Brandon Patton

September 30, 2018

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

Brandon Patton

p. 67

4.

a. Computes the square of each number leading up to n and adds it to the previous square.

b. The function takes in a non-negative integer n and then uses the values of in a calculation that takes the previous square (S) and adds it to i\*i and sets it equal to S to store for the next calculation. The loop iterates using values of i up to n and returns S at the end of the loop.

c. The basic operation is executed n times.

d. Θ(n)

e. Using one could optimize this program to do the calculation without needing to loop

p. 76

1.

a.

x(n) = x(n – 1) + 5 for n > 1, x(1) = 0

x(n – 1) = x(n – 2) + 5

1st step: x(n) = x(n – 2) + 5 + 5

x(n – 2) = x(n – 3 ) + 5

2nd step: x(n) = x(n – 3) + 5 + 5 + 5

3rd step: x(n) = x(n – i) + 5i

4th step: n – i = 1, n = i + 1

5th step: sub in: x(n) = x(n – (n – 1)) + 5 + 5(n – 1)

= 5n, Θ(n)

b.

x(n) = 3x(n – 1) + 5 for n > 1, x(1) = 4

x(n – 1) = 3x(n – 2)

1st step: x(n) = 32x(n – 2)

x(n – 2) = 3x(n – 3)

2nd step: x(n) = 33x(n – 3)

3rd step: x(n) = 3ix(n – i)

4th step: n – i = 1, n = i + 1

5th step: sub in: x(n) = 3n-1x(n – (n - 1))

= 3n-1\*4, Θ(3n)

c.

x(n) = x(n – 1) + n for n > 0, x(0) = 0

x(n – 1) = x(n – 2) + (n – 1)

1st step: x(n) = x(n) = x(n – 2) + (n – 1) + n

x(n – 2) = x(n – 3) + (n – 2)

2nd step: x(n) = x(n – 3) + (n – 2) + (n – 1) + n

3rd step: x(n) = x(n – i) + (n – i + 1) + (n – i + 2) + . . . + n

4th step: n – i = 0, n = i

5th step: sub in: x(n) = x(n – n) + (n – n + 1) + (n – n + 2) + n

= n(n+1)/2, Θ(n)

d.

x(n) = x() + n for n > 1, x(1) = 1 shouldve plugged in 2^k from the start

x() = x() +

1st step: x(n) = x() + + n

x() = x() +

2nd step: x(n) = x() + + + n

3rd step: x(n) = x() + () + () + . . . + n

4th step: () = 1, n = 2^k, lg n = k

5th step: sub in: x(n) = x() + () + () + n

= 7 + n, Θ(n)

e.

x(n) = x() + 1 for n > 1, x(1) = 1

x() = x() + 1

1st step: x(n) = x() + 1 + 1

x() = x() + 1

2nd step: x(n) = x() + 1 + 1 + 1

3rd step: x(n) = x() + 1 \* k

4th step: () = 1, n = 3^k, n = k

5th step: sub in: x(n) = () + 1(n)

= 1 + n, Θ(n)

p. 77

3.

a.

x(n) = x(n – 1) + for n > 1, x(1) = 1 n cubed is 2 or 3 for some reason

x(n – 1) = x(n – 2) +

1st step: x(n) = x(n – 2) + +

x(n – 2) = x(n – 3)

2nd step: x(n) = x(n – 3) +

3rd step: x(n) = x(n – i) + (n – i + 1)3 + (n – i + 2)3 + . . . + n3

4th step: n – i = 1, n = i + 1

5th step: sub in: x(n) = x(n – (n – 1)) + (n – (n – 1) + 1)3 + (n – (n – 1) + 2)3 + n3

= 36 + n3, Θ(n3)

b. They will have the same running time; however, if the recursive algorithm is fed a 1, it simply outputs a 1, making the running time much shorter. They achieve the same goal of computing the sum of the first n cubes.