# ADS Homework: 1

Aashish Paudel
Mat. Id: 30001861

February 14, 2019

## Problem 1

### (a)

Given,
$f(n) = 3n$
$g(n) = n^3$
Now, we can define,
$\phi(g(n)) = \{f(n)|$ for any constant c $\exists n_0 > 0 : 0 \leq f(n) \leq cg(n), \forall n \geq n_0\}$
$\because$ cubic function increases faster than linear function

Hence, comparing $\phi$ with the function we have, we can say:
$f(n) \in o(g(n))$, or
$g(n) \in \omega(f(n))$

### (b)

Let's compute:

$$\lim_{x\to\infty} \frac{7n^0.7 + 2n^0.2 + 13\log n}{\sqrt{n}}$$

$=\infty$

Thus, by above definition,
$f(x) \in \omega g(n)$, or reversely,
$g(x) \in of(n)$

### (c)

Let's compute:

$$\lim_{x\to\infty} \frac{\frac{n^2}{\log n}}{n \log n}$$

$$= \infty$$

$f(x) \in \omega g(n)$, or reversely,
$g(x) \in o f(n)$

### (d)

Given,
$f(n) = (log(3n))^3$
$g(n) = 9 \log n$
As we know for n¿1 the logarithmic graph increases slowly. The cube of graph will increase at a faster rate. So, no matter the constant before g(n), the f(n) will meet the function and eventually surpass it, i.e. $\exists N_0 s.t. f(n) > g(n) \forall n > N_0 (\forall$ constants$)$
Hence,
$f(n) \in \omega g(n)$, or reversely,
$g(n) \in o(f(n))$

## Problem 2

### (a)

The code for algorithm is:

```
for i in range(len(lst)):
    key = lst[i]
    min = i
    for j in range(i, len(lst)):
        if lst[j] < lst[min]:
            min = j
    lst[i] = lst[min]
    lst[min] = key
```

### (b)

Consider, the boolean variable loop_var which represents if the list [1, ..., i-1] is sorted
At the beginning of every loop,
list = [1, ..., i-1, ...], loop_var=True

If we continue to the end of the loop in this order,
At the beginning of last loop (at the end), i = length(list) + 1
lst = [1, ..., i-1], loop_var = True

Hence, the list must be sorted at the end of the loop.

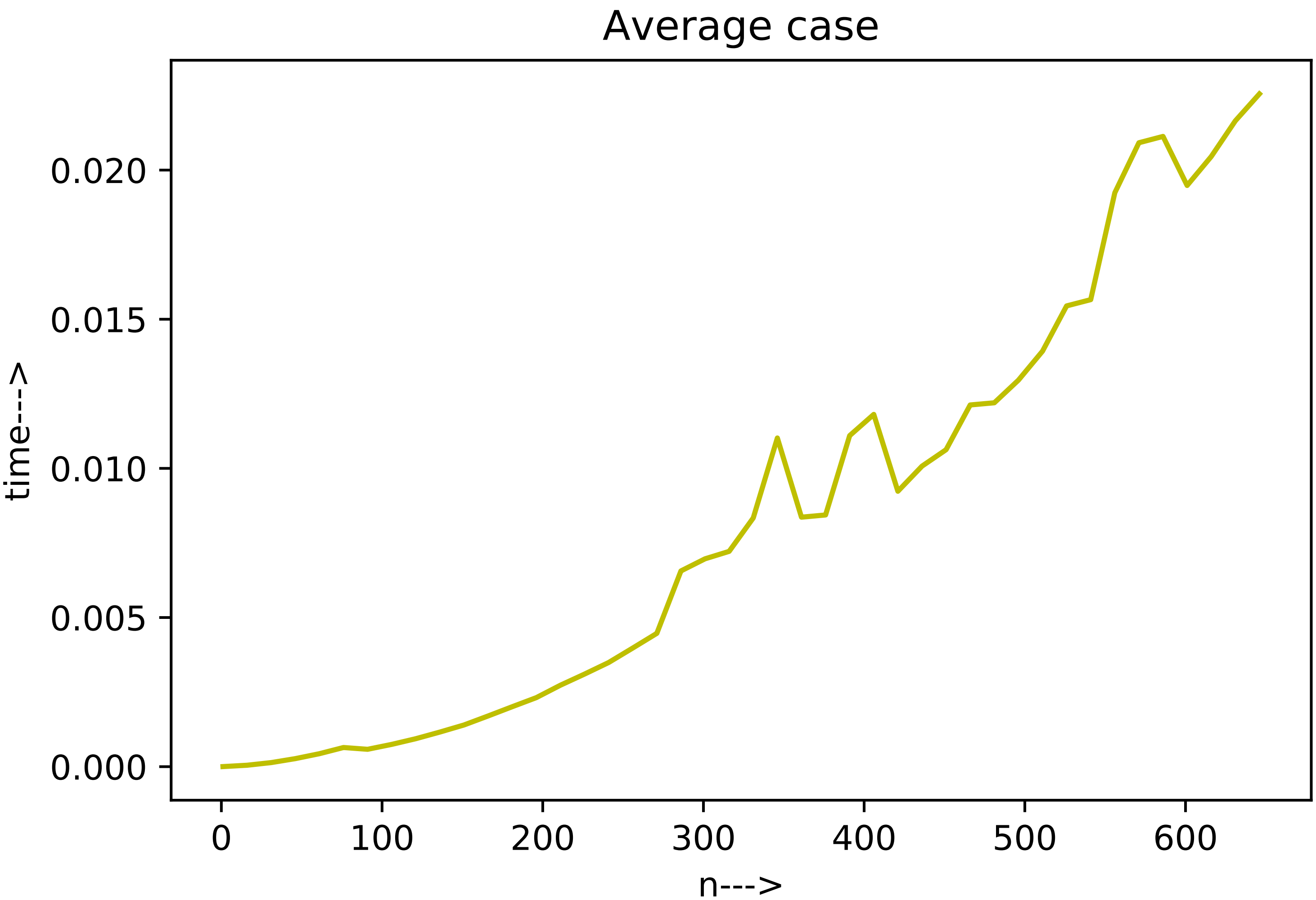
∵ Throughout the proof, the index of the list elements is shown instead of real elements because real elements are uncertain.

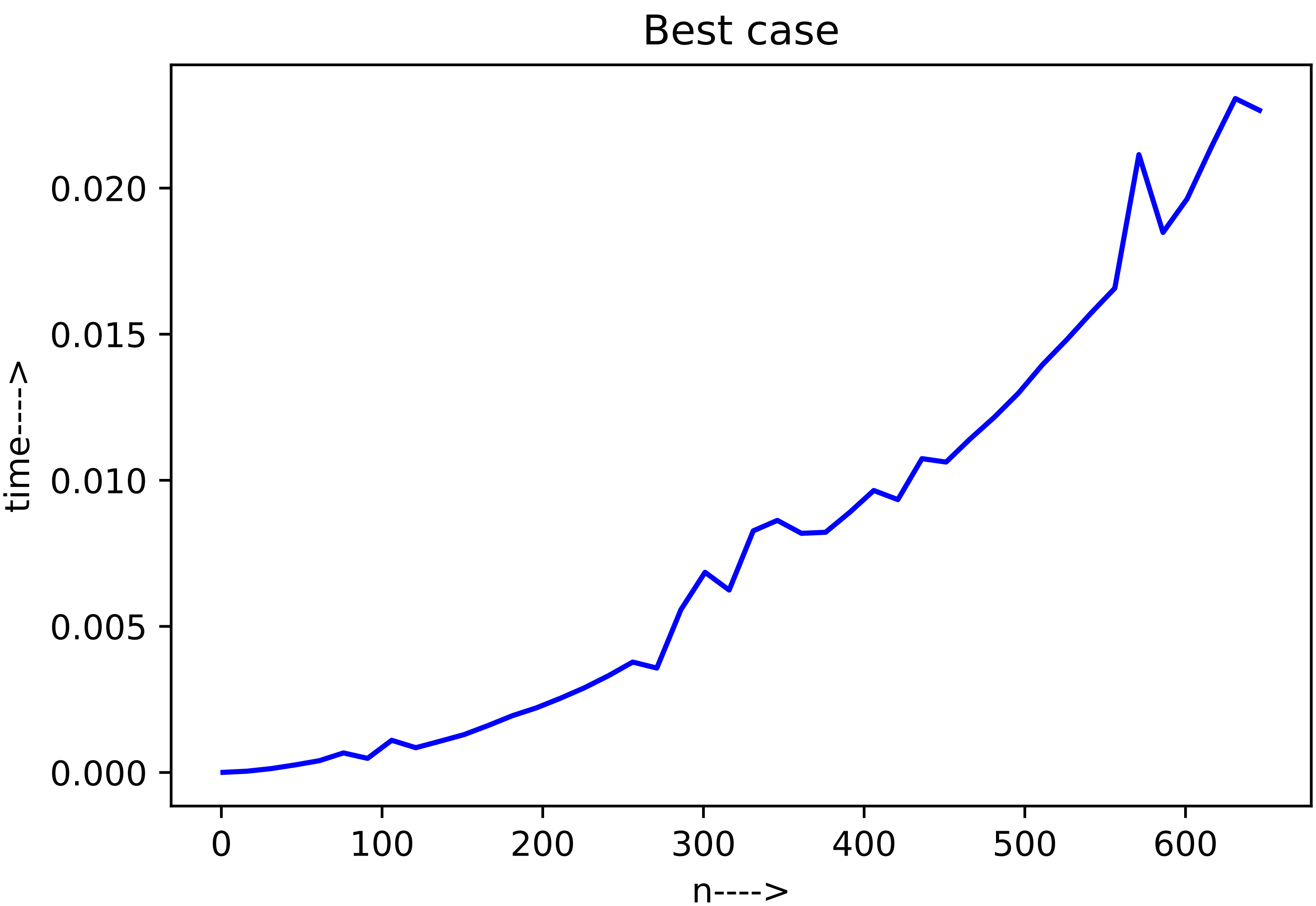
### (c)

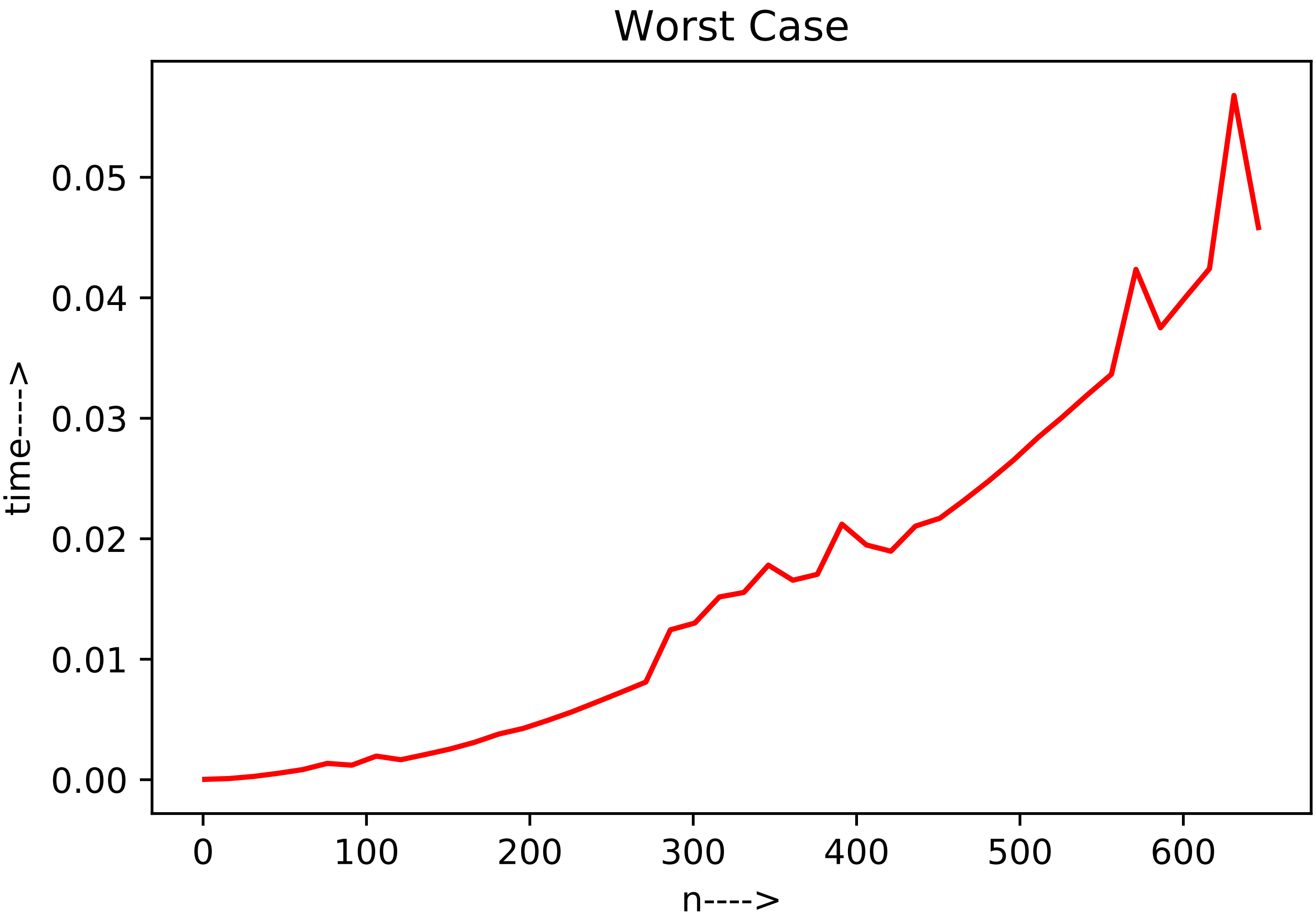
The code used to generate the required seqences is:

```
import random
NO_SEQ = 5
radom_sequences = []
for i in range(NO_SEQ):
    random_seqeunces.append([random.randrange(100) for n in range(15)])
best_case = list(range(15))
worst_case = best_case[::-1]
print(random_sequences)
print(best_case, worst_case)
```

### (d)


Average case
0.000
0.005
0.010
0.015
0.020
time--->
0
100
200
300
400
500
600
n--->


Best case
time---->
n---->
0.000
0.005
0.010
0.015
0.020
0
100
200
300
400
500
600

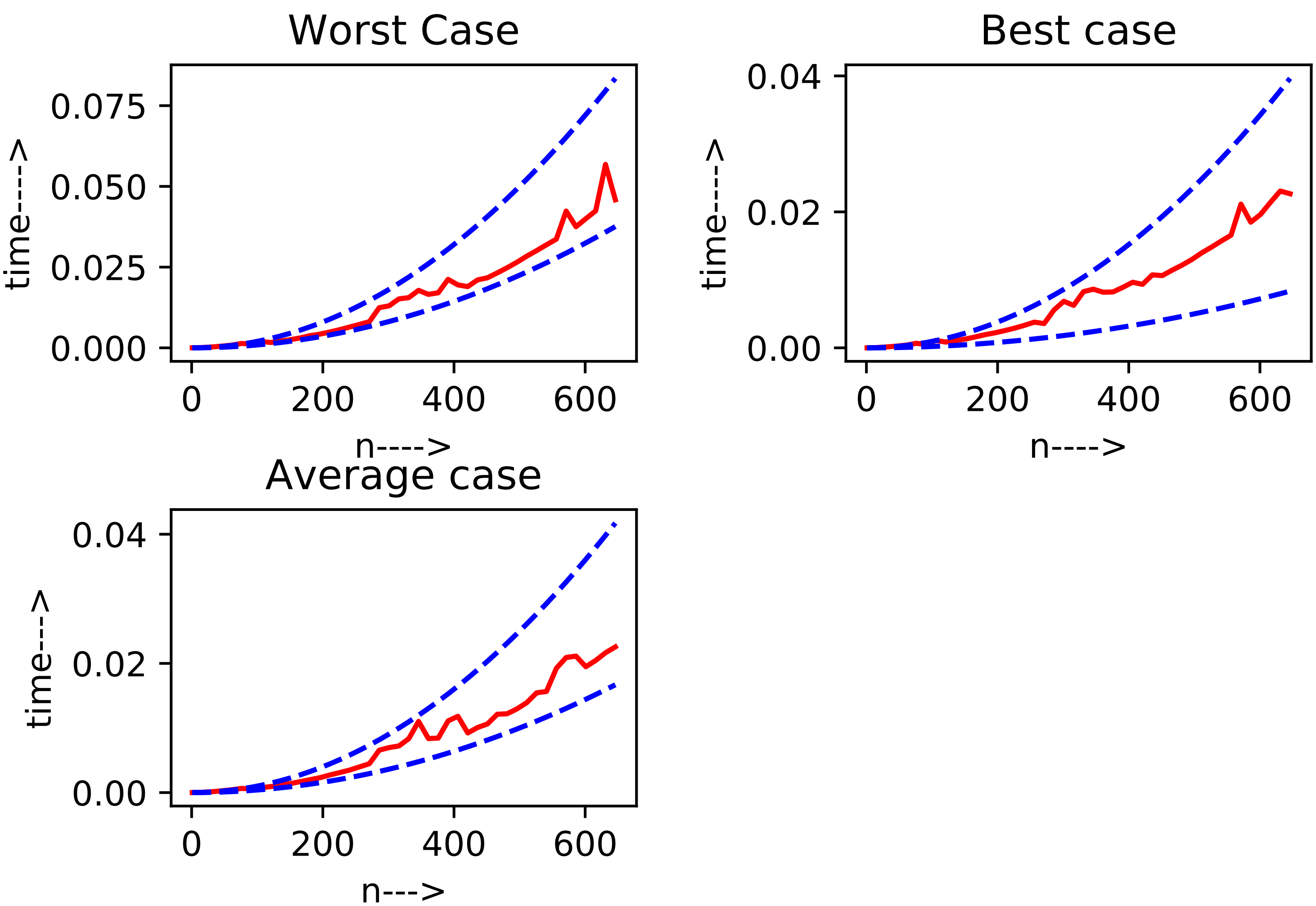

Worst Case
0.05
0.04
0.03
0.02
0.01
0.00
time---->
0
100
200
300
400
500
600
n---->

The code used to generate the graph (matplotlib.pyplot) is with the zip file.

### (e)

**Asymptotic behavior study**

For worst case, we define,
$\phi(n^2) = \{f(n) | \exists\ c_1 = 0.0000002, c_2 = 0.00000009, n_0 :\ 0 \leq c_1 n^2 \leq f(n) \leq c_2 n^2, \forall n > n_0\}$
For best case, we define,
$\phi(n^2) = \{f(n) | \exists\ c_1 = 0.000000095, c_2 = 0.00000002, n_0 :\ 0 \leq c_1 n^2 \leq f(n) \leq c_2 n^2, \forall n > n_0\}$
For worst case, we define,
$\phi(n^2) = \{f(n) | \exists\ c_1 = 0.0000001, c_2 = 0.00000004, n_0 :\ 0 \leq c_1 n^2 \leq f(n) \leq c_2 n^2, \forall n > n_0\}$
And, we plot accordingly...



Worst Case
0.075
0.050
0.025
0.000
time---->
0
200
400
600
n---->
Best case
0.04
0.02
0.00
time---->
0
200
400
600
n---->
Average case
0.04
0.02
0.00
time--->
0
200
400
600
n--->