Name: Felipe Lima

ID: 10929055

CSCI 3104, Algorithms **Due January 22, 2021**

Problem Set 1 (50 points) **Spring 2021, CU-Boulder**

Advice 1: For every problem in this class, you must justify your answer: show how you arrived at it and why it is correct. If there are assumptions you need to make along the way, state those clearly.
Advice 2: Verbal reasoning is typically insufficient for full credit. Instead, write a logical argument, in the style of a mathematical proof.

Instructions for submitting your solution:

- The solutions **should be typed** and we cannot accept hand-written solutions. Here's a short intro to Latex.
- You should submit your work through **Gradescope** only.
- The easiest way to access Gradescope is through our Canvas page. There is a Gradescope button in the left menu.
- Gradescope will only accept **.pdf** files.
- It is vital that you match each problem part with your work. Skip to 1:40 to just see the matching info.

Name: Felipe Lima
ID: 10929055

1. (a) Identify and describe the components of a loop invariant proof.
   (b) Identify and describe the components of a mathematical induction proof.

Solution:

(a) .

- **Initialization.** Loop invariant holds true prior to the first iteration of the loop.
- **Maintenance.** If the loop invariant is true before the ith iteration of the loop, it remains true before the iteration $i + 1$.
- **Termination.** When the loop terminates, the invariant gives us a useful property that helps show the correctness of the algorithm.

(b) .

- **Statement of inductive intent.**
- **Base Case.** Shows that the proposition holds for k = 0 (holds for a simple case).
- **Inductive Hypotheses.** Assume that the proposition holds true for a particular value k.
- **Induction Step.** Prove from hyphoteses that if the proposition holds true for k, it is also true for *k + 1*.
- **Conclusion.** State that based on base case and the inductive step, the proposition holds true for all values.

2. Identify the loop invariant for the following algorithms.

(a)
```
function Sum(A)
    answer=0;
    n=length(A);
    for i=1 to n
        answer += A[i]
    end
    return answer
end
```

(b)
```
function Reverse(A)
    n=length(A)
    i=ceiling(n/2)
    j=ceiling(n/2) + (n+1) mod 2
    while i>0 and j<=n
        tmp=A[i]
        A[i]=A[j]
        A[j]=tmp
        i=i-1
        j=j+1
    end
end
```

(c) Assume that **A** is sorted such that the largest value is at **A**[n]. Assume **A** contains the value `target`.

```
function Search(A,target) //returns the index of the value target
    left=1
    right=length(A)
    while left <=right
        m=floor((left+right)/2)
        if A[m] < target
            left=m+1
        else if A[m]>target
            right=m-1
        else
            return m
        end
    end
end
```

Solution:

(a) At the start of each iteration of the for loop, the variable *answer* is equal to the sum of all elements in the subarray $A[1..i-1]$.

(b) At the start of each iteration of the while loop, i is the $ceiling(n/2)-$ the number of iterations.

(c) At the start of each iteration of the while loop, the variable *target* is in the subarray $A[left..right]$.

Name: Felipe Lima

ID: 10929055

CSCI 3104, Algorithms **Due January 22, 2021**

Problem Set 1 (50 points) **Spring 2021, CU-Boulder**

3. Prove the correctness of the following algorithm. (*Hint: You need to prove the correctness of the inner loop before you can prove the correctness of the outer loop.*)

```
function Sort(A)
    n=length(A)
    for i=1 to n
        for j=2 to n
            if A[j]<A[j-1]
                swap(A[j],A[j-1]) //a function that swaps the elements in the array
            end
        end
    end
end
```

Solution:

Inner loop:

- **Loop Invariant:** At the start of each iteration of the inner for loop, the subarray $A[j-1..j]$ is sorted.

Outter loop:

- **Loop Invariant:** At the start of each iteration of the outter for loop, the subarray $A[1..i]$ is sorted.
- **Initialization:** At the start of the first iteration of the outter for loop, where $i = 1$, the subarray $A[1..i]$ is $A[1..1]$, which consists of one single element and is therefore sorted.
- **Maintenance:** At the start of the ith iteration,
- **Termination:** At th

Name: Felipe Lima

ID: 10929055

4. (a) Suppose you have a whole chocolate bar composed of $n \geq 1$ individual pieces. Prove that the minimum number of breaks to divide the chocolate bar into n pieces is $n - 1$.

   (b) Show that for fibonacci numbers $\sum_{i=1}^{n} f_i^2 = f_n f_{n+1}$
   Recall that the fibonacci numbers are defined as
   $f_0 = 0,\ f_1 = 1$
   $\forall n > 1,\ f_n = f_{n-1} + f_{n-2}$

   (c) For which nonnegative integers n is $3n + 2 \leq 2^n$? Prove your answer.

Solution: