Exercise - Queue with Linked List

1. Implement a queue using a linked list. You may use the following as a template.

```
#include<iostream>
using namespace std;
class A{
public:
  int x;
  char c;
  A* next;
  A(int i=0, char y='a'){  //Question: Is the default constructor necessary here?
    x=i;
    c=y;
    next=NULL;
  }
  void display(){
    cout<<"x = "<<x<<"; c = "<<c<<endl;
  }
};
const int MAX_CAPACITY=128;
typedef A QueueObject;
class Queue{
//necessary stuff goes here
};
int main(){
 A a1(2,'a'),a2(4,'A'),a3(6,'C'),a4(8,'J'),a5(5,'k');
 Queue* q=new Queue;
 q->push(&a2);
 q->display();
 q->push(&a1);
 q->push(&a5);
 q->push(&a3);
 q->push(&a4);
 q->display();
 cout<<"popping the queue\n";
 q->pop();
 q->display();
}
/***Output*****************

---------Size = 1---------
x = 4; c = A
---------Size = 5---------
x = 4; c = A
x = 2; c = a
x = 5; c = k
x = 6; c = C
x = 8; c = J
popping the queue
---------Size = 4---------
x = 2; c = a
x = 5; c = k
x = 6; c = C
x = 8; c = J
**************************/
```

2.In the previous implementation, we separated the Queue Nodes (QueueObject, A) from the Queue class itself. This is similar to the separation of Linked List nodes from the List itself. Essentially, this achieves separation of data from the logic of the data structure. It is possible to put data into the queue class. Use the following as a template to create a queue.

```
#include<iostream>
#include<string>
using namespace std;
const int MAX_CAPACITY=128;
class Queue{
public:
  char c;
  string s;
  Queue* front;
  Queue* back;
  Queue* next;
  int size;
  Queue(char w='X', string y=" "){
    s=y; c=w;
    front=NULL;
    back=NULL;
    next=NULL;
    size=0;
  }
  void addQ(Queue* a){            //Add to the queue
      //Your code goes here
  }
  Queue* removeQ(){             //pop the queue
      //Your code goes here
  }
  void display() const { //display the queue
      //your code goes here
  }
};
int main(){
  Queue* q1=new Queue('A', "hello");
  Queue* q2=new Queue('B', "how");
  Queue* q3=new Queue('C', "are");
  Queue* q4=new Queue('D', "you?");
  Queue* q=new Queue;
  q->addQ(q1);
  q->display();
  q->addQ(q2);
  q->addQ(q3);
  q->addQ(q4);
  q->display();
  cout<<"popping the queue\n";
  q->removeQ();
  q->display();
}
/****output***********
---------Size = 1---------
string = hello; char = A
---------Size = 4---------
string = hello; char = A
string = how; char = B
string = are; char = C
string = you?; char = D
```

```
popping the queue
---------Size = 3---------
string = how; char = B
string = are; char = C
string = you?; char = D
**********************/
```

3. In the above program, we are not getting hold of the item removed from the queue. Modify the program so that you can capture the removed item and display it.