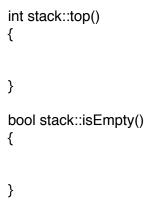
Stack and Queue

```
Problem 1: Implement a Stack using Singly Linked List.
class Stack
{
public:
       Stack();
       ~Stack();
       void push();
       void pop();
       int top();
       bool isEmpty();
private:
       struct Node
       {
              int value;
              Node* node;
       Node* head;
};
Stack::Stack()
Stack::~Stack()
       while(head != nullptr)
       {
       }
}
void Stack::push(int value)
}
void Stack::pop()
}
```



Problem 2: Design a stack which can also return the minimum element in the stack. You can use the stack library in C++.

Problem 3: Implement a queue using two stacks.

Inheritance

```
Problem 1: What is the output for the following program?
#include <iostream>
using namespace std;
class A
public:
       void print(){cout << "A::print()"<<endl;}</pre>
};
class B: public A
{};
int main()
       Aa;
       a.print();
       Bb;
       b.print();
}
Problem 2: What is the output for the following program?
#include <iostream>
using namespace std;
class A
{
public:
       void print(){cout << "A::print()"<<endl;}</pre>
};
class B: public A
public:
       void print(){cout << "B::print()"<<endl;}</pre>
};
int main()
       Aa;
       a.print();
       Bb;
       b.print();
       A* c = new B;//Polymorphism
       c->print();
}
```

Problem 3: What if we add virtual keyword to print() in class A?