CS213M: Assignment 2

Problem 1: Implementation of a Stack

Due Date: 04/02/2015

We are providing you with a file **`stack.hpp`** with declarations for functions you have to define in a file to be named **`stack.cpp`**. The functions you need to define are reproduced below for quick reference. As you can see, this time around we want to define a template class.

Note: Do not modify the given code. Do not change the signatures of the functions below. Do not use C++ STL stacks for this problem..

Functions to be defined

1. `stack::stack();`

   This is a constructor for your class. Initialise your member variables here if applicable.

2. `stack::stack(const stack &to_copy_to);`

   This is a copy constructor for your class. It is called when an object (call this object `A`) of your `stack` class is assigned to another `stack` object (call this object `B`) i.e. if you write `B = A`. In that case, the copy constructor of `A` will get called with `B` as the argument.

3. `stack::~stack();`

   This is a destructor for your class, called when an object of stack type is destroyed. Implement it if you have some cleaning up to do (freeing up heap memory and the like).

4. `void stack<T>::push(T obj);`

   Push the object **`obj`** on the top of the stack.

5. `int stack<T>::top(T *top_element);`

   This function sets the value of the location pointed by **`top_element`** to the object at the top on the stack. It returns a positive quantity on successful execution. If the stack is empty, it returns a negative quantity.

6. `void stack<T>::pop();`

   This function removes the object at the top of the stack. It does nothing if called on an empty stack.

7. `int stack<T>::size();`

   This functions returns the number of elements in the stack.

Note: In the above, `T` is the template type.

Don't forget to read the comments in the given header file.