CS 240 Homework 2

Alex Vondrak

Fall 2011

1. What do the initially empty stacks `stack1` and `stack2` look like after each of the following operations, and what are the outputs (if any) of said operations? (If the operation triggers an error, just write "error".)

Operation	**Output**	**`stack1`'s contents (bottom, . . . , top)**	**`stack2`'s contents (bottom, . . . , top)**
`stack1.push(3)`			
`stack2.push(2)`			
`stack2.push(4)`			
`stack2.push(4)`			
`stack1.pop()`			
`stack2.push(stack1.pop())`			
`stack1.push(stack2.top())`			
`stack1.push(6)`			
`stack1.pop()`			
`stack1.top()`			
`stack1.isEmpty()`			
`stack2.size()`			
`stack1.pop()`			
`stack1.pop()`			

2. Show the contents of the stack for each step of evaluating the following postfix expressions.

   (a) `3 1 + 4 / 1 *`

   (b) `5 9 + 2 - 6 /`

   (c) `5 3 5 + - 8 *`

   (d) `9 7 * 9 - 3 /`

3. Show the process of converting the following infix expressions to postfix.

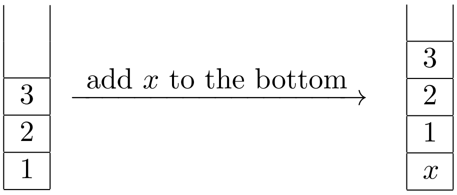
   (a) `1 * 2 + 3 - 4`

   (b) `(((8 + 6) / 7) - 5)`

   (c) `((3 - 0) * (9 - (3 + 1)))`

   (d) `(((4 - 1) * 5) + 9) / 2)`

4. Given a stack `s1`, `s1.push(x)` inserts `x` at the top of the stack. Write code that shows how we can use an auxiliary stack `s2` to insert `x` at the *bottom* of `s1`. E.g.,

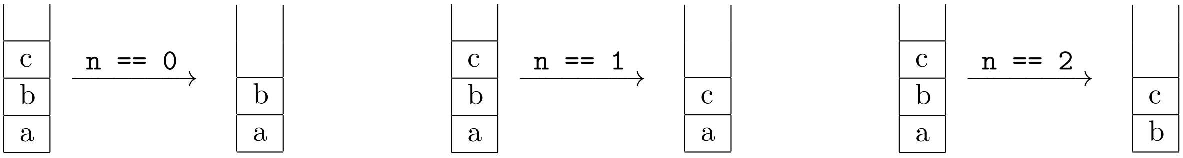


```
Stack s1 = /* any stack */;
int x = /* any int */;

Stack s2 = new Stack(); // empty

/* Implement me */
```

5. Given a stack `s1`, `s1.pop()` removes the top element of the stack. Write code that shows how we can use an auxiliary stack `s2` to remove an arbitrary item `n` elements from the top of `s1`. E.g.,



```
Stack s1 = /* any stack */;
int n = /* n elements from the top of s1 */;

Stack s2 = new Stack(); // empty

/* Implement me */
```