CPSC 331 — Practice Midterm Test

Name: _____

Please **DO NOT** write your ID number on this page.

Instructions:

1. Answer all questions in the space provided. Continue answers on the final page if you need more room.
2. Point form answers are acceptable if complete enough to be understood.
3. One double-sided letter-sized page of notes is allowed. No other aids are allowed.
4. This test is out of 45.

Duration: 90 minutes

ID Number: _____

Question	Score	Available
1		10
2		5
3		15
4		10
5		5
Total:		45

ID Number: _____

1. Consider the "Slytherin Number Computation" problem and the recursive `slytherin` algorithm shown on the first page of the supplement for this test.

(3 marks)

(a) Prove that the function

$$f(\mathtt{n}) = \mathtt{n}$$

is a ***bound function*** for this recursive algorithm.

Proof That This is a Bound Function:

(7 marks)

(b) Prove the following claim (on this and the next page).

> If the `slytherin` algorithm is executed with a nonnegative integer n as input, then this execution of the algorithm eventually halts, returning the n^{th} Slytherin number, S_n, as output.

Note: Since this algorithm does not modify its input, access or modify global data, or generate other output, this is sufficient to prove that this algorithm correctly solves the "Slytherin Number Computation" problem.

Proof of the Claim:

ID Number: _____

ID Number: _____

(5 marks)

2. Write a ***recurrence*** for the number $T_{\texttt{slytherin}}(\texttt{n})$ of steps that are executed by the algorithm `slytherin` when this algorithm is executed with a nonnegative integer `n` as input.

   Your Recurrence:

ID Number: _____

3. Now consider the `cSlytherin` algorithm, for the "Slytherin Number Computation" problem, that is shown on the *second* page of the supplement for this test.

(2 marks)

(a) Note that `i` $= 1$, `hocus` $= 0 = S_0$, and `pocus` $= 1 = S_1$ before execution of the `while` loop begins (during an execution of the algorithm with a nonnegative integer `n` ≥ 2 as input).

Suppose j is a positive integer such that the body of the `while` loop is executed at least j times.

Describe how the values of `hocus` and `pocus` are related to the value of `i` at the *beginning* and `end` of the j^{th} execution of the body of the `while` loop. (*Hint:* One of them is $S_{\texttt{i}}$.)

Relationships between the Values of These Variables:

(4 marks)

(b) Give a ***loop invariant*** for the `while` loop in this algorithm. Your loop invariant should be both correct and complete enough to establish the partial correctness of this algorithm.

Note: Your answer for part (a) might be useful here.

Loop Invariant:

ID Number: _____

(6 marks)

(c) Use a theorem, given in class, to prove that your loop invariant is correct.

Proof That Your Loop Invariant is Correct:

ID Number: _____

(3 marks)

(d) Assuming that that your assertion for the end of the loop is correct, give a **brief** proof of the partial correctness of this algorithm.

Note: You may assume (or observe, by inspection of the code) that an execution of this algorithm has no undesired side-effects.

Brief Proof of Partial Correctness:

ID Number: _____

4. Once again, consider the `cSlytherin` algorithm on the second page of the supplement for this test.

(5 marks)

(a) Give a **bound function** for the `while` loop in this algorithm and give a **brief** proof that your answer is correct.

Bound Function for Loop:

Brief Proof That This is a Bound Function:

ID Number: _____

(4 marks)

(b) Use this to write an **upper bound** for the number of steps that are executed during an execution of the `while` loop in this algorithm, using the *uniform cost criterion* to define this. This should be a function of the input `n`.

If you gave this is a *summation* then you should also give an upper bound that *does not* include a summation.

Upper Bound for Worst-Case Running Time of Loop:

(1 mark)

(c) Finally, use this to give an **upper bound** for the number of steps executed during an execution of this algorithm, using the *uniform cost criterion* to define this. This should also be a function of the input `n`.

ID Number: _____

5. Consider the **bounded stack** abstract data type.

(3 marks)

(a) Describe how an **array** (or `ArrayList`) `A` can be used to implement a **bounded stack**. Where should the value at the **bottom** of the stack be stored in the array and (if this stack has size n and the array has length $m \geq n$)? Where should the value at the **top** of the stack be stored, if stack operations are to be implemented efficiently? Where should the *other* values on the stack be stored?

Location of Bottom of Stack:

Location of Top of Stack:

Location of Other Values on the Stack:

(2 marks)

(b) Give (very short) pseudocode or a description in English for a method to remove (or "**pop**") value `x` from a stack when this array-based implementation is being used — assuming that the `size` n of the stack is also being stored.

Method To Pop a Value Off of the Stack:

ID Number: _____