Khayyam Saleem
CS385 HW2
10/2/2016
I pledge my honor that I have abided by the Stevens Honor System.

p. 67; #4

a. The algorithm computes the sum of the first n real squares, beginning at 1.
b. The basic operation of the algorithm is the multiplication within the loop.
c. The basic operation of the algorithm is computed n times.
d. The efficiency class of the algorithm is θ(n) because the basic operation is performed once for every number between 1 and n by the parameters of the loop.
e. There exists a formula for the computation of the first n squares.

$$\sum_{i=1}^{n} i^2 = \frac{n(n+1)(2n+1)}{6}$$

Using this formula, we can compute the first n squares in θ(1) time given n.

P. 76; #1

a. $x(n) = x(n-1) + 5;\ n > 1;\ x(1) = 0$

$$\begin{aligned}
&= (x(n-2) + 5) + 5 = x(n-2) + 5*2 \\
&= (x(n-3) + 5) + 5*2 = x(n-3) + 5*3 \\
&\quad \ldots \\
&= (x(n-1) + 5) + 5*i \\
&= x(1) + 5*(n-1) \\
&= 5*(n-1)
\end{aligned}$$

b. $x(n) = 3x(n-1);\ n > 1;\ x(1) = 4$

$$\begin{aligned}
&= 3(3x(n-2)) = 3^2x(n-2) \\
&= 3(3(3x(n-3))) = 3^3x(n-3) \\
&\quad \ldots \\
&= 3^ix(n-i) \\
&= 3^{n-1}x(n-1) \\
&= 3^{n-1}x(1) \\
&= 4*3^{n-1}
\end{aligned}$$

c. $x(n) = x(n-1) + n;\ n > 0;\ x(0) = 0$

$$\begin{aligned}
&= (x(n-2) + (n-1)) + n = x(n-2) + (n-1) + n \\
&= (x(n-3) + (n-2) + (n-1)) + n = x(n-3) + (n-2) + (n-1) + n \\
&\quad \ldots \\
&= x(n-i) + (n-i+1) + (n-i+2) + \ldots + n \\
&= x(0) + 1 + 2 + 3 + \ldots + n = \frac{n(n+1)}{2}
\end{aligned}$$

d. $x(n) = x(\frac{n}{2}) + n;\ n > 1;\ x(1) = 1$
... Solving for $n = 2^k$ ...
$x(n) = x(2^{k-1}) + 2^k$
$= [x(2^{k-2}) + 2^{k-1}] + 2^k = x(2^{k-2}) + 2^{k-1} + 2^k$
...
$= x(2^{k-i}) + 2^{k-i+1} + 2^{k-i+2} + \ldots + 2^k$
$= x(2^{k-k}) + 2^1 + 2^2 + \ldots + 2^k$
$= 2^{k+1} - 1$
$= 2 * 2^k - 1$
$= 2n - 1$

e. $x(n) = x(\frac{n}{3}) + 1;\ n > 1;\ x(1) = 1;\ solve\ for\ n = 3^k$
$= x(3^k) + 1$
$= [x(3^{k-2}) + 1] + 1 = x(3^{k-2}) + 2$
$= [x(3^{k-3}) + 1] + 2 = x(3^{k-3}) + 3$
...
$= x(3^{k-i}) + i$
$= x(3^{k-k}) + k = x(1) + k$
... *Substitute* $n = 3^k$ ...
$= 1 + log_3 n$

p. 76-77; #3

a. *Algorithm* => $A(n);\ A(1) = 0$
$A(n) = A(n-1) + 2$
$= [A(n-2) + 2] + 2 = A(n-2) + 2 * 2$
$= [A(n-3) + 2] + 2 + 2 = A(n-3) + 2 * 3$
...
$= A(n-i) + 2i$
$= M(1) + 2(n-1)$
$= 2(n-1)$

b. *Nonrecursive Algorithm*
```
int c = 1;
for (int i = 2; i <= n; i++){
    c += i*i*i;
};
return c
```

This algorithm performs the exact same number of basic operations as the recursive version, but without the overhead that comes with the repeated function calls required for the recursive version.