# **Queue using array:**

#include <iostream>

class Queue {

private:

int maxSize;

int front;

int rear;

int\* array;

public:

Queue(int size) {

maxSize = size;

front = -1;

rear = -1;

array = new int[maxSize];

}

~Queue() {

delete[] array;

}

bool isEmpty() {

return front == -1;

}

bool isFull() {

return (rear + 1) % maxSize == front;

}

void enqueue(int value) {

if (isFull()) {

std::cout << "Queue is full. Cannot enqueue." << std::endl;

} else {

if (isEmpty()) {

front = rear = 0;

} else {

rear = (rear + 1) % maxSize;

}

array[rear] = value;

}

}

void dequeue() {

if (isEmpty()) {

std::cout << "Queue is empty. Cannot dequeue." << std::endl;

} else {

if (front == rear) {

front = rear = -1;

} else {

front = (front + 1) % maxSize;

}

}

}

int frontValue() {

if (isEmpty()) {

std::cout << "Queue is empty." << std::endl;

return -1; // Return a sentinel value to indicate an error

} else {

return array[front];

}

}

};

int main() {

Queue queue(5);

queue.enqueue(1);

queue.enqueue(2);

queue.enqueue(3);

queue.enqueue(4);

std::cout << "Front of the queue: " << queue.frontValue() << std::endl;

queue.dequeue();

queue.dequeue();

std::cout << "Front of the queue: " << queue.frontValue() << std::endl;

return 0;

}