Homework 1 Solutions
CSCI-2300: Data Structures and Algorithms
Spring 2007

1. 2.1) Order the following functions by growth rate: N, $\sqrt{N}$, $N^{1.5}$, N^2, $N \log N$, $N \log\log N$, $N \log^2 N$, $N \log(N^2)$, $2/N$, 2^N, $2^{N/2}$, 37, $N^2 \log N$, N^3. Indicate which functions grow at the same rate.

   Solution: $2/N$, 37, $\sqrt{N}$, N, $N \log\log N$, $N \log N$, $N \log(N^2)$, $N \log^2 N$, $N^{1.5}$, N^2, $N^2 \log N$, N^3, $2^{N/2}$, 2^N. $N \log N$ and $N \log(N^2)$ grow at the same rate.

2. 2.2) Suppose $T_1(N) = O(f(N))$ and $T_2(N) = O(f(N))$. Which of the following are true?

   (a) $T_1(N) + T_2(N) = O(f(N))$
   (b) $\frac{T_1(N)}{T_2(N)} = O(1)$
   (c) $T_1(N) = O(T_2(N))$

   Solution:

   (a) True.
   (b) False. A counterexample is $T_1(N) = N^2$, $T_2(N) = N$, and $f(N) = N^2$.
   (c) False. The same counterexample applies.

3. 2.7) For each of the following six program fragments:

   (a) Give an analysis of the running time (Big-Oh will do).
   (b) Implement the code in the language of your choice, and give the running time for several values of N
   (c) Compare your analysis with the actual running times

(1)
```
sum = 0;
for( i = 0; i < n; i++ )
    sum++;
```
(2)
```
sum = 0;
for( i = 0; i < n; i++ )
    for( j = 0; j < n; j++ )
        sum++;
```
(3)
```
sum = 0;
for( i = 0; i < n; i++ )
    for( j = 0; j < n * n; j++ )
        sum++;
```
(4)
```
sum = 0;
for( i = 0; i < n; i++ )
    for( j = 0; j < i; j++ )
        sum++;
```
(5)
```
sum = 0;
for( i = 0; i < n; i++ )
    for( j = 0; j < i * i; j++ )
        for( k = 0; k < j; k++ )
            sum++;
```
(6)
```
sum = 0;
for( i = 1; i < n; i++ )
    for( j = 1; j < i * i; j++ )
        if( j % i == 0 )
            for( k = 0; k < j; k++ )
                sum++;
```

Answer:

(a) i. The running time is $O(N)$

ii. The running time is $O(N^2)$

iii. The running time is $O(N^3)$

iv. The running time is $O(N^2)$

v. j can be as large as i^2, which could be as large as N^2. k can be as large as j, which is N^2. The running time is thus proportional to $N \cdot N^2 \cdot N^2$, which is $O(N^5)$.

vi. The if statement is executed at most N^3 times, by previous arguments, but it is true only $O(N^2)$ times (because it is true exactly i times for each i). Thus the unnermost loop is only exectued $O(N^2)$ times. Each time through, it takes $O(j^2) = O(N^2)$ time, for a total of $O(N^4)$. This is an example where multiplying loop sizes can occasionally give an overestimate.

(b) Running times of the code segments for several values of N

i.

Size (N)	Time	growth rate
64	0.0048 ms	
128	0.0051 ms	1.0601
256	0.0057 ms	1.1136
512	0.0068 ms	1.1955
1024	0.0090 ms	1.3223
2048	0.0135 ms	1.5010
4096	0.0217 ms	1.6683
8192	0.0381 ms	1.7544
16384	0.0715 ms	1.8754
32768	0.1369 ms	1.9151

ii.

Size (N)	Time	growth rate
64	0.0248 ms	
128	0.0831 ms	3.3482
256	0.3145 ms	3.7836
512	1.2706 ms	4.0407
1024	4.9505 ms	3.8960
2048	19.6078 ms	3.9608

iii.

Size (N)	Time	growth rate
32	0.1513 ms	
64	1.1723 ms	7.7491
128	9.3458 ms	7.9720
256	76.9231 ms	8.2308

iv.

Size (N)	Time	growth rate
64	0.0151 ms	
128	0.0435 ms	2.8865
256	0.1570 ms	3.6095
512	0.6098 ms	3.8848
1024	2.4272 ms	3.9806

v.

Size (N)	Time	growth rate
16	0.1079 ms	
32	3.1056 ms	28.7733
64	100.0000 ms	32.2000

vi.

Size (N)	Time	growth rate
16	0.0467 ms	
32	0.6596 ms	14.1194
64	10.2041 ms	15.4694
128	166.6667 ms	16.3333

(c) i. The running time for this algorithm is $O(N)$. At each new row we have doubled the input size, so we would expect the experimental growth rate to approach 2, which it does.

ii. The running time for this algorithm is $O(N^2)$. At each new row we have doubled the input size, so we would expect the experimental growth rate to approach 2^2, which it does.

iii. The running time for this algorithm is $O(N^3)$. At each new row we have doubled the input size, so we would expect the experimental growth rate to approach 2^3, which it does.

iv. The running time for this algorithm is $O(N^2)$. At each new row we have doubled the input size, so we would expect the experimental growth rate to approach 2^2, which it does.

v. The running time for this algorithm is $O(N^5)$. At each new row we have doubled the input size, so we would expect the experimental growth rate to approach 2^5, which it does.

vi. The running time for this algorithm is $O(N^4)$. At each new row we have doubled the input size, so we would expect the experimental growth rate to approach 2^4, which it does.

4. 2.11) An algorithm takes 0.5 ms for input size 100. How long will it take for input size 500 if the running time is the following (assume low-order terms are negligible)

   (a) linear

   (b) $O(N \log N)$

   (c) quadratic

   (d) cubic

   Answer:

   (a) $\dfrac{500}{100} = \dfrac{N}{0.5}$ and solving for N, 2.5 ms.

   (b) $\dfrac{500 \log 500}{100 \log 100} = \dfrac{N}{0.5}$ and solving for N, 3.3737 ms.

   (c) $\dfrac{500^2}{100^2} = \dfrac{N}{0.5}$ and solving for N, 12.5 ms.

(d) $\dfrac{500^3}{100^3} = \dfrac{N}{0.5}$ and solving for N, 62.5 ms.

5. 2.12) An algorithm takes 0.5 ms for input size 100. How large a problem can be solved in 1 min if the running time is the following (assume low-order terms are negligible)

   (a) linear

   (b) $O(N \log N)$

   (c) quadratic

   (d) cubic

   Answer: 1 min = 60 s = 60000 ms.

   (a) $\dfrac{x}{100} = \dfrac{60,000}{0.5}$ and solving for x gives an input size of 12,000,000.

   (b) $\dfrac{x \log x}{100 \log 100} = \dfrac{60,000}{0.5}$ and solving for x gives an input size of 3,656,807.

   (c) $\dfrac{x^2}{100^2} = \dfrac{60,000}{0.5}$ and solving for x gives an input size of 34,641.

   (d) $\dfrac{x^3}{100^3} = \dfrac{60,000}{0.5}$ and solving for x gives an input size of 4,932.