OOP Sections
Lab # 2

EX2: Integer stack

Implement a stack data structure using a class. A stack is called a LIFO (last in first out) structure because the last value placed in the stack is the first one to be taken out, Similar to a stack of plates. One always removes the plate at the top rather than one from the middle or bottom.
Stacks are used in great many computer applications and are discussed in nearly every intermediate level programming textbox.
Here is the class definition:

```
Class Stack{
Public:
Stack();  // default constructor
int Empty() const;  // return 1 if empty 0 if not.
int Full() const;  // return 1 if full 0 if not.
void push(int item);
int pop();

Private:
Stacksize = 40;
int top;
int data [stacksize];
};
```

Solution:

Main.cpp

```
#include <iostream>
#include "Stack.h"
using namespace std;
int main()
{
   Stack s;
   s.push(1);
   s.push(2);
   s.push(3);
   int x=s.pop();
   cout<< x << endl;
   x=s.pop();
   cout<< x << endl;
   return 0;
}
```

Stack.h

```
#ifndef STACK_H
#define STACK_H


class Stack
{

public:
   Stack(); // Default constructor
   int empty() const; // Return 1 if the stack is empty, 0 if not
   int full() const; // Return 1 if the stack is full, 0 if not
   void push(int item); // Push a new value onto the stack
   int pop(); // Pop a value from the stack



private:
   enum {StackSize = 40};
   int top;
   int data[StackSize];
};

#endif // STACK_H
```

Stack.cpp

```
#include "Stack.h"
#include <iostream>

using namespace std;
Stack::Stack()
{
   top=0;
}
int Stack::empty() const
{
   if(top==0)
      return 1;
   else
      return 0;
}
int Stack::full() const
{
   if(top==StackSize)
      return 1;
   else
      return 0;
}
void Stack::push(int item)
{
   if(!full())
      data[top++]=item;
   else
      cout<<"The stack is full";
}
int Stack::pop()
{
   if(!empty())
   {
      return data[--top];
   }
   else
      cout<<"The stack is empty";
   return -1;
}
```