Swetha Jayapathy

Student ID : 934041047

Instructor : Julianne Schutfort

January 15th, 2020

CS 325 HW 1 – 30 points

1) (4 pts) For each of the following pairs of functions, select the best relationship from the options:

f(n) is O(g(n)), f(n) is Ω(g(n)), or f(n) is Θ(g(n))

a. f(n) = n0.25; g(n) = n0.5

lim n->∞ n0.25/ n0.5 = 0

f(n) is O(g(n))

b. f(n) = log n2; g(n) = lg n

lim n->∞ (log n2 )/lg n

lim n->∞  (2log n )/lg n = goes to infinity

f(n) is Θ(g(n))

c. f(n) =log logn g(n) = log n

lim n->∞ (log logn )/log n

f(n) is Ω(g(n))

d. f(n) = 5000n3+ n2  g(n) = 0.000001 n4

lim n->∞ (5000n3+ n2  )/( 0.000001 n4) -> Denominator grows faster = 0

f(n) is O(g(n))

e. f(n) = nlogn + n; g(n) =n√𝑛

f(n) is O(g(n))

f. f(n) = en; g(n) = 2n

lim n->∞ (e/2) n

f(n) is Ω(g(n))

g. f(n) = 2n; g(n) = 2n+1

lim n->∞ 2n /2n+1 = ½

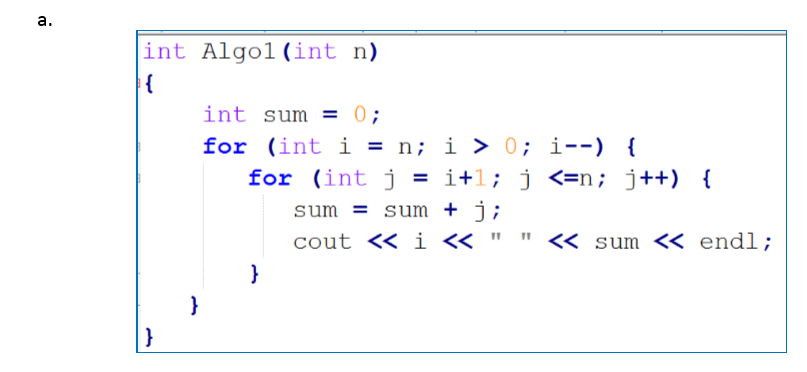
f(n) is Θ(g(n))

h. f(n) = nn; g(n) = n!

lim n->∞ nn / n! = ∞

f(n) is Ω(g(n))

2) (6 pts) Determine the theoretical running time of the following algorithms. Use theta notation and give a brief explanation.

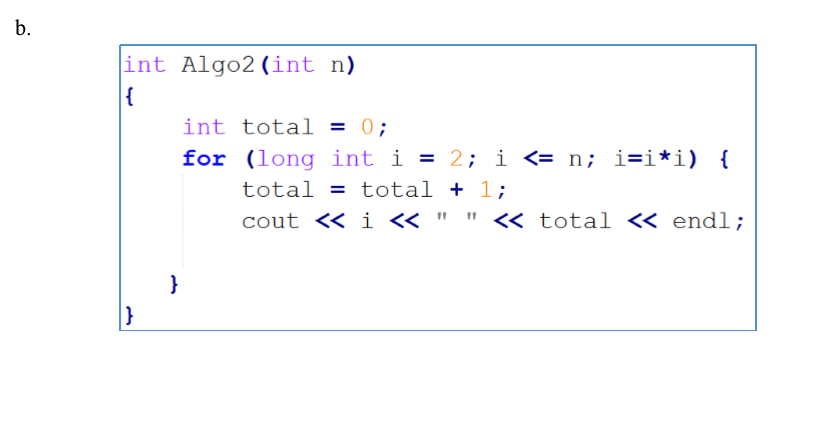


The first for loop runs for n times

The second for loop also runs for n times, lets take n = 5,

The second loop fails for the first time, but then runs upto n times after i gets decremented. And it takes constant time for the sum and Print statements.

Therefore, the Run time complexity is Θ(n2)



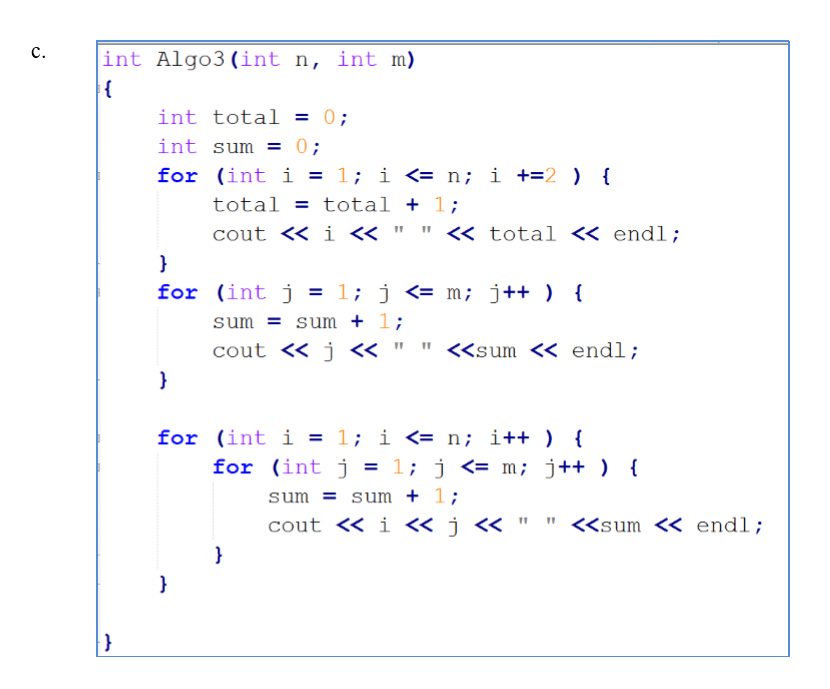
We can see that i value is doubled for each loop as below

i2 <= n

=O(√𝑛)

The other statements are run for a constant time, hence it is negligible.

Thereby we would get 2k ,since it is raised to the powers of 2, we can write it as Θ (lg n)



* The first for loop runs for n2 times, the inner statements takes constant time, therefore we can say the complexity of this as n2
* The second for loop runs m times
* The third for loop runs n\*m times

Hence we would get a polynomial as n2+( n\*m)+m

Therefore the run time complexity is Θ(n2).

4.

Insertion Sort

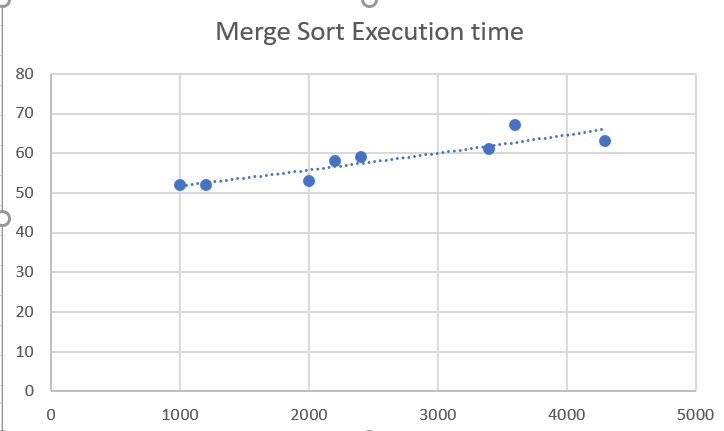
Execution time in Milliseconds:

|  |  |
| --- | --- |
| Insertion Sort | |
| n value | Execution time |
| 1000 | 51 |
| 1200 | 58 |
| 1300 | 54 |
| 1500 | 60 |
| 1600 | 64 |
| 2300 | 67 |
| 4000 | 70 |
| 5000 | 85 |



Merge Sort :

|  |  |
| --- | --- |
| Merge Sort | |
| n value | Execution time |
| 1000 | 52 |
| 1200 | 52 |
| 2000 | 53 |
| 2200 | 58 |
| 2400 | 59 |
| 3400 | 61 |
| 3600 | 67 |
| 4300 | 63 |



c. Run time are as below :

Merge sort – logn\*n

Insertion sort – n^2

e) Comparison

Comparing the two curves of merge sort and insertion sort, there are no anomalies. Both fits the run time data perfectly.