

HOMEWORK 2 SOLUTIONS

1. What is the asymptotic complexity of the following methods, in terms of the Big-O notation.

a.

- The loop is executed in the series of 1, 4, 7, 10 and so on.... until N, so the number of times, the loop is being executed is : $((N - 1)/3 + 1)$.
- The Big (O) of such a function will be given by $O(N)$ as the constants are usually not taken into consideration.
- Hence the complexity of the given program is $O(N)$.

b.

- The Total number of times the loop is executed is given by $N + N - 1 + \dots + 2 + 1 = N(N-1)/2$.
- The Big (O) of $N(N-1)/2$ is given by $O(N^2)$.

c.

- The outer loop is executed only once.
- The inside loop is executing for $j = 1, 4, 16 \dots n$ i.e, the number of iterations will be $\log_4(N) + 1$.
- This can be written as $(\log_2 N)/2 + 1$.
- The big O of $(\log_2 N)/2 + 1$ is $O(\log N)$.

d.

- The fun() function gets executed $(\log_2 N)$ times.
- As the loop is executed N times, the number of iterations will be $((\log_2 N) + 2(\log_2 N) + 3(\log_2 N) + \dots + N(\log_2 N))$. This will be $N(N-1)(\log_2 N)/2$.
- The Big(O) of this equation is $N^2 (\log_2 N)$.

e.

Computing the Big(O) of all the given cases:

For case 'a' : the number of iterations is N

For case 'b' : the number of iterations is $(\log_2 N)$

For case 'c' : the number of iterations is $N * (\log_2 N)$

For default: the number of iterations is N^2 .

The longest execution time among the cases is chosen to be the Big(O) of the switch program.

Hence, the Big(O) is N^2 .

2. Prove that $3^n = O(9^n)$. Is it true that $9^n = O(3^n)$? Justify your answer.

3^n and 9^n are functions of n . Then we say that $3^n = O(9^n)$, provided there is a constant $C > 0$ and $N > 0$ such that $3^n \leq C * 9^n$ for all $n > N$

$$\lim_{n \rightarrow \infty} (3^n/9^n) = 0 \leq C.$$

$$n \rightarrow \infty$$

However the other way round does not hold good as there is no constant variable $C > 0$ such that $9^n = C * 3^n$ and for the range $n > N$.

$$\lim_{n \rightarrow \infty} (9^n/3^n) = \infty \not\leq C$$

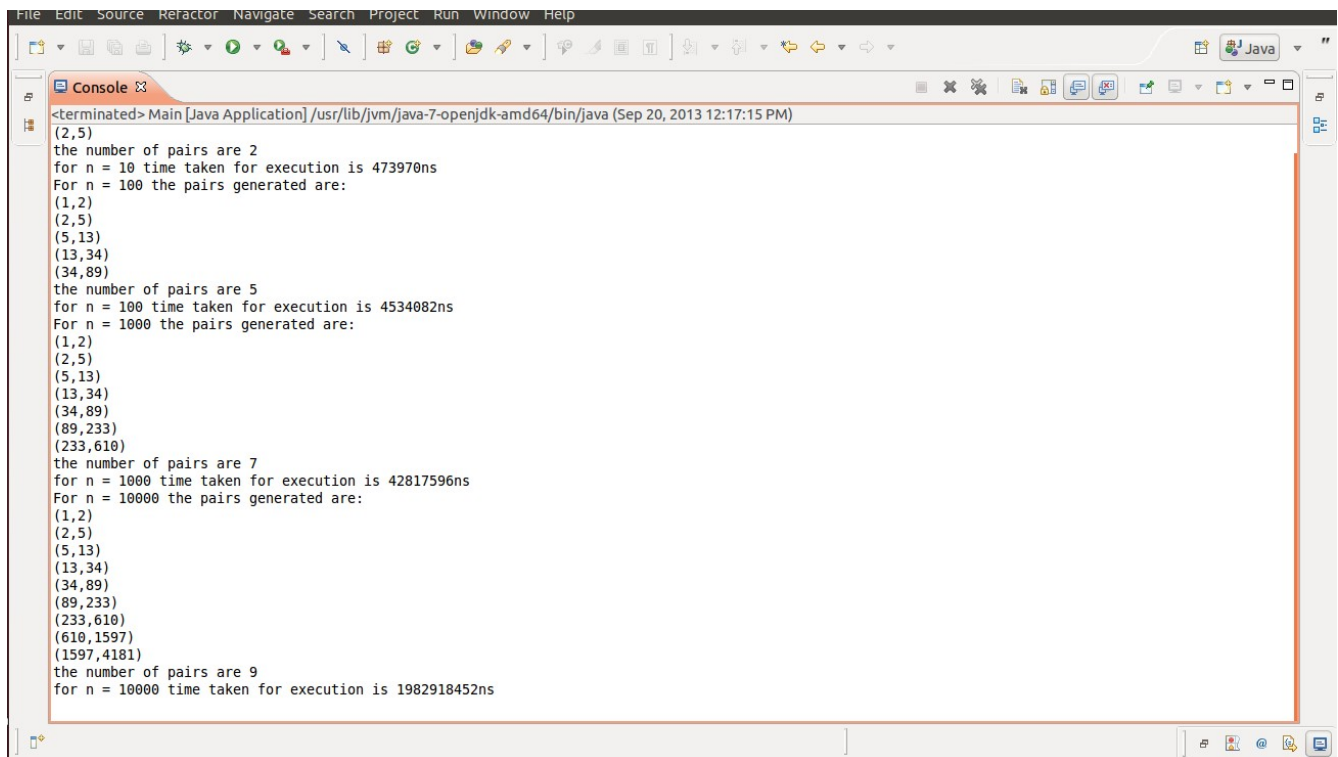
$$n \rightarrow \infty$$

3. Rank the following functions by order of growth from the slowest to the fastest (lg means \log_2).

Given below is the Big(O) of all the functions arranged by the order of the growth from the slowest to the fastest.

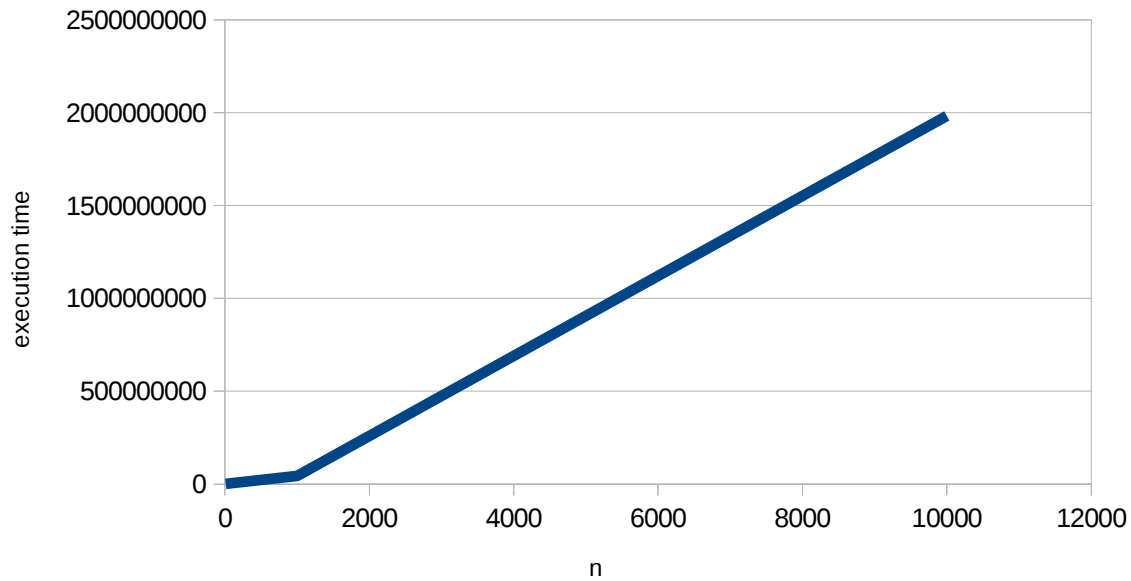
Function	Big(O)
$n^{2/3}$	$O(n^{2/3})$
$2^{\lg n}$	$O(N)$
$20n + 1000$	$O(N)$
$n^2 + 100$	$O(N^2)$
$4^{\lg n}$	$O(N^2)$
$4n^2$	$O(N^2)$
$(3/2)^n$	$O((3/2)^n)$
$2^n + n^2 + 10n$	$O(2^n)$
$n * 2^n$	$O(n \cdot 2^n)$
3^n	$O(3^n)$
$n!$	$O(N!)$
$(n+1)!$	$O(N!)$

4. Write a small Java program to determine all pairs of positive integers, (a, b), such that $a < b < n$ and $(a^2 + b^2 + 1)/(ab)$ is an integer. What is the complexity of the program? Measure the runtime of the program and plot it for values of $n=10, 100, 1000$ and 10000 . Use `System.nanoTime()` to get time measurements. Include screenshots of the program running and submit your .java code in a separate file.



```
<terminated> Main [Java Application] /usr/lib/jvm/java-7-openjdk-amd64/bin/java (Sep 20, 2013 12:17:15 PM)
(2,5)
the number of pairs are 2
for n = 10 time taken for execution is 473970ns
For n = 100 the pairs generated are:
(1,2)
(2,5)
(5,13)
(13,34)
(34,89)
the number of pairs are 5
for n = 100 time taken for execution is 4534082ns
For n = 1000 the pairs generated are:
(1,2)
(2,5)
(5,13)
(13,34)
(34,89)
(89,233)
(233,610)
the number of pairs are 7
for n = 1000 time taken for execution is 42817596ns
For n = 10000 the pairs generated are:
(1,2)
(2,5)
(5,13)
(13,34)
(34,89)
(89,233)
(233,610)
(610,1597)
(1597,4181)
the number of pairs are 9
for n = 10000 time taken for execution is 1982918452ns
```

Complexity = N^2



5. Consider two arrays (the first array and the second array) of the same size, each consisting of n random positive integers (each integer between 1 and n). Write a Java program to remove duplicates between the two arrays. If a positive integer exists in both arrays, replace it with zero in the second array. And then return total number of zeros in the second array. What is the complexity of the program? Time the program for array sizes of 10, 100, 1000 and 10000. Use `System.nanoTime()` to get time measurements. Include screenshots of the program running and submit your .java code in a separate file.

Complexity = N^2

```
File Edit Source Refactor Navigate Search Project Run Window Help
<terminated> assign2 [Java Application] /usr/lib/jvm/java-7-openjdk-amd64/bin/java (Sep 20, 2013 12:17:48 PM)
The number of elements in second array which was present in first array are 9
Execution time is 65404ns
The number of elements in second array which was present in first array are 65
Execution time is 348302ns
The number of elements in second array which was present in first array are 589
Execution time is 18866677ns
The number of elements in second array which was present in first array are 6398
Execution time is 66855817ns
```

