enter number to be inserted: 4
Enter your choice: 1
Enter number to be inserted: 3
Enter your choice: 3
5 4 3
Enter your choice: 1
Enter number to be inserted: 2
Enter your choice: 1
Enter number to be inserted: 1
Enter your choice: 1
Enter number to be inserted: 0
Queue Overflow
Enter your choice: 3
5 4 3 2 1
Enter your choice: 2
Element deleted is 5
Enter your choice: 2
Element deleted is 4
Enter your choice: 3
3 2 1
Enter your choice: 4
Goodbye!!!

Process exited after 21.01 seconds with return value 0
Press any key to continue . . .

EXPERIMENT 4

AIM: TO DEMONSTRATE INSERTION AND DELETION OF ELEMENTS IN DEQUE

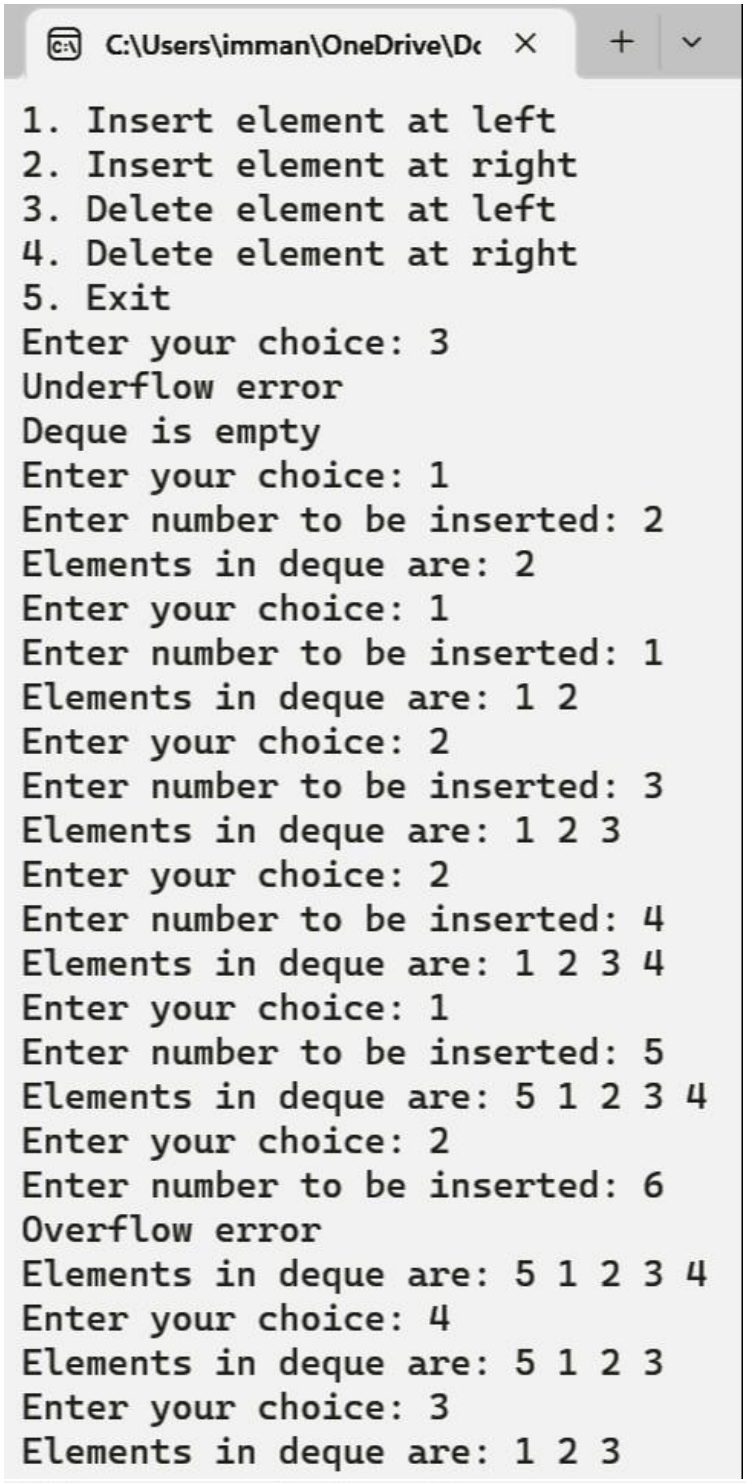
SOURCE CODE:

```
#include<iostream>
#define n 5
using namespace std;
int deque[n],l=-1,r=-1;
void display()
{
    int i=l;
    if (l==-1&&r==-1)
        cout<<"Deque is empty"<<endl;
    else
    {
        cout<<"Elements in deque are: ";
        while(i!=r)
        {
            cout<<deque[i]<<" ";
            i=(i+1)%n;
        }
        cout<<deque[r]<<endl;
    }
}
void insertleft(int x)
{
    if((l==0&&r==n-1)||(l==r+1))
    {
        cout<<"Overflow error"<<endl;
    }
    else
    {
        if((l==-1)&&(r==-1))
            l=r=0;
        else if(l==0)
            l=n-1;
        else
```

```
34              l--;
35          deque[l]=x;
36      }
37      display();
38  }
39  void insertright(int x)
40  {
41      if((l==0&&r==n-1)||(l==r+1))
42          cout<<"Overflow error"<<endl;
43      else
44      {
45          if(((l==-1)&&(r==-1))||(r==n-1))
46              r=0;
47          else
48              r++;
49          deque[r]=x;
50      }
51      display();
52  }
53  void deleteleft()
54  {
55      if(l==-1&&r==-1)
56          cout<<"Underflow error"<<endl;
57      else if(l==r)
58          l=r=-1;
59      else if(l==(n-1))
60          l=0;
61      else
62          l++;
63      display();
64  }
```

```
void deleteright()
{
    if(l==-1&&r==-1)
        cout<<"Underflow error"<<endl;
    else
    {
        if(l==r)
            l=r=-1;
        else if(r==0)
            r=n-1;
        else
            r--;
        display();
    }
}
int main()
{
    int ch,num;
    cout<<"1. Insert element at left\n2. Insert element at right\n3. Delete element at left\n4. Delete element at right\n5. Exit\n";
    while (ch!=5)
    {
        cout<<"Enter your choice: ";
        cin>>ch;
        switch(ch)
        {
            case 1:
                cout<<"Enter number to be inserted: ";
                cin>>num;
                insertleft(num);
                break;
            case 2:
                cout<<"Enter number to be inserted: ";
                cin>>num;
                insertright(num);
                break;
            case 3:
                deleteleft();
                break;
            case 4:
                deleteright();
                break;
            case 5:
                cout<<"Goodbye!!!";
                break;
            default:
                cout<<"Wrong input"<<endl;
        }
    }
    return 0;
}
```

OUTPUT:


C:\Users\imman\OneDrive\Dc
1. Insert element at left
2. Insert element at right
3. Delete element at left
4. Delete element at right
5. Exit
Enter your choice: 3
Underflow error
Deque is empty
Enter your choice: 1
Enter number to be inserted: 2
Elements in deque are: 2
Enter your choice: 1
Enter number to be inserted: 1
Elements in deque are: 1 2
Enter your choice: 2
Enter number to be inserted: 3
Elements in deque are: 1 2 3
Enter your choice: 2
Enter number to be inserted: 4
Elements in deque are: 1 2 3 4
Enter your choice: 1
Enter number to be inserted: 5
Elements in deque are: 5 1 2 3 4
Enter your choice: 2
Enter number to be inserted: 6
Overflow error
Elements in deque are: 5 1 2 3 4
Enter your choice: 4
Elements in deque are: 5 1 2 3
Enter your choice: 3
Elements in deque are: 1 2 3
Enter your choice: 5
Goodbye!!!

Process exited after 29.47 seconds with return value 0
Press any key to continue . . .

EXPERIMENT 5

AIM: TO CONDUCT INSERTION OPERATION IN LINKED LIST

SOURCE CODE:

```
#include <iostream>
using namespace std;
struct Node
{
    int data;
    struct Node* next;
};
void insertAtBeginning(struct Node** head_ref, int new_data)
{
    Node* new_node = new Node();
    new_node->data = new_data;
    new_node->next = (*head_ref);
    (*head_ref) = new_node;
}
void insertAfter(struct Node* prev_node, int new_data)
{
    if (prev_node == NULL)
    {
        cout << "the given previous node cannot be NULL";
        return;
    }
    struct Node* new_node = new Node();
    new_node->data = new_data;
    new_node->next = prev_node->next;
    prev_node->next = new_node;
}
void insertAtEnd(struct Node** head_ref, int new_data)
{
    struct Node* new_node = new Node();
    struct Node* last = *head_ref;
    new_node->data = new_data;
    new_node->next = NULL;
    if (*head_ref == NULL)
```

```
    {
        *head_ref = new_node;
        return;
    }
    while (last->next != NULL)
        last = last->next;
    last->next = new_node;
}
void printList(struct Node* node)
{
    cout<<"Linked list: ";
    while (node != NULL)
    {
        cout<<node->data<<" ";
        node=node->next;
    }
    cout<<endl;
}
int main()
{
    struct Node* head = NULL;
    insertAtBeginning(&head,5);
    printList(head);
    insertAtEnd(&head,1);
    cout<<"Inserted 1 at end"<<endl;
    insertAtBeginning(&head,2);
    cout<<"Inserted 2 at beginning"<<endl;
    printList(head);
    insertAtEnd(&head,4);
    cout<<"Inserted 4 at end"<<endl;
    insertAfter(head->next,3);
    cout<<"Inserted 3 after 5"<<endl;
    printList(head);
}
```

OUTPUT:

```
C:\Users\imman\OneDrive\Dc
Linked list: 5
Inserted 1 at end
Inserted 2 at beginning
Linked list: 2 5 1
Inserted 4 at end
Inserted 3 after 5
Linked list: 2 5 3 1 4

--------------------------------
Process exited after 0.1327 seconds with return value 0
Press any key to continue . . .
```

EXPERIMENT 6

AIM: TO CONDUCT DELETION OPERATION IN LINKED LIST

SOURCE CODE:

```
#include <iostream>
using namespace std;
struct Node
{
    int data;
    struct Node* next;
};
void insert(struct Node** head_ref, int new_data)
{
    Node* new_node = new Node();
    new_node->data = new_data;
    new_node->next = (*head_ref);
    (*head_ref) = new_node;
}
void printList(struct Node* node)
{
    cout<<"Linked list: ";
    while (node != NULL)
    {
        cout<<node->data<<" ";
        node=node->next;
    }
    cout<<endl;
}
void deleteAtBeginning(struct Node** head_ref)
{
    struct Node* temp;
    temp=*head_ref;
    *head_ref=temp->next;
    delete temp;
}
void deleteAtEnd(struct Node** head_ref)
{
    struct Node* end=*head_ref;
    struct Node* prev=NULL;
    while(end->next)
    {
        prev=end;
```

```
39          end=end->next;
40      }
41      prev->next=NULL;
42      delete end,prev;
43  }
44  void deleteNode(struct Node** head_ref,int key)
45  {
46      struct Node* temp=*head_ref,*prev;
47      while (temp!=NULL&&temp->data!=key)
48      {
49          prev=temp;
50          temp=temp->next;
51      }
52      prev->next=temp->next;
53      delete temp;
54  }
55  int main()
56  {
57      struct Node* head = NULL;
58      int i;
59      for (i=5;i>0;i--)
60          insert(&head,i);
61      printList(head);
62      deleteAtBeginning(&head);
63      cout<<"Deleted element at beginning"<<endl;
64      printList(head);
65      deleteAtEnd(&head);
66      cout<<"Deleted element at end"<<endl;
67      printList(head);
68      deleteNode(&head,3);
69      cout<<"Deleted element 3"<<endl;
70      printList(head);
71  }
```

OUTPUT:

```
C:\Users\imman\OneDrive\Dc
Linked list: 1 2 3 4 5
Deleted element at beginning
Linked list: 2 3 4 5
Deleted element at end
Linked list: 2 3 4
Deleted element 3
Linked list: 2 4

--------------------------------
Process exited after 0.06504 seconds with return value 0
Press any key to continue . . .
```

EXPERIMENT 7

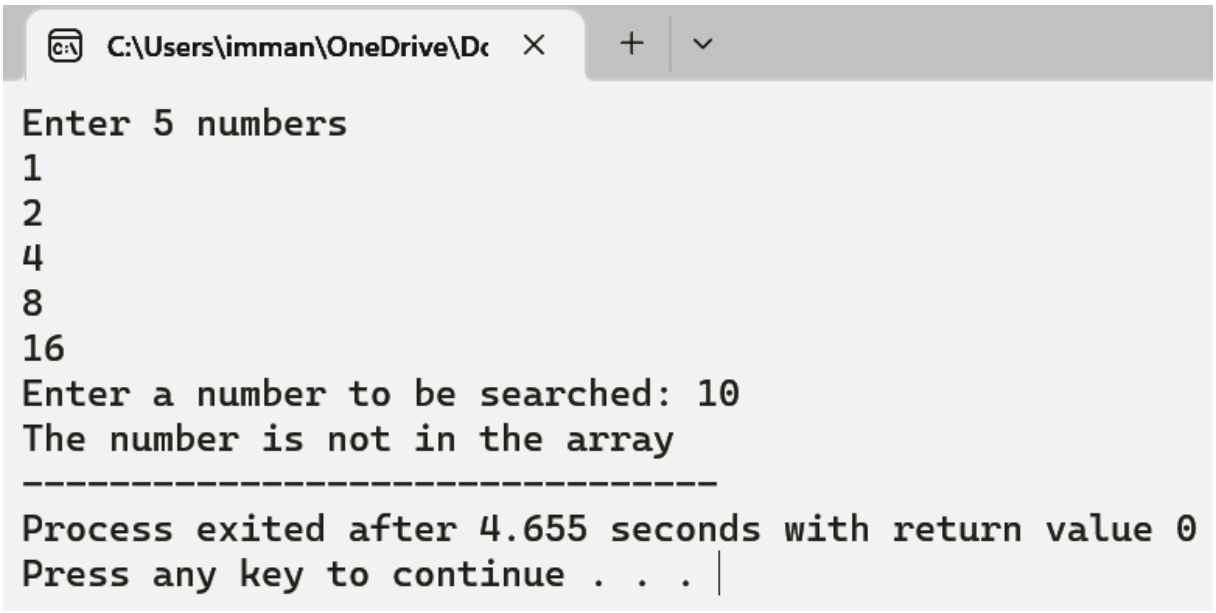
AIM: TO CONDUCT LINEAR SEARCH IN AN ARRAY

SOURCE CODE:

```
#include<iostream>
using namespace std;
int main()
{
    int a[5],i,n;
    cout<<"Enter 5 numbers\n";
    for (i=0;i<5;i++)
        cin>>a[i];
    cout<<"Enter a number to be searched: ";
    cin>>n;
    for (i=0;i<5;i++)
        if (a[i]==n)
        {
            cout<<"The number is at index "<<i;
            break;
        }
        else if (i==4)
            cout<<"The number is not in the array";
    return 0;
}
```

OUTPUT:

```
C:\Users\imman\OneDrive\Dc
Enter 5 numbers
1
2
4
8
16
Enter a number to be searched: 4
The number is at index 2
--------------------------------
Process exited after 8.468 seconds with return value 0
Press any key to continue . . .
```


C:\Users\imman\OneDrive\Dc
Enter 5 numbers
1
2
4
8
16
Enter a number to be searched: 10
The number is not in the array

Process exited after 4.655 seconds with return value 0
Press any key to continue . . .

EXPERIMENT 8

AIM: TO WRITE A PROGRAM TO EXECUTE BINARY SEARCH

SOURCE CODE:

```
from math import floor
print("Immanuel, A501132622002")
def rbs(list,i,low,high):
    if low>high:
        return None
    mid=floor((low+high)/2)
    if i==list[mid]:
        return mid
    if i>list[mid]:
        return rbs(list,i,mid+1,high)
    else:
        return rbs(list,i,low,mid-1)
a=0
try:
    arr=[]
    while True:
        arr.append(int(input("Enter element of array (non-int to stop): ")))
        a+=1
except:
    arr.sort()
    print("Array is: ",arr)
    n=int(input("Enter element to search for: "))
    print("Element is at position ",rbs(arr,n,0,a))
```

OUTPUT:

```
Immanuel, A501132622002
Enter element of array (non-int to stop): 4
Enter element of array (non-int to stop): 2
Enter element of array (non-int to stop): 6
Enter element of array (non-int to stop): 3
Enter element of array (non-int to stop): 9
Enter element of array (non-int to stop): 1
Enter element of array (non-int to stop): 147
Enter element of array (non-int to stop):
Array is:  [1, 2, 3, 4, 6, 9, 147]
Enter element to search for: 9
Element is at position  5
```

EXPERIMENT 9

AIM: TO SORT AN ARRAY USING BUBBLE SORT

SOURCE CODE:

```
#include <bits/stdc++.h>
using namespace std;
void bubbleSort(int arr[], int n)
{
    int i, j;
    bool swapped;
    for (i = 0; i < n - 1; i++)
    {
        swapped = false;
        for (j = 0; j < n - i - 1; j++)
        {
            if (arr[j] > arr[j + 1])
            {
                swap(arr[j], arr[j + 1]);
                swapped = true;
            }
        }
        if (swapped == false)
            break;
    }
}
void printArray(int arr[], int size)
{
    int i;
    for (i = 0; i < size; i++)
        cout << " " << arr[i];
}
int main()
{
    int arr[] = { 64, 34, 25, 12, 22, 11, 90 };
    int N = sizeof(arr) / sizeof(arr[0]);
    cout << "Original array: ";
    printArray(arr, N);
    bubbleSort(arr, N);
    cout << "\nSorted array: ";
    printArray(arr, N);
    return 0;
}
```

OUTPUT:

```
C:\Users\imman\OneDrive\Dc
Original array:  64 34 25 12 22 11 90
Sorted array:  11 12 22 25 34 64 90
--------------------------------
Process exited after 0.1249 seconds with return value 0
Press any key to continue . . .
```

EXPERIMENT 10

AIM: TO SORT AN ARRAY USING MERGE SORT

SOURCE CODE:

```
print("Immanuel, A501132622002")
def mergesort(arr,x,z):
    if x<z:
        y=(x+z)//2
        mergesort(arr,x,y)
        mergesort(arr,y+1,z)
        merge(arr,x,y,z)
def merge(arr,x,y,z):
    a=y-x+1
    b=z-y
    l=[0]*a
    r=[0]*b
    for i in range(0,a):
        l[i]=arr[x+i]
```


```
print("Immanuel, A501132622002")
def mergesort(arr,x,z):
    if x<z:
        y=(x+z)//2
        mergesort(arr,x,y)
        mergesort(arr,y+1,z)
        merge(arr,x,y,z)
def merge(arr,x,y,z):
    a=y-x+1
    b=z-y
    l=[0]*a
    r=[0]*b
    for i in range(0,a):
        l[i]=arr[x+i]
```

```
            i += 1
            k += 1
        while j < b:
            arr[k] = r[j]
            j += 1
            k += 1
a=0
try:
    arr=[]
    while True:
        arr.append(int(input("Enter element of array (non-int to stop): ")))
        a+=1
except:
    print("Array is: ",arr)
    mergesort(arr,0,a-1)
    print("Sorted array is: ",arr)
```

OUTPUT:

```
Immanuel, A501132622002
Enter element of array (non-int to stop): 3
Enter element of array (non-int to stop): 2
Enter element of array (non-int to stop): 5
Enter element of array (non-int to stop): 7
Enter element of array (non-int to stop): 4
Enter element of array (non-int to stop): 49
Enter element of array (non-int to stop): 5
Enter element of array (non-int to stop):
Array is:  [3, 2, 5, 7, 4, 49, 5]
Sorted array is:  [2, 3, 4, 5, 5, 7, 49]
```