Switch to other questions

!!!Caution!!!

Copy/Paste is Disabled for Assessments. Do not Try to overcome by Shortcut Keys or Drag/Drop
You will Lose Data

dequeue

Let Q be a Queue data structure and it has ‘n ‘ elements. A ***D_QUE*** is a double ended queue and it allows insertion and deletion at both ends. Assume $X_1, X_2, \ldots, X_n$ are *n*-positive integers. Write an algorithm and subsequent C code to insert element all even numbers at one end of a ***D_QUE*** and all odd integers at another end of a ***D_QUE***.

Input format

Enter the size of queue ‘n’

Insert the ‘n’ elements in the ***D_QUE***

Output format

Print elements from ***D_QUE***

4

14 15 16 17

14 16 17 15

5

1 2 3 4 5

2 4 5 3 1

Font Size 18 | Language | Editor Theme: Select a Theme

Your code has Passed Execution!

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

class DoubleEndedQueue{
public:
stack<int>ans;
int  N;
int arr[100];
int front=-1;
int rear=-1;

bool isEmpty(){
if(front==rear){

return true;
}
else{
return false;
 }
}
```

```
void enque_front(int x){

if((front==0 && rear==N-1) || front==rear+1){
    cout<<"Stack is Full";
}
else if(front==-1 && rear==-1){
    front=0;
    rear=0;

    arr[front]=x;
}
else if(front==0){
    front=N-1;

    arr[front]=x;

}
else{
    front--;

    arr[front]=x;

}
}

void  enque_rear(int x){
if(front==0 && rear==N-1 || front==rear+1){
    cout<<"Stack is Full";
}
else if(front==-1 && rear==-1){
    front=0;
    rear=0;
    arr[rear]=x;
}
else if(rear==N-1){
    rear=0;
    arr[rear]=x;

}
else{
    rear++;
    arr[rear]=x;

}


}

void  deque_front(){


    if(front==-1 && rear==-1){
    cout<<"Empty Queue";
    }
    else if(front==rear){
    ans.push(arr[front]);

    front=-1;
    rear=-1;

    }

    else if(front==N-1){
    ans.push(arr[front]);

    front=0;

    }

    else{
    ans.push(arr[front]);


    front++;
    }
```

```
}
void deque_rear(){
if(front==-1 && rear==-1){
        cout<<"Empty Queue";
    }
    else if(front==rear){

        cout<<arr[rear]<<endl;
        front=-1;
        rear=-1;

    }

    else if(rear==0){


        cout<<arr[rear]<<endl;
        rear=N-1;

    }

    else{

        cout<<arr[rear]<<endl;
        rear--;
    }

}

void print_Front(){
     if(front==-1 && rear==-1){
        cout<<"Empty Queue";
    }
    else{
    cout<<arr[front]<<endl;

    }
    }
void print_Rear(){
     if(front==-1 && rear==-1){
        cout<<"Empty Queue";
    }
    else{
    cout<<arr[rear]<<endl;
    }
}
void printfront(){

    while(!(ans.empty())){
              cout<<ans.top()<<endl;
              ans.pop();

          }
}


};



int main() {
      DoubleEndedQueue obj;
      int n;
      cin>>n;
      obj.N=n;
      int a;
      stack<int>data;
      int even=0;
      int odd=0;

      for(int i=0;i<n;i++ ){
          cin>>a;
          if(a%2==0){
              obj.enque_front(a);
```

```
            even++;

        }
        else{
            obj.enque_rear(a);
            odd++;
        }
    }



        for(int i=0;i<even;i++){
            obj.deque_front();


        }
        obj.printfront();




        for(int i=0;i<odd;i++){
            obj.deque_rear();
        }



        return 0;
}
```

Save Pause Test

Submit Code

Status: