CPSC 331 — Solution for Question #1 on the Practice Midterm Test

(a) You were first asked to prove that the function

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

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

Proof: Since $\mathtt{n}$ is an input for this algorithm, this is a function of the function's input(s). It remains only to show that this function satisfies all three properties of functions that are included in the definition of a "bound function for a recursive algorithm."

- Since $\mathtt{n}$ is an integer input, this is certainly an ***integer-valued*** function.
- The `slytherin` algorithm only calls itself recursively (twice) during the execution of the step at line 5. One can see at the input $\mathtt{n}$ is replaced by either $\mathtt{n} - 1$ or $\mathtt{n} - 2$ when the algorithm calls itself. Thus (since f is the identity function) the value of this function is ***decreased in value by at least one*** every times the algorithm calls itself recursively.
- Suppose that the precondition for the "Slytherin Number Computation" problem is satisfied, and that the value of this function is less than or equal to zero when the `slytherin` algorithm is called. Then $\mathtt{n} \geq 0$, since the precondition for the problem is satisfied, but $\mathtt{n} \leq 0$ since $f(\mathtt{n}) = \mathtt{n}$. Thus $\mathtt{n} = 0$.

  In this case the execution of the algorithm halts after the test at line 1, which is passed, and the execution of the `return` statement at line 2. Thus the algorithm halts without calling itself again recursively in this case, as required.

It follows that the function $f(\mathtt{n}) = \mathtt{n}$ is a bound function for this recursive algorithm, as claimed.

Note: Full marks will be awarded if it is confirmed that the function has the expected type and satisfies all three of the required properties — with at least some details given about the problem to be solved and the recursively algorithm being considered. (Without these details, an answer might be just a restatement of the definition of a "bound function for a recursive algorithm.")

(b) You were next asked to prove the following claim.

> 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.

Proof of This Claim:

The claim will be proved by induction on n, using the strong form of mathematical induction. The cases $n = 0$ and $n = 1$ will be considered in the basis.

Basis:

Case: $n = 0$. Suppose first that the algorithm is executed with the integer $n = 0$ as input. In this case, the test at line 1 is checked and passed, and the value 0 is returned after the execution of the `return` statement at line 2. Since $S_n = S_0 = 0$, this establishes the claim in this case.

Case: $n = 1$. Suppose, next, that the algorithm is executed with the integer $n = 1$ as input. In this case, the test at line 1 is checked and fails, the test at line 3 is checked and passes, and the value 1 is returned as output after the execution of the `return` statement at line 4. Since $S_n = S_1 = 1$, this establishes the claim in this case as well.

Inductive Step: Let k be an integer such that $k \geq 1$. It is necessary and sufficient to use the following

> Inductive Hypothesis: For every integer ℓ such that $0 \leq \ell \leq k$, if the `slytherin` algorithm is executed with the integer ℓ as input, then this execution of the algorithm eventually halts, returning the ℓ^{th} Slytherin number, S_ℓ, as output.

to prove the following

> Inductive Claim: If the `slytherin` algorithm is executed with the integer $k+1$ as input, then this execution of the algorithm eventually halts, returning the $k+1^{\text{st}}$ Slytherin number, S_{k+1}, as output.

With that noted, suppose that the `slytherin` algorithm is executed on the input $k + 1$. Since $k \geq 1$, $k + 1 \geq 2$, so that the tests at lines 1 and 3 are both checked and fail, and the execution of the algorithm continues with the execution of the statement at line 5.

- The first of the recursive applications of the algorithm at line 5 has input $n - 1 = k$. Since $0 \leq k \leq k$ it follows by the inductive hypothesis that this execution of the algorithm eventually halts, with the k^{th} Slytherin number, S_k, returned as output.

- The second of the recursive applications of the algorithm at line 5 has input $n-2 = k-1$. Since $k \geq 1$, $0 \leq k-1 \leq k$, and it follows by the inductive hypothesis that this execution of the algorithm eventually halts, with the $k-1^{\text{st}}$ Slytherin number, S_{k-1}, returned as output.
- It follows that this execution of the algorithm ends with the value

  $$2 \times S_k - S_{k-1}$$

  returned as output. Now, since $k \geq 1$, $k+1 \geq 2$, and it follows by the recursive definition of the Slytherin numbers that

  $$2 \times S_k - S_{k-1} = S_{k+1}.$$

  Thus S_{k+1} is returned as output, as needed to establish the inductive claim.

The claim now follows by induction on n. $\square$