Name: _____

USC loginid (e.g., ttrojan): _____

CS 455 Final Exam
Spring 2012 [Bono]

May 8, 2012

There are 3 problems on the exam, with 60 points total available. There are 5 pages to the exam, including this one; make sure you have all of them. There is also a separate one-page code handout. If you need additional space to write any answers, you may use the backs of exam pages (just direct us to look there).

Note: if you give multiple answers for a problem, we will only grade the first one. Avoid this issue by labeling and **circling** your final answers and crossing out any other answers you changed your mind about (though it's fine if you show your work).

Put your name and USC loginid at the top of the exam. Please read over the whole test before beginning. Good luck!

	value	score
Problem 1	22 pts.	
Problem 2A	15 pts.	
Problem 2B	3 pts.	
Problem 3	20 pts.	
TOTAL	60 pts.	

Problem 1 [22 pts. total]

Part A (20) Implement the boolean **Java** method `isPatternSequence`, which tells if its array argument is an example of the sequence [1, 2, 2, 3, 3, 3, 4, 4, 4, 4, …] ending with *k* copies of the value *k*, for some *k* (in the example shown here, *k* would have to be > 4). The array must have at least one element.

Some examples:

`arr`	return value of `isPatternSequence(arr)`
[1]	*true*
[1, 2, 2]	*true*
[1, 2, 2, 3, 3, 3]	*true*
[1, 2, 2, 3, 3]	*false*
[1, 2, 2, 3, 3, 3, 3]	*false*
[1, 2, 2, 3, 3, 3, 4, 3, 4, 4]	*false*
[4, 1, 7]	*false*

```
// PRE: arr.length > 0
public boolean isPatternSequence(int[] arr) {
```

Part B (2). Using big-O notation give the worst-case time for your method as a function of the size of the array.

Problem 2 [18 pts. total]

Part A (15). Write the complete class definition and implementation (i.e., method definitions) for a **C++** class for a `Stack` of `int`s that uses a linked list representation. All stack operations must run in constant time and have no memory leaks. You may assume our usual `Node` struct and `ListType` are already defined (see code handout for definitions). You may not use STL container classes in your implementation. You do not have to write method comments for your class.

This is a C++ question, since we are using linked lists here. However, the question is also to test your knowledge and skills of issues we have been dealing with from even before we started with C++: the Stack abstract data type, defining a class and its methods using object-oriented principles, using a data structure to solve a problem efficiently, and representation invariants (Part B).

Because you are less familiar with the syntax for C++ classes we have given you a complete example of a C++ class definition with associated methods (`Student` class) on the code handout.

The interface for the `Stack` class is shown by example below:

```
Stack s;             // creates an empty stack

s.push(3);           // pushes 3 onto the stack

s.push(4);           // pushes 4 onto the stack

s.pop();         // pops top element from the stack (Note: does not return a value)
                 // only legal on non-empty stack

cout << t.top();     // returns the top element (stack is unchanged)
                     // only legal on non-empty stack

if (s.isEmpty()) {  // returns true if stack is empty, otherwise false

  cout << "stack is empty." << endl;

}
```

Space for your answer is given on this and the next page.

Problem 2 (cont.)

(additional space for your answer to Part A; Part B)

Part B (3). Write a representation invariant (a.k.a., implementation invariant) for your class implementation:

Problem 3 [20 pts.]

Implement the **C++** function `remove7s` which removes all the 7s from a linked list of `int`s. The `Node` and `ListType` definitions are on the code handout.

`list` before call	**`list`** after call to **`remove7s(list)`**
(0 1 2)	*(0 1 2)*
(7 6 7 5 7 7)	*(6 5)*
(7 7 7)	*()*
()	*()*
(3)	*(3)*

```
// PRE: list is a well-formed linked list
void remove7s(ListType & list) {
```