**IMPLEMENT ENQUEUE AND DEQUEUE**

#include <iostream>

#define MAX\_SIZE 100

using namespace std;

class Queue {

private:

int front, rear;

int arr[MAX\_SIZE];

public:

Queue() {

front = -1;

rear = -1;

}

bool isEmpty() {

return (front == -1 && rear == -1);

}

bool isFull() {

return (rear == MAX\_SIZE - 1);

}

void enqueue(int val) {

if (isFull()) {

cout << "Queue is full. Cannot enqueue.\n";

return;

} else if (isEmpty()) {

front = rear = 0;

} else {

rear++;

}

arr[rear] = val;

cout << val << " enqueued to the queue.\n";

}

void dequeue() {

if (isEmpty()) {

cout << "Queue is empty. Cannot dequeue.\n";

return;

} else if (front == rear) {

cout << arr[front] << " dequeued from the queue.\n";

front = rear = -1;

} else {

cout << arr[front] << " dequeued from the queue.\n";

front++;

}

}

int peek() {

if (isEmpty()) {

cout << "Queue is empty. No element to peek.\n";

return -1;

} else {

return arr[front];

}

}

};

int main() {

Queue q;

q.enqueue(10);

q.enqueue(20);

q.enqueue(30);

q.dequeue();

q.enqueue(40);

cout << "Front element: " << q.peek() << endl;

q.dequeue();

q.dequeue();

q.dequeue();

q.dequeue(); // Trying to dequeue from an empty queue

return 0;

}