Julia Nelson Homework Assignment 2 September 23, 2019

"I pledge my honor that I have abided by the Stevens Honor System." J Nelson

pg 67

#4(abcde) ALGORITHM:

Mystery(n)
// Input: A non-neg int n
S ← 0
for i ← 1 to n do
 S ← S + i * i
return S

a) This algorithm computes... $\sum_{i=1}^{n} i^2$

sum of numbers squared up to n

$n=1 \rightarrow 1$
$n=2 \rightarrow 1^2+2^2 = 5$
$n=3 \rightarrow 14$, $1^2+2^2+3^2$, $1+4+9$

b) The basic operation is Multiplication

c) The basic operation is ~~comput~~ executed n times.

d) The efficiency class of the algorithm is $\Theta(n)$ because loop executed n times
$\{1+n+1 = n+2 \approx n\} = \Theta(n)$

e) find better efficiency than $\Theta(n) \rightarrow$ single line of arithmetic seq. $\Rightarrow \Theta(1)$

unsure how to compute

pg 76 (#1abcde)

#1 a) $x(n) = x(n-1) + 5$ for $n > 1$, $x(1) = 0$ Backwards Subs.

$= x(n-2)+5+5$

$\Rightarrow x(n-i) + 5i$ $= 0$ if $i = n-1$

$x(n-2)+5+5 = x(n-2)+2(5)$

$x(n-3)+5+5+5 = x(n-3)+3(5)$

$= x(n-(n-1)) + 5(n-1)$

$= x(1) + 5(n) - 5$

$= x(1) + 5 \cdot n - 5$

$= \underbrace{x(1)}_{0} + \underbrace{(n-1)5}_{5(n-1)}$ $\Rightarrow x(n) = \boxed{5(n-1)}$

b)

$x(n) = 3x(n-1)$ for $n > 1$ $x(1) = 4$

$x(n) = 3x(n-1)$
$= 3 \cdot 3x(n-2)$
$= 3^2 x(n-2)$
...
$= 3^i x(n-i)$
(let $i = n-1$)
$\Rightarrow 3^{(n-1)} x(n-(n-1))$
$= 3^{n-1} x(1)$
$= 3^{n-1}(4)$
$= 4 \cdot 3^{n-1}$
$\rightarrow \boxed{4 * 3^{n-1}}$

$\Rightarrow 3^2(3x(n-3)) \Rightarrow$
$\Rightarrow 3^3 x(n-3)$

c)

$x(n) = x(n-1) + n$ for $n > 0$ $x(0) = 0$

$= x(n-2) + n + n$
$= x(n-i) + n \cdot i$
~~(let $i = n$)~~
$\Rightarrow \cancel{x(n-n) + n \cdot n}$
$= \cancel{x(0) + n^2}$
$= \cancel{0 + n^2}$

$= x(n-2) + (n-1) + n$

$= x(n-3) + (n-2) + (n-1) + n$

$x(n-2) + (n-1) + n$
$x(n-3) + (n-2) + (n-1) + n$
$= x(n-i) + (n-i+1) + (n-i+2) \ldots n$
$= x(0) + 1 + 2 + \ldots + n$
$\frac{n(n+1)}{2}$ $\boxed{= \frac{n^2+n}{2}}$

$1 \quad \frac{1+1}{2} = 1 \quad \frac{0}{2}$
$4 \quad \frac{4+2}{2} = 3 \quad \frac{9+3}{2} = 6$

d)

$x(n) = x(n/2) + n$ $n > 1$
$x(1) = 1$ (solve for $n = 2^k$)

let $n = 2^k$
$= x(2^k) = x\left(\frac{2^k}{2}\right) + 2^k \Rightarrow x(2^{k-1}) + 2^k$
$= x(2^{k-2}) + 2^{k-1} + 2^k \leftarrow x(2^{k-2}) + 2^{k-1} + 2^k$
$x(2^{k-3}) + 2^{k-2} + 2^{k-1} + 2^k$
$= x(2^{k-i}) + 2^{k-i+1} + \ldots + 2^k$
let $i = k$
$= x(2^{k-k}) + 2^{k-k+1} + 2^k$
$= x(2^{k-k}) + 2^1 + 2^2 + \ldots + 2^k$
$x(2^0)$
$x(1) = 1$
$= 1 + 2^1 + 2^2 + \ldots + 2^k = 2^{k+1} - 1$
$= 2(2^k) - 1$
$= \boxed{2n-1}$

e) $x(n) = x(n/3) + 1$ for $n > 1$

$x(1) = 1$

solve for $n = 3^k$

$x(3^k) = x(3^k/3) + 1$

$= x(3^{k-1}) + 1$

$= x(3^{k-2}) + 1 + 1$ ← $x(3^{k-2}) + 2$

$= x(3^{k-3}) + 1 + 1 + 1$ ← $x(3^{k-3}) + 3$

$\Rightarrow x(3^{k-i}) + i$

if $n = 3^k$

$k = \log_3 n$

let $i = k$

$x(3^{k-k}) + k$

$= x(3^0) + k$

$= x(1) + k \Rightarrow 1 + k$

$\boxed{= 1 + \log_3 n}$

pg 76 #3

Algorithm: $S(n)$

// Input: A positive int n

// output: Sum of the first n cubes

if $n = 1$ return 1

else

return $S(n-1) + n*n*n$

$S(n) = 1^3 + 2^3 + \ldots + n^3$

times Mult. is executed

a) $X(n) = X(n-1) + 2$ $X(1) = 0$

$= X(n-2) + 2 + 2$ ← $X(n-2) + 2i$

$= X(n-3) + 2 + 2 + 2$ ← $X(n-3) + 2^3 i$

$\Rightarrow X(n-i) + 2i$

let $i = n-1$

$X(n-(n-1)) + 2(n-1)$

$= X(1) + 2(n-1)$

$= 0 + 2(n-1)$ $\boxed{= 2(n-1)}$

b) Non recursive sum of first n cubes executes the same # of multiplications but rather than being recursive, it could use a for loop.

$$\left\{ \begin{array}{l} S \leftarrow 1 \\ \text{for } i \leftarrow 2 \text{ to } n \\ \quad S \leftarrow S + i*i*i \\ \text{return } S \end{array} \right\}$$

(similar to question #4 pg 67)