The Queue

A Stack is a Last-in-First-out (LIFO) data structure since the last element that goes on the stack is the first one that gets popped. A Queue, on the other hand, is a First-in-First-Out data structure (FIFO) where the item that was put in first gets out first – in other words the item that has been waiting longest gets attention first. A simple way to implement a queue is to use an array, much as we did with a stack.

```
#include<iostream>
using namespace std;
const int MAX_CAPACITY=128;
typedef int QueueObject;

class Queue{
    private:
        QueueObject myQ[MAX_CAPACITY];
        int myFront,myBack;
    public:
        Queue(){myFront=0;myBack=0;} //Constructor
        bool empty()const;
        bool full() const;
        void addQ(const QueueObject&); //
        void removeQ(); //
        void display() const;
    }

    bool Queue::empty() const{
        return (myBack==myFront);
    }

    bool Queue::full() const{
        return ((myBack+1)%MAX_CAPACITY==myFront);
    }

    void Queue::addQ(const QueueObject& qo){
        if (full())
```
cerr<<"Queue Full\n";
}
else{
    myQ[myBack]=qo;
    myBack=(myBack+1)%MAX_CAPACITY;
}
}

void Queue::removeQ(){
    if( !empty()){
        myFront=(myFront+1)%MAX_CAPACITY;
    } else{
        cerr<<"Empty queue\n";
    }
}

void Queue::display() const{
    cout<<"Queue contents:\n";
    int beginNum=myFront;
    while((beginNum)%MAX_CAPACITY != myBack){
        cout<<myQ[beginNum]<<endl;
        beginNum++;
    }
}

int main(){
    Queue q1;
    q1.addQ(10);
    q1.addQ(12);
    q1.addQ(91);
    q1.addQ(92);
    q1.display();
    cout<<"Removing two items from the queue\n";
    q1.removeQ();
    q1.removeQ();
    q1.display();
}