Adam Gincel

Professor Engling

CS 135-A

10 March 2015

*I pledge my honor that I have abided by the Stevens Honor System.*

Homework 5

5.1

* 1. P(1) = 1^2
  2. P(1) = 1^2 = 1(1 + 1)(1 + 2)/6 = 6/6 = 1
  3. P(n+1) = true for all n
  4. We need to prove that P(n+1) will be valid, in which case P(n) will imply P(n+1), meaning P is true for all integers n. Because we have a basis step, we assume P(n) is true. We then prove that P(n + 1) is true using P(n) as our assumption. If we prove P(n+1) is true, we prove P for all n is true.
     1. P(n) = n(n + 1)(2n + 1)/6 = (n^2 + n)(2n + 1)/6 = (2n^3 + 3n^2 + n)/6
     2. P(n + 1) = 1^2 + 2^2 + … + n^2 = (n + 1) (n + 2) (2n + 3)/6 = (n^2 + 3n + 2)(2n + 3)/6 = (2n^3 + 3n^2 + 6n^2 + 9n + 4n + 6)/6 = (2n^3 + 9n^2 + 13n + 6)/6 = (2n^3 + 3n^2 + n)/6 + (6n^2 + 12n + 6)/6 = (2n^3 + 3n^2 + n)/6 + (n^2 + 2n + 1) = (2n^3 + 3n^2 + n)/6 + (n + 1)^2. This is equal to P(n) + (n + 1)^2, which proves that P(n + 1) is true if P(n) is true.
  5. Because P(1) is true, we know that a value P(n) can be true. By showing that P(n+1) is equal to P(n) + (n+1)^2, making it the correct next step in the series, we have proven that P(n) is true for all positive integers, starting at n = 1.
  6. Basis step: P(1) = 1^2 + 1 = 2, 2 is even therefore P(1) is true.
  7. P(n) is P(n) = n^2 + n, which we assume to be true.
  8. P(n + 1) = n^2 + 2n +1 + n + 1 = (n^2 + n) + (2n + 2) = (n^2 + n) + 2(n + 1)
     1. Because we know n^2 + n is even, because any quantity divided by 2 has to be even (ie 2(n + 1) is even), and because two even numbers added together result in an even number, P(n + 1) = (n^2 + n) + 2(n + 1) is divisible by two, making P(n) true for all positive n.

5.2

1. 3, 6, 9, 10, 12, 13, 15, 16, 18, 19, 20, 21, 22, 23, 24, 25, etc…
   1. Hypothesis: any number ending in a 3, 6, 9, any multiple of 3 or 10, and any number greater than or equal to 18 will be possible to make using only 3 and 10 cent stamps.
   2. Inductive hypothesis: Assuming a number has been formed using only 3 and 10 cent stamps, there are ways to add to our current number to get the set shown above.
      1. Our basis step: 18 cents is formed with 6 3 cent stamps, making it true.
      2. There are 3 ways to add to our current value to get the set of numbers listed above:
         1. When we have zero or one 10 cent stamp(s), we remote three 3 cent stamps and replace them with a 10 cent stamp, adding 1.
         2. When we have 2 10 cent stamps, you remove those 10 cent stamps and add seven 3 cent stamps, adding 1.
         3. These do not work before n = 18 because there are not enough threes to replace until then.
   3. Strong induction
      1. Basis cases:
         1. P(18) = six 3 cent stamps.
         2. P(19) = one 10 cent stamp and three 3 cent stamps (true)
         3. P(20) = two 10 cent stamps
      2. Inductive hypothesis: some number has been made using only 3 and 10 cent stamps. Strong inductive hypothesis: all values from 18 to n have been successfully formed.
      3. Based on that hypothesis, P(n + 1) can be made with P(n – 2) + a 3 cent stamp, proving the Strong hypothesis.

5.3

* 1. f(2) = -1, f(3) = 5, f(4) = 2, f(5) = 17

1. Induction: Fibonacci Sequence: 1, 1, 2, 3, 5, 8, 13, 21…
   1. Basis step: P(1): f1 = f(2\*1) = f2; 1 = 1, true.
   2. Inductive Hypothesis: P(n) = f1 + f3 + f5 + … + f(2n – 1) = f2n; we assume that to be true.
   3. P(n + 1) = f1 + f3 + f5 + … + f(2n-1) + f(2n + 1) = f(2n + 2), meaning
      1. f(2n) + f(2n + 1) = f(2n + 2) by definition of the Fibonacci sequence, making P(n + 1) true.

--

Basis steps:

length = 1

(if (or (null? numList) (null? (cdr numList)) becomes (if (or (false) (true)), forcing the program to return true.

Length = 2 with a strictly ascending list

The first if statement returns false, and the second statement (if (>= (car numList) (cadr numList)) becomes (if (false)), which recursively calls itself.

The second time through, the first if statement returns true because the cdr of the list is null, and the program returns true.

Length = 2 with a descending list

The first if statement returns false, and the second statement becomes (if (true)), which makes the program return false.

Inductive step: For a list of length n, our program will return true if it is strictly ascending, and false if it is not strictly ascending.

Using length of n + 1,

The program iterates through the list each time, checking if the cdr of the list is null; if it is null and the program has reached that point without returning false, it returns true as the list is strictly ascending. If not, it checks if the current value of the list is greater than the next element of the list; if it is, the list is not strictly ascending and the program returns false; otherwise, the program recursively calls itself with the cdr of the list and continues checking. This works for n + 1, and therefore works for all n.