Name: Connor Deakin
04/10/17

CS 330 Homework 2

① Prove $Sumsq(n) = \frac{2n^3 + 3n^2 + n}{6}$

```
Sumsq(n)
{
  if (n==1)  return 1;
  else  return (Sumsq(n-1) + n*n)
}
```

$Sumsq(n) = \sum_{i=1}^{n} (i^2)$ ← our recursive function in terms of summation (not really necessary).

Base Case: $n=1$

$Sumsq(1) = \sum_{i=1}^{1} (i^2) = (2(1^3) + 3(1^2) + 1)/6$

$= \quad 1 = 6/6$

$1 = 1$

Base case proven to be true

Induction

Assume: $Sumsq(n) = (2n^3 + 3n^2 + n)/6$

Show: $Sumsq(n+1) = (2(n+1)^3 + 3(n+1)^2 + (n+1))/6$

$\underbrace{1^2 + 2^2 + \dots + n^2} + (n+1)^2 = (2(n+1)^3 + 3(n+1)^2 + (n+1))/6$

$\frac{2n^3 + 3n^2 + n}{6} + (n+1)^2 =$ "

$\frac{2n^3 + 3n^2 + n + 6(n+1)^2}{6} =$ "

$$\frac{2n^3 + 3n^2 + n + 6n^2 + 12n + 6}{6} = \quad ''$$

$$\frac{2n^3 + 9n^2 + 13n + 6}{6} = \frac{2(n+1)(n^2+2n+1) + 3(n^2+2n+1) + n+1}{6}$$

$$'' \quad = \frac{2(n^3+2n^2+n+n^2+2n+1) + 3n^2+6n+3+n+1}{6}$$

$$'' \quad = \frac{2(n^3+3n^2+3n+1) + 3n^2+7n+4}{6}$$

$$\frac{2n^3 + 9n^2 + 13n + 6}{6} = \frac{2n^3 + 9n^2 + 13n + 6}{6}$$

We have proven that Sumsq(n) works $\forall (n \geq 1)$

$$\downarrow$$

$$\forall (n \geq 1)\left(\text{Sumsq}(n) = (2n^3 + 3n^2 + n)/6\right)$$

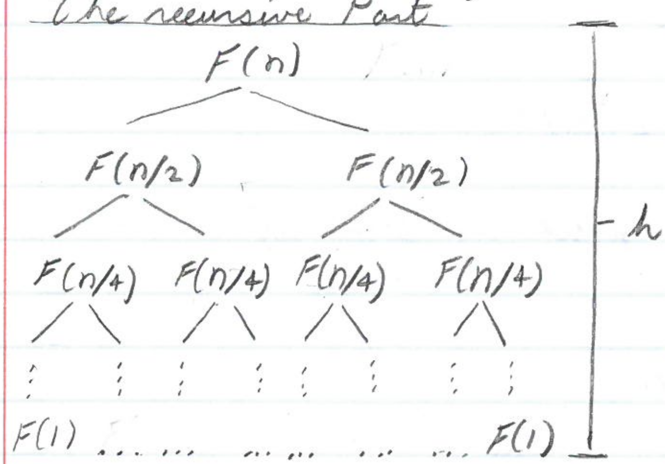
② Runtime equation: $F(n) = 2F(n/2) + n$: $F(1) = 1$

a. write an algorithm.

```
func (n)
{
    if (n==1) return 1;
    thing += func(n/2);
    thing += func(n/2);
    for ( i=0; i<n; ++i ) { thing += other_thing; }
    return thing;
}
```

b. Solve Recurrence using Recursion Tree

<u>The recursive Part</u>



h = tree height

$h = \log_2(n)$

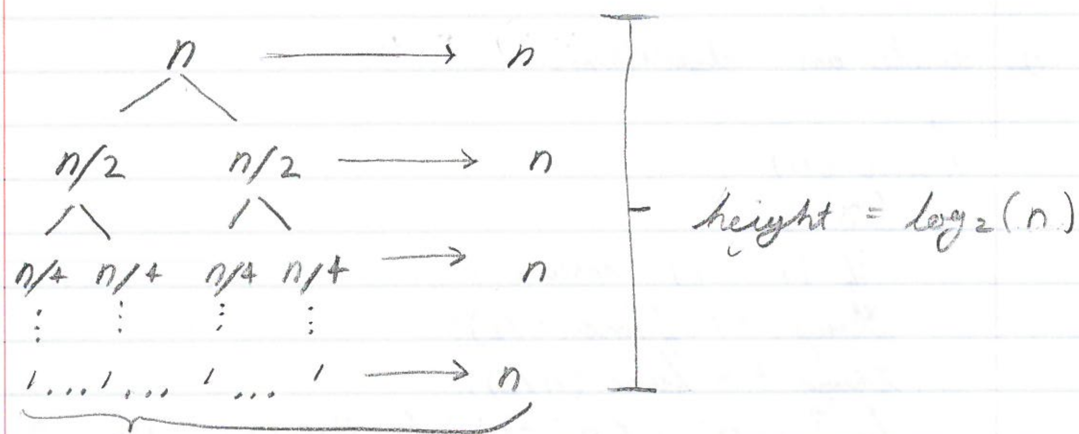
2^h = Width of bottom leaves

$2^{\log_2(n)}$ = "

$n^{\log_2(2)}$ = "

n = Width / Run time of base cases

The iterative part



these are from the base cases, which we already know, from the previous tree. So the actual height of our iterative part is

$$\log_2(n) - 1$$

Our iterative part in term of run time.

$$\sum_{i=1}^{\log_2(n)-1} n$$

With the run time of our iterative and recursive parts, we can find the total run time $T(n)$

$$T(n) = \sum_{i=1}^{\log_2(n)-1} n + Cn$$

C = runtime of the base case.

$$T(n) = (\log_2(n)-1)(n) + Cn$$

c. find and prove the order of growth

$f(n) = n(\log_2(n) - 1) + n$

$g(n) = n(\log_2(n))$

Show that $f(n) \in O(c\,g(n))$

$f(n) \leq c\,g(n)$ say $c=1$

$n(\log_2(n) - 1) + n \leq c\,n(\log_2(n))$

$n(\log_2(n)) - n + n \leq n(\log_2(n))$

$n(\log_2(n)) \leq n(\log_2(n))$

These are equivalent statements. It's always true, for all n where $n \neq 0$, so

$\forall(n > 0)(f(n) \leq g(n))$