NAME: MOAZZAM FAROOQUI

ROLLNO: CT-24068

COURSE CODE: CT-159

ASSIGNMENT: DSA LAB#03

INSTRUCTOR: SAYYDA SAHAR FATIMA

Q1.

SOURCE CODE:

```cpp
#include<iostream>
using namespace std;

class Node{
    public:
        int data;
        Node*next;
        Node(int val){
            data=val;
            next=nullptr;
        }
};

class CircularQueue{
    private:
        Node*front;
        Node*rear;
    public:
        CircularQueue(){
            front=nullptr;
            rear=nullptr;
        }
        void enqueue(int val){
            Node*newnode=new Node(val);
            if(front==nullptr){
                front=newnode;
                rear=newnode;
                rear->next=front;
                return;
            }
            rear->next=newnode;
            rear=newnode;
            rear->next=front;
        }
        void dequeue(){
            if(front==nullptr){
                cout<<"QUEUE IS EMPTY!"<<endl;
                return;
            }
            if(front==rear){
                delete front;
                front=nullptr;
                rear=nullptr;
                return;
            }
            Node*temp=front;
            front=front->next;
            rear->next=front;
            delete temp;
        }

        int peek(){
            if(isEmpty()){
                cout<<"QUEUE IS EMPTY!"<<endl;
                return -1;
            }
            return front->data;
        }

        bool isEmpty(){
            return front==nullptr;
        }
```

```
65          void display(){
66              if(isEmpty()){
67                  cout<<"QUEUE IS EMPTY!"<<endl;
68                  return;
69              }
70              Node*temp=front;
71              while(true){
72                  cout<<temp->data<<" ";
73                  temp=temp->next;
74                  if(temp==front)
75                      break;
76              }
77              cout<<endl;
78          }
79  };
80
81  int main(void){
82      CircularQueue cq;
83      cq.enqueue(1);
84      cq.enqueue(2);
85      cq.enqueue(3);
86      cq.dequeue();
87
88      cq.display();
89      cq.peek();
90      cq.isEmpty();
91      return 0;
92
93  }
```

OUTPUT:

```
2 3

--------------------------------
Process exited after 8.106 seconds with return value 0
Press any key to continue . . .
```

Q2.

SOURCE CODE:

```
1   #include<iostream>
2   using namespace std;
3
4   class Node{
5       public:
6           int data;
7           Node*next;
8           Node*prev;
9       Node(int val){
10          data=val;
11          next=nullptr;
12          prev=nullptr;
13      }
14  };
15
16  class DoubleEndedQueue{
17      private:
18          Node*front;
19          Node*rear;
20      public:
21          DoubleEndedQueue(){
22              front=nullptr;
23              rear=nullptr;
24          }
```

```
26      void insert_at_front(int val){
27          Node*newnode=new Node(val);
28          if(front==nullptr){
29              front=newnode;
30              rear=newnode;
31              return;
32          }
33          newnode->next=front;
34          front->prev=newnode;
35          front=newnode;
36          return;
37      }
38
39      void insert_at_rear(int val){
40          Node*newnode=new Node(val);
41          if(front==nullptr){
42              front=newnode;
43              rear=newnode;
44              return;
45          }
46          rear->next=newnode;
47          newnode->prev=rear;
48          rear=newnode;
49          return;
50      }
```

```
52      void delete_at_front(){
53          if(front==nullptr){
54              cout<<"QUEUE IS EMPTY!"<<endl;
55              return;
56          }
57          if(front==rear){
58              delete front;
59              front=nullptr;
60              rear=nullptr;
61              return;
62          }
63          front=front->next;
64          delete front->prev;
65          front->prev=nullptr;
66          return;
67      }
68
69      void delete_at_rear(){
70          if(front==nullptr){
71              cout<<"QUEUE IS EMPTY!"<<endl;
72              return;
73          }
74          if(front==rear){
75              delete front;
76              front=nullptr;
77              rear=nullptr;
78              return;
79          }
80          rear=rear->prev;
81          delete rear->next;
82          rear->next=nullptr;
83          return;
84      }
```

```
        void display(){
            Node*temp=front;
            while(temp!=nullptr){
                cout<<temp->data<<" ";
                temp=temp->next;
            }
            cout<<endl;
        }
};

int main(void){
    DoubleEndedQueue ceq;
    ceq.insert_at_front(1);
    ceq.insert_at_front(2);
    ceq.insert_at_rear(3);
    ceq.delete_at_front();
    ceq.display();

    ceq.insert_at_rear(4);
    ceq.delete_at_rear();
    return 0;
}
```

OUTPUT:

```
1 3


--------------------------------
Process exited after 8.018 seconds with return value 0
Press any key to continue . . .
```

Q3.

SOURCE CODE:

```
#include<iostream>
using namespace std;

class Node{
   public:
    int data;
    Node*next;
    Node*prev;

    Node(int val){
        data=val;
        next=nullptr;
        prev=nullptr;
    }
};

class DoublyLinkedList{
    private:
        Node*head;
        Node*tail;

    public:
        DoublyLinkedList(){
            head=nullptr;
            tail=nullptr;
        }
```

```
28          void insertrear(int val){
29              Node*newnode=new Node(val);
30              if(head==nullptr){
31                  head=newnode;
32                  tail=newnode;
33                  return;
34              }
35              tail->next=newnode;
36              newnode->prev=tail;
37              tail=newnode;
38          }
39
40          void concatenate(DoublyLinkedList&other){
41              if(other.head==nullptr)return;
42              if(this->head==nullptr){
43                  head=other.head;
44                  tail=other.tail;
45                  return;
46              }
47              tail->next=other.head;
48              other.head->prev=tail;
49              tail=other.tail;
50          }
51
52          void display(){
53              Node*temp=head;
54              while(temp!=nullptr){
55                  cout<<temp->data<<" ";
56                  temp=temp->next;
57              }
58              cout<<endl;
59          }
60  };
```

```
62  int main(void){
63      DoublyLinkedList L;
64      DoublyLinkedList M;
65      DoublyLinkedList N;
66
67      for(int i=2;i<=10;i+=2){
68          L.insertrear(i);
69      }
70
71      for(int i=1;i<=9;i+=2){
72          M.insertrear(i);
73      }
74
75      cout<<"LIST L(EVEN):"<<endl;
76      L.display();
77
78      cout<<"LIST M(ODD):"<<endl;
79      M.display();
80
81      N.concatenate(L);
82      N.concatenate(M);
83      cout<<"LIST N:"<<endl;
84      N.display();
85      return 0;
86  }
```

OUTPUT:

```
LIST L(EVEN):
2 4 6 8 10
LIST M(ODD):
1 3 5 7 9
LIST N:
2 4 6 8 10 1 3 5 7 9

--------------------------------
Process exited after 8.056 seconds with return value 0
Press any key to continue . . .
```

Q4.

SOURCE CODE:

```cpp
#include<iostream>
using namespace std;

class Node{
    public:
     int data;
     Node*next;
     Node*prev;

    Node(int val){
        data=val;
        next=nullptr;
        prev=nullptr;
    }
};

class DoublyLinkedList{
    private:
        Node*head;
        Node*tail;

    public:
        DoublyLinkedList(){
            head=nullptr;
            tail=nullptr;
        }

        void insertrear(int val){
            Node*newnode=new Node(val);
            if(head==nullptr){
                head=newnode;
                tail=newnode;
                return;
            }
            tail->next=newnode;
            newnode->prev=tail;
            tail=newnode;
        }

        void concatenate(DoublyLinkedList&other){
            if(other.head==nullptr)return;
            if(this->head==nullptr){
                head=other.head;
                tail=other.tail;
                return;
            }
            tail->next=other.head;
            other.head->prev=tail;
            tail=other.tail;
        }
```

```
52          void sortdescending(){
53              if(head==nullptr)return;
54
55              for(Node*i=head;i!=nullptr;i=i->next){
56                  for(Node*j=i->next;j!=nullptr;j=j->next){
57                      if(i->data<j->data){
58                          int temp=i->data;
59                          i->data=j->data;
60                          j->data=temp;
61                      }
62                  }
63              }
64          }
65
66          void display(){
67              Node*temp=head;
68              while(temp!=nullptr){
69                  cout<<temp->data<<" ";
70                  temp=temp->next;
71              }
72              cout<<endl;
73          }
74  };
```

```
76  int main(void){
77      DoublyLinkedList L;
78      DoublyLinkedList M;
79      DoublyLinkedList N;
80
81      for(int i=2;i<=10;i+=2){
82          L.insertrear(i);
83      }
84
85      for(int i=1;i<=9;i+=2){
86          M.insertrear(i);
87      }
88
89      cout<<"LIST L(EVEN):"<<endl;
90      L.display();
91
92      cout<<"LIST M(ODD):"<<endl;
93      M.display();
94
95      N.concatenate(L);
96      N.concatenate(M);
97      cout<<"LIST N:"<<endl;
98      N.display();
99      cout<<"LIST N IN DESCENDING ORDER:"<<endl;
100     N.sortdescending();
101     N.display();
102     return 0;
103 }
```

OUTPUT:

```
LIST L(EVEN):
2 4 6 8 10
LIST M(ODD):
1 3 5 7 9
LIST N:
2 4 6 8 10 1 3 5 7 9
LIST N IN DESCENDING ORDER:
10 9 8 7 6 5 4 3 2 1

--------------------------------
Process exited after 8.023 seconds with return value 0
Press any key to continue . . .
```

Q5.

SOURCE CODE:

```cpp
#include<iostream>
using namespace std;

class Node{
    public:
        string url;
        Node*next;
        Node*prev;

        Node(string u){
            url=u;
            next=nullptr;
            prev=nullptr;
        }
};

class BrowserHistory{
    private:
        Node*curr;
    public:
        BrowserHistory(string homepage){
            curr=new Node(homepage);
        }

        void visit(string url){
            Node*newnode=new Node(url);
            curr->next=newnode;
            newnode->prev=curr;
            curr=newnode;
        }

        string back(int steps){
           while(steps>0 && curr->prev!=nullptr){
               curr=curr->prev;
               steps--;
           }
           return curr->url;
        }

        string forward(int steps){
            while(steps>0 && curr->next!=nullptr){
                curr=curr->next;
                steps--;
            }
            return curr->url;
        }
};

int main(void){
    BrowserHistory bh("leetcode.com");
    bh.visit("google.com");
    bh.visit("facebook.com");
    bh.visit("youtube.com");

    cout<<bh.back(1)<<endl;
    cout<<bh.back(1)<<endl;
    cout<<bh.forward(1)<<endl;

    bh.visit("instagram.com");
    cout<<bh.forward(1)<<endl;
    cout<<bh.back(2)<<endl;
    cout<<bh.back(7)<<endl;
    return 0;
}
```

OUTPUT:

```
facebook.com
google.com
facebook.com
instagram.com
google.com
leetcode.com

--------------------------------
Process exited after 8.291 seconds with return value 0
Press any key to continue . . .
```