2) IMPLEMENTING QUEUES FROM THE SCRATCH

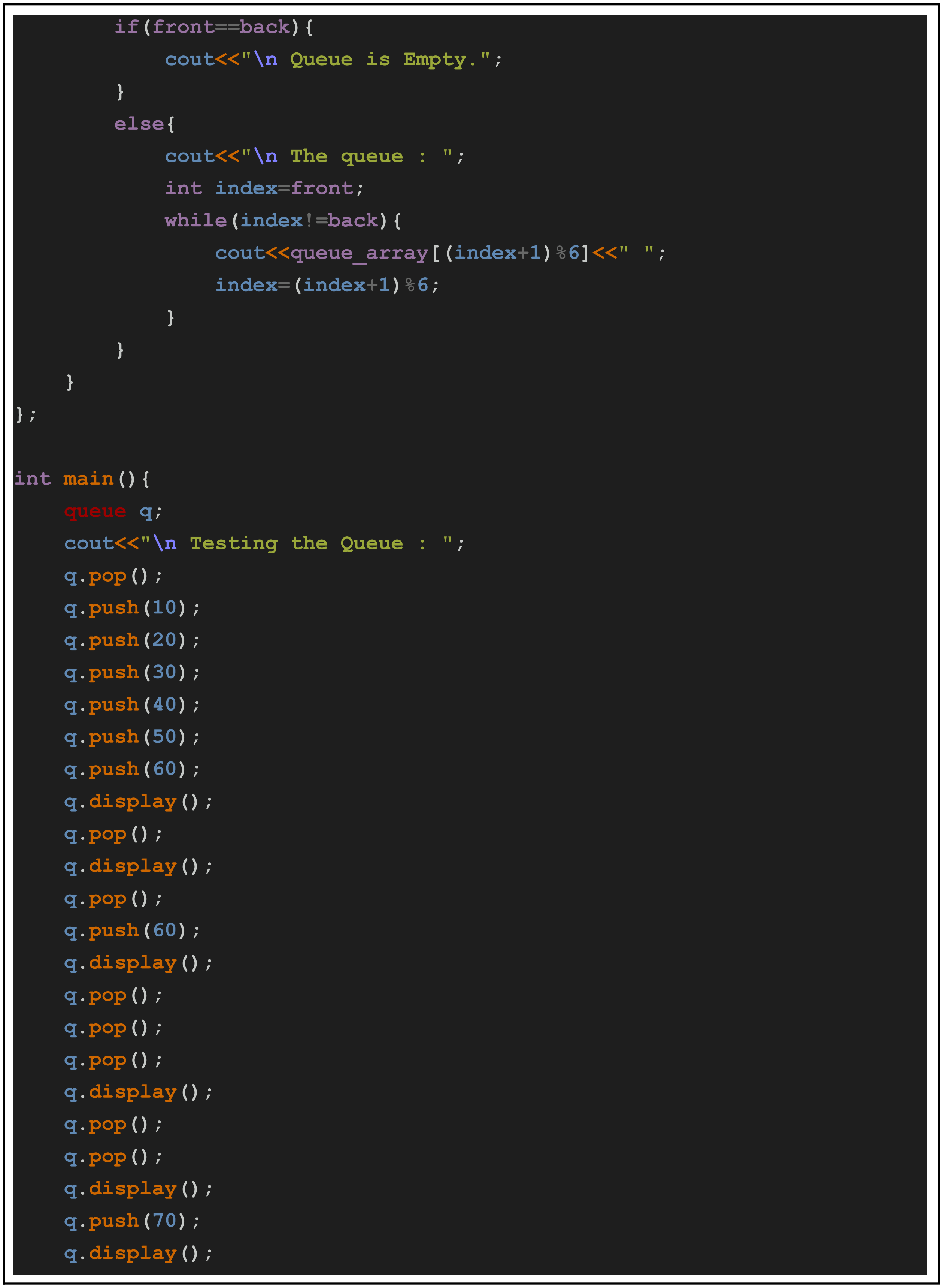
This is the standard sum of the queue.

USING ARRAYS:

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

class queue{
    private:
    int queue_array[6];
    int queue_size=0;
    int front=0;
    int back=0;

    public:
    void push(int n){
        int index=(back+1)%6;
        if(index==front){
            cout<<"\n The queue space is exhausted (queue overflow).";
        }
        else{
            queue_size++;
            queue_array[index]=n;
            back=index;
        }
    }
    void pop(){
        if(front==back){
            cout<<"\n The queue is empty (no elements to pop).";
        }
        else{
            front=(front+1)%6;
            queue_size--;
            cout<<"\n The popped element : "<<queue_array[front];
        }
    }
    void display(){
```



```cpp
        if(front==back){
            cout<<"\n Queue is Empty.";
        }
        else{
            cout<<"\n The queue : ";
            int index=front;
            while(index!=back){
                cout<<queue_array[(index+1)%6]<<" ";
                index=(index+1)%6;
            }
        }
    }
};

int main(){
    queue q;
    cout<<"\n Testing the Queue : ";
    q.pop();
    q.push(10);
    q.push(20);
    q.push(30);
    q.push(40);
    q.push(50);
    q.push(60);
    q.display();
    q.pop();
    q.display();
    q.pop();
    q.push(60);
    q.display();
    q.pop();
    q.pop();
    q.pop();
    q.display();
    q.pop();
    q.pop();
    q.display();
    q.push(70);
    q.display();
    q.push(80);
    q.push(90);
    q.push(100);
    q.push(110);
    q.display();
    q.push(120);
    return 0;
}
```

USING LINKED LIST:

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

class node{
    public:
    int data;
    node *next=NULL;
};

class queue{
    private:
    int queue_size();
    node *front=NULL;
    node *back=NULL;

    public:
    void push(int n){
        node *t=new node;
        if(t==NULL){
            cout<<"\n The heap is exhausted (no further elements can be
added into the queue.";
        }
        else{
            t->data=n;
            if(front==NULL){
                front={back=t};
            }
            else{
                back->next=t;
                back=t;
            }
        }
    }
    void pop(){
        node *t=front;
        if(front==NULL){
            cout<<"\n The queue is Empty (no elements can be popped).";
        }
        else if(front==back){
            front={back=NULL};
            delete(t);
        }
        else{
            front=front->next;
            delete(t);
        }
    }
    void display(){
        node *traverse=front;
        if(traverse==NULL){
            cout<<"\n The Queue is Empty.";
        }
        else{
            cout<<"\n The queue : ";
            while(traverse!=NULL){
                cout<<traverse->data<<" ";
                traverse=traverse->next;
            }
        }
    }
};

int main(){
    queue q;
    cout<<"\n Testing the queue class : ";
    q.pop();
    q.push(10);
    q.push(20);
    q.push(30);
    q.push(40);
    q.display();
    q.pop();
    q.display();
    q.pop();
    q.pop();
    q.pop();
    q.display();
    q.pop();
    q.push(10);
    q.display();
    return 0;
}
```