A purple text on a black background

AI-generated content may be incorrect.

**LAB 11 - Queue with Linkedlist and Array**

Name: Omar Afef Mohammed

Roll number: SU92-BSSEM-S24-086

Section: 3A

Teacher: Mr. Rasikh Ali

Subject: DSA LAB

Date: March 27, 2025

Tasks:

1. With Array; Enqueue, Dequeue, Display

2. With Linkedlist; Enqueue, Dequeue, Display

**ANSWER**

**CODE**

#include <iostream>

using namespace std;

class QueueArray {

private:

int arr[100];

int front, rear;

public:

QueueArray() {

front = -1;

rear = -1;

}

void enqueue(int data) {

if (rear == 99) {

cout << "Queue overflow" << endl;

return;

}

if (front == -1) {

front = 0;

}

rear++;

arr[rear] = data;

}

void dequeue() {

if (front == -1 || front > rear) {

cout << "Queue underflow" << endl;

return;

}

front++;

if (front > rear) {

front = rear = -1;

}

}

void display() {

if (front == -1) {

cout << "Queue is empty" << endl;

return;

}

cout << "Queue elements: ";

for (int i = front; i <= rear; i++) {

cout << arr[i] << " ";

}

cout << endl;

}

};

class Node {

public:

int data;

Node\* next;

Node(int value) {

data = value;

next = NULL;

}

};

class QueueLinkedList {

private:

Node\* front;

Node\* rear;

public:

QueueLinkedList() {

front = NULL;

rear = NULL;

}

void enqueue(int data) {

Node\* newNode = new Node(data);

if (rear == NULL) {

front = rear = newNode;

return;

}

rear->next = newNode;

rear = newNode;

}

void dequeue() {

if (front == NULL) {

cout << "Queue empty" << endl;

return;

}

Node\* temp = front;

front = front->next;

if (front == NULL) {

rear = NULL;

}

delete temp;

}

void display() {

if (front == NULL) {

cout << "Queue is empty" << endl;

return;

}

cout << "Queue elements: ";

Node\* temp = front;

while (temp != NULL) {

cout << temp->data << " ";

temp = temp->next;

}

cout << endl;

}

};

int main() {

cout << "Queue using Array:" << endl;

QueueArray qArray;

qArray.enqueue(10);

qArray.enqueue(20);

qArray.enqueue(30);

qArray.display();

qArray.dequeue();

qArray.display();

cout << "\nQueue using Linked List:" << endl;

QueueLinkedList qList;

qList.enqueue(10);

qList.enqueue(20);

qList.enqueue(30);

qList.display();

qList.dequeue();

qList.display();

return 0;

}

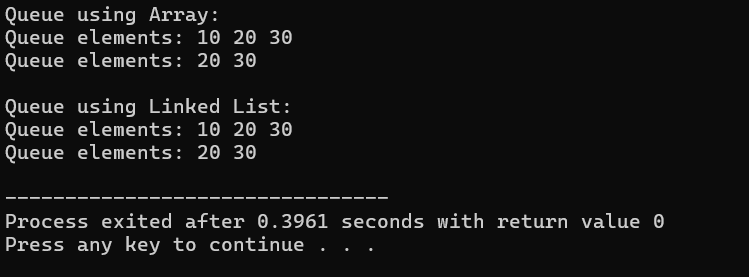
**How it works:**

1. **Array queue: Uses array with position marker**
2. **Enqueue: Adds element at top position**
3. **Dequeue: Removes top element**
4. **Linked queue: Uses connected nodes**
5. **Enqueue: Adds new node at head**
6. **Dequeue: Removes head node**

**Why it works:**

1. **Array: Fast direct access, simple**
2. **Linked list: Dynamic size, no overflow**
3. **Both follow FIFO principle**

**OUTPUT**

****