Eli Szobylak
HW 2

20-oct-15

Ch 2

2.2 $T_1(N) = \Theta(f(N))$
$T_2(N) = \Theta(f(N))$

a. $T_1(N) + T_2(N) = \Theta(f(N))$

$T_1(N) + T_2(N) = \Theta(f(N) + f(N))$

false

b. $T_1(N) - T_2(N) = o(f(N))$

true

c. $\frac{T_1(N)}{T_2(N)} = \Theta 1$

True

d. $T_1(N) = \Theta(T_2(N))$

true

2.3 which grows faster?

$N \log N$ $\qquad$ $N^{1+\epsilon/\sqrt{\log N}}, \epsilon > 0$

$N^{1+\epsilon/\sqrt{\log N}}$ grows way faster since it is exponential, and $N \log N$ is actually closer to linear.

2.7

a.

	points
(1) Sum=$\overset{1}{0}$	1
for($i=\overset{1}{0}$; $\overset{N+1}{i<N}$; $\overset{N}{i++}$)	$2N+1$
++Sum	N
total	$3N+3$
$\Theta(N)$	
(2) Sum =0	1
for(i=0; i<N; i++)	$2N+1 > 2N^2+1$
for(J=0;J<N;J++)	$2N+1$
++Sum	N
total	~~$5N+3$~~ $2N^2+N+1$
$\Theta(N^2)$	
(3) sum=0	1
for(i=0; i<N; i++)	$2N+1 > 4N^2+8N+5$
for(J=$\overset{1}{0}$;$\overset{N+2}{J<N*N}$; $\overset{N}{J++}$)	$2N+3$
++Sum	N
total	$4N^2+7N+5$
$\Theta(N^2)$	
(4) sum=0	1
for(i=0; i<N; i++)	$2N+1$
for(J=0; J<i*i; J++)	$2N+3$
for(K=0; K<J; K++)	$2N+1$
++Sum	N
total	N^3+N
$\Theta(N^3)$	
(5) Sum=0	1
for(i=0; i<n; i++)	$2N+1$
for(J=0;J<i*i; J++)	$2N+3$
for(K=0; k<J; K++)	$2N+1$
++Sum	N
total	N^3
$\Theta(N^3)$	

Scanned by CamScanner

(6) Sum=0	1
for(i=0;i<N;i++)	2N+1
for(J=1;J<i*i;J++)	3N+1
~~for~~(J % i ==0)	2N+1
for(k=0;k<J;k++)	2N+1
++Sum	N
Total	N^4

$O(N^4)$

run Time

6.	N=10	1000	2000 ~~100,000~~	4000 ~~1,000,000~~
(1)	0.1s	0.1s	0.1s	0.1s
(2)	0.1s	0.1s	0.4s	1.5s
(3)	0.1s	N=200 0.7s	N=300 2.4s	N=400 5.5s
(4)	0.1s	0.1s	0.2s	0.7s
(5)	0.3s	N=80 1.7s	N=55 4.1s	N=60 6.4s
(6)	0.1s	N=100 1.1s	N=150 5.6s	N=200 19.4s

c. The fondonss above do match the run-tomes I calculated by hand in part a.

2.14

$$f(x) = 4x^4 + 8x^3 + x + 2 \quad , \quad x = 3$$

```
acc = 0
for c in (coeff)
    acc = acc * x + c
return acc
```

a)

1: acc = 0 * 3 + 4
   acc = 4

2: acc = 4 * 3 + 8
   acc = 20

3: acc = 20 * 3 + 1
   acc = 61

4: acc = 61 * 3 + 2
   acc = 185

b) Itworks because a function at a specific value will return a constant

c)
```
def horner (coeffs, x):                          4
    acc=0                                        1
    for c in reversed(coeffs):                   2N+1
        acc = acc *x+c                           3
    print acc                                    2N+5
```

$O(N^2)$

2.15

Takes array a of sorted ints
and a search element i

```
int low = 0
int high = a.length()-1

while (low <= high)
    int middle = (low + high) / 2

    if ( a[middle] < i)
        low = middle+1
    elif (a [middle] > i)
        high = middle+1
    else
        print middle

print "not found"
```

run-time analysis

Points by line:

1
2
N
1
N
1
N
1
1
1

The run-time analysis would be $\Theta(N^3)$