CS lab 6 Written Parts

Jason Kim Ghazi Randhawa

October 2019

1 Question 1

Prove each of the following claims by induction.

1.1 Part a

$$\sum_{i=1}^{n} 2i = n^2 + n$$

1.1.1 Solution a

The base case for our situation would be for n=1. Then the summation of the first n even numbers is going to be 2 while the Right hand side is going to be equal to 2 as well. Hence, we can say that the statement is true for the base case.

Now we perform the inductive step. We first see that we can express $n^2 + n$ as $n(n + 1)$. Now for the n+1th even number, the sum should be equal to $(n + 1)^2 + n + 1$ which can be expressed as the expression $(n + 1)(n + 2)$. This follows the pattern suggested by our nth case. Hence, we have proved by induction that for every $n \geq 1$,

$$\sum_{i=1}^{n} 2i = n^2 + n$$

is true.

1.2 Part b

$$\sum_{i=1}^{n} \frac{2}{3^i} = 1 - \frac{1}{3^n}$$

1.2.1 Solution b

The base case for our situation would be for n=1. Then the summation of the first n even numbers is going to be $\frac{2}{3}$ while the Right hand side is going to be equal to $1 - \frac{1}{3} = \frac{2}{3}$ as well. Hence, we can say that the statement is true for the base case.

Now, we perform the inductive step. We add $\frac{2}{3^{n+1}}$ to both left and right hand sides. Then on the right hand side, we can carry out the following procedure:

$$RHS : 1 - \frac{1}{3^n} + \frac{2}{3^{n+1}}$$

$$RHS : 1 - \frac{3}{3^{n+1}} + \frac{2}{3^{n+1}}$$

$$RHS : 1 - \frac{1}{3^{n+1}}$$

Hence, we can see that for every $n \geq 1$,

$$\sum_{i=1}^{n} \frac{2}{3^i} = 1 - \frac{1}{3^n}$$

.

1.3 Part c

Base case for $n = 1$:
P(1)=$5^1 - 1 = 4 * 1$.
Inductive case: P(n) $\Rightarrow P$(n+1)

$5^(n + 1) - 1 = 5(5^n) - 1$
$= 5(5^n - 1 + 1) - 1$
$= 5(5^n - 1) + 5 - 1$
$= 5(5^n - 1) + 4$

Since we know that $5^n - 1$ is divisible by 4 (by the inductive step) and 4 is obviously divisible by 4, we know that $5^(n + 1) - 1$ is divisible by 4. Thus for every $n \geq 1$, $5^n - 1$ is divisible by 4.

2 Question 2

The function minPos, given below in pseudocode, takes as input an array A of size n of numbers. It returns the smallest positive number in the array. If no positive numbers appear in the array, it returns positive infinity (+). (Note that zero is neither positive nor negative). Using induction, prove that the minPos function works correctly. Clearly state your recursive invariant at the beginning of your proof.

Function minPos(A,n)
If n = 0 Then
Return +
Else
best ¡- minPos(A,n-1)
If A[n-1] ¡ best And A[n-1] ¿ 0 Then
best ¡- A[n-1]
EndIf
Return best
EndIf
EndFunction

2.1 Solution

For p(n) ($n > 0$), the invariant is that P(n) will return the smallest positive number in the array of n values.
Let's consider the base case where n=0. Our algorithm will go into the first condition and return positive infinity. This is something that we should expect since when n=0, there is no positive number in the array.
Now, we consider the inductive step. We consider an array of size n and we assume that the algorithm works perfectly on this array and returns a number X. This X will either be positive infinity or a positive number. Then we increase our array size by 1 to n+1. Once we run the algorithm on this new array, we go into the else condition and run the line '$best < -minPos(A, (n+1)-1)$'. This means we are calling minPos on the original array of size n(assuming that the newest element is added to the end of the array A) and we know that it will return a number X. Now, we have three cases:

- Our newest element is positive and smaller than X
  In this case, we go into the condition 'If $A[n-1] < best$ And $A[n-1] > 0$' and that updates our best variable from X to our new element at $A[n-1]$, which is then returned

- Our newest element is positive and larger than X
  In this case, X remains the smallest positive number in our array, we do not go into the 'If $A[n-1] < best$ And $A[n-1] > 0$' condition, and the minPos returns X, which is what we wanted it to be.

- Our newest element is negative:

In this case, X remains the smallest positive number in our array, we do not go into the 'If $A[n-1] < best$ And $A[n-1] > 0$' condition, and the minPos returns X, which is what we wanted it to be.

We have proved that the algorithm works correctly through induction. QED