Analysis_WrittenQuestions

Question 1)

Part a)

i) Fine is 0.2 and doubles each day. The fine after *n* days will be 2□*0.2

ii) How many days will it take for the fine to reach 50 euro →

2□*0.2 = 50 euro

Nlog(2) + log(0.02) = log(50)

N ≈ 11 days.

Part b)

Download of a file from the internet takes 10 seconds to establish the initial connection. After that, the download rate is 5mb/s.

i) Time complexity of downloading a 500mb file: seconds*5mb/s + 10s.

500mb = 500/5 seconds + 10 = 110 seconds

ii) The time complexity of downloading 100 separate files is linear: O(N)

Part c)

For each of the following fragments of code, give the running time in terms of N:

1. i.
```
sum = 0;
for(int i = 0; i < n; ++i) {
    sum += i;
}
```
Single for loop from 0 to n → O(n)

2. ii.
```
sum = 0;
for(int i = 0; i < n; ++i) {
    for(int j = 0; j < n*n; ++j) {
        sum++;
    }
}
```
Two separate for loops: first from 0 to n, second from 0 to n*n → O(n³)

3. iii.
```
sum = 0;
for(int i = 0; i < n; ++i) {
    for(int j = 0; j <i ++j) {
        sum++;
    }
}
```
Two separate for loops, both from 0 to n → O(n²)

Part d)

An algorithm takes 0.5ms for input size n = 100. How long will the algorithm take to run for input size n = 5000 if the running time is…

1. <u>linear</u>
   0.5ms : 100 = x : 5000
   → x = 25

2. <u>quadratic</u>
   0.5ms : 100² = x : 5000²

→ x = 1250

3. O(n log n)
   0.5ms : 100log(100) = x : 500log(500)
   → x = 46.24

4. O(2□)
   0.5ms : 2^{100} = x : 2^{5000}
   → x =

Part e)
The number of operations performed by algorithm Foo() is $3n^3+6$ and by algorithm Bar() is $30n^2+4n$. Determine n_0 such that Foo() is better than Bar() for $n \geq n_0$.

A. $3n^3+6 \geq 30n^2+4n$
   $3n^3 - 30n^2 - 4n + 6 \geq 0$
   → $n \geq 10$

Part f)
Order the following functions by asymptotic growth rate:

1. $5n$
2. 8^n
3. $2^{\log n+4n}$
4. 2^{12}
5. $6n \log n + 6$
6. $6n \log n^2 + 6n^2$

A. 4 < 1 < 5 < 6 < 2 < 3

Part g)
Show that $(n + 5)^5$ is $O(n^5)$?

A. $(n + 5)^5 = n^5 + 25n^4 + 250n^3 + 1250n^2 + 3125n + 3125$
   Since we only consider the n with the highest degree, then the time complexity is $O(n^5)$.

Question 2)
The Java class TripleSum reads a list of long integers from the command line and counts the number of triples in the input integers which sum to exactly 0.

```
METHOD TripleSum()
Input: an array of integers, int[] array.

      for i in range 0 to n, do
          for j in range i to n, do
              for k in range j to n, do
                  if array[i] + array[j] + array[k] = 0, then
```

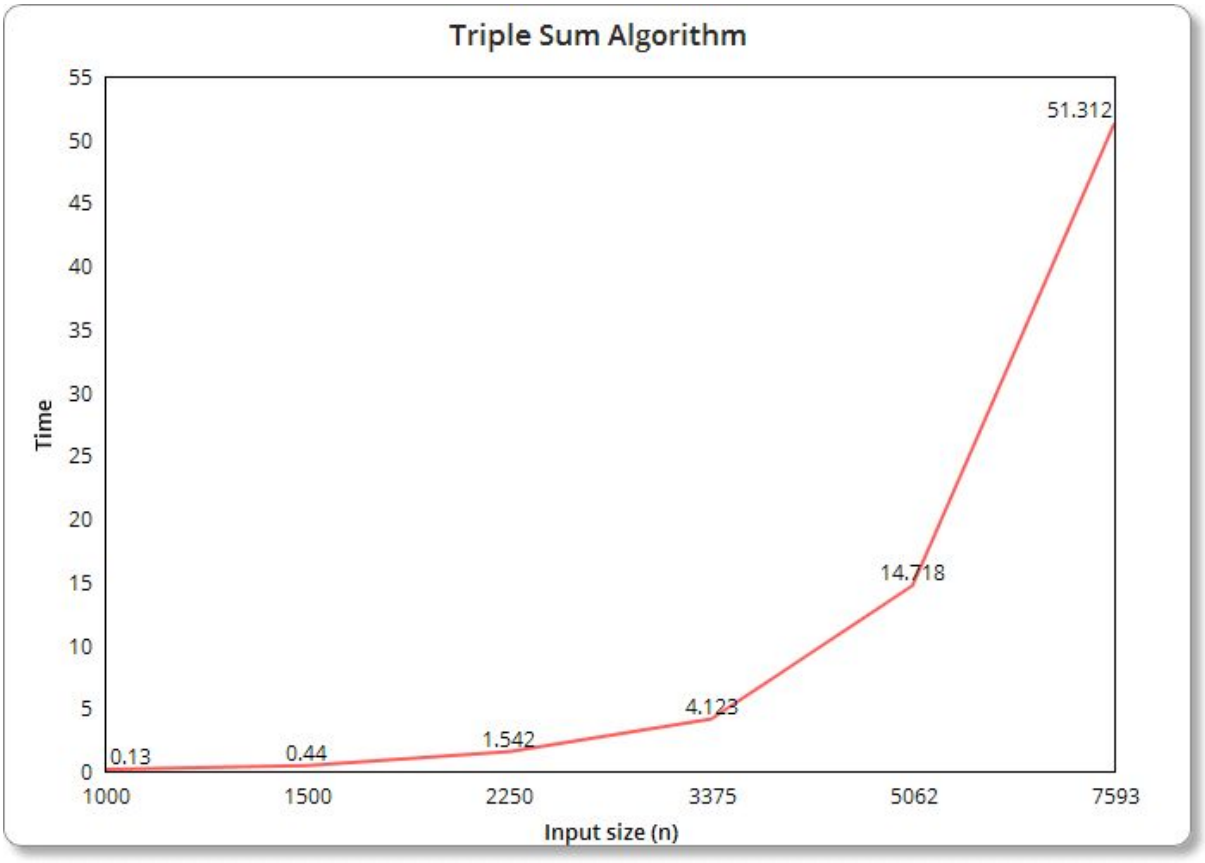
```
                    print array[i], array[j], array[k]
return
```

Due to the three for loops that traverse the entire array, we are led to believe that the time complexity of the algorithm will be $O(n^3)$

Empirical Analysis:

Input Size (n)	Time elapsed - TripleSum
1000	0.13
1500	0.44
2250	1.542
3375	4.123
5062	14.718
7593	51.312

Timing Graph:



Question 3)
Sorting Algorithms - Empirical Analysis

Input Size (n)	Time elapsed - Insertion Sort	Time elapsed - Selection Sort	Time elapsed - Bubble Sort
10	0.0	0.0	0.0
100	0.0	0.0	0.0
1000	0.006	0.005	0.006
10000	0.041	0.097	0.082
100000	3.336	3.865	3.921

From the empirical analysis, we are lead to believe that the Time complexity of the analysed sorting algorithms (Insertion Sort, Bubble Sort and Selection Sort) is O(n). This assumption is confirmed by the timing graph below: