**Assignment- E12**

**Name of Student: Sumit Bhamare**

**Roll No.:08**

**Problem Statement:**

# A double-ended queue (deque) is a linear list in which additions and deletions may be made at either end. Obtain a data representation mapping a deque into a one- dimensional array. Write C++ program to simulate deque with functions to add and delete elements from either end of the deque.

**Program:**

#include<iostream>

using namespace std;

#define size 5

/\*Data structure for deque\*/

struct queue {

int que[size];

int front, rear;

}Q;

class DeQue

{

public:

DeQue();

int Qfull();

int Qempty();

int insert\_rear(int item);

int delete\_front();

int insert\_front(int item);

int delete\_rear();

void display();

};

DeQue::DeQue()

{

Q.front = -1;

Q.rear = -1;

for (int i = 0; i<size; i++)

Q.que[i] = -1;

}

int DeQue::Qfull()

{

if (Q.rear == size - 1)

return 1;

else

return 0;

}

int DeQue::Qempty()

{

if ((Q.front>Q.rear) || (Q.front == -1 && Q.rear == -1))

return 1;

else

return 0;

}

int DeQue::insert\_rear(int item)

{

if (Q.front == -1 && Q.rear == -1)

Q.front++;

Q.que[++Q.rear] = item;

return Q.rear;

}

int DeQue::delete\_front()

{

int item;

if (Q.front == -1)

Q.front++;

item = Q.que[Q.front];

Q.que[Q.front] = -1;

Q.front++;

return item;

}

int DeQue::insert\_front(int item)

{

int i, j;

if (Q.front == -1)

Q.front++;

i = Q.front - 1;

while (i >= 0)

{

Q.que[i + 1] = Q.que[i];

i--;

}

j = Q.rear;

while (j >= Q.front)

{

Q.que[j + 1] = Q.que[j];

j--;

}

Q.rear++;

Q.que[Q.front] = item;

return Q.front;

}

int DeQue::delete\_rear()

{

int item;

item = Q.que[Q.rear];

Q.que[Q.rear] = -1;/\*logical deletion\*/

Q.rear--;

return item;

}

void DeQue::display()

{

int i;

cout << "\n Straight Queue is: \n";

for (i = Q.front; i <= Q.rear; i++)

cout << " " << Q.que[i];

}

int main()

{

int choice,item;

char ans;

ans = 'y';

DeQue obj;

cout << "\n\t\t Program For Simulation of doubly ended queue using arrays";

do

{

cout << "\n1.Insert by rear\n2.Delete by front\n3.Insert by front\n4.Delete by rear";

cout << "\n5.display\n6.exit";

cout << "\n Enter Your choice: ";

cin>>choice;

switch (choice)

{

case 1:if (obj.Qfull())

cout << "\n Doubly ended Queue is full";

else

{

cout << "\n Enter The item to be inserted: ";

cin>>item;

Q.rear = obj.insert\_rear(item);

}

break;

case 2:if (obj.Qempty())

cout << "\n Doubly ended Queue is Empty";

else

{

item = obj.delete\_front();

cout << "\n The item deleted from queue is: "

<< item;

}

break;

case 3:if (obj.Qfull())

cout << "\n Doubly ended Queue is full";

else

{

cout << "\n Enter The item to be inserted: ";

cin >>item;

Q.front = obj.insert\_front(item);

}

break;

case 4:if (obj.Qempty())

cout << "\n Doubly ended Queue is Empty";

else

{

item = obj.delete\_rear();

cout << "\n The item deleted from queue is: "

<< item;

}

break;

case 5:obj.display();

break;

case 6:exit(0);

}

cout << "\n Do You Want To Continue? (Y/N) : ";

cin >> ans;

} while (ans == 'y' || ans == 'Y');

}

**Output:**

