American University of Sharjah
School of Engineering
Computer Engineering Department
P. O. Box 26666 Sharjah, UAE

Instructor: Dr. Ghassan Z. Qadah
Lab Instructor: Ms. Praveena Kolli
Office: EB2-126
Phone: 971-6-515-2352
e-mail: pkolli@aus.edu
Semester: Spring 2019

CMP305L Data Structures and Algorithms

Objectives:
The objective of this assignment:

- To understand computation of time complexity

Exercise 1
This part of the lab will give you practice of estimating the running time in your programs.

Consider the C++ code for *prgm_A, prgm_B, prgm_C, and prgm_D* as provided in appendix.

Before you run the program code, copy the table shown in fig 1 and fill in the second column(Time Complexity (before execution)). The formula should be a function of the program input *n*. The time complexity should be one of: O(1), O(n), O(n^2), O(n^3), O(n^4), etc. Review your lecture notes and the course textbook as needed.

Program	*Estimate of Time Complexity (before execution, Asymptotic complexity)*	*Estimate of Time Complexity (after execution, Equation of the trend graph)*	*Comments if column 2 and 3 differ*
prgm_A	*O(n)*	O(n)	
prgm_B	*$O(n^3)$*	$O(n^3)$	
prgm_C	*$O(n^2)$*	$O(n^2)$	
prgm_D	*O(n)*	O(n)	

Fig 1

Report:

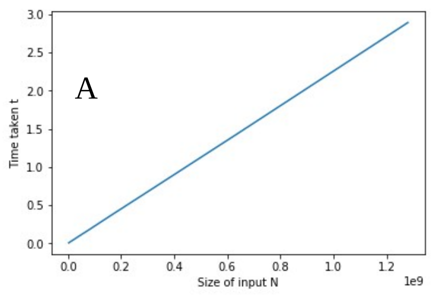
The program was compiled and run in a terminal and the output was saved in a file using `./la7 > *.txt` (where la7 is the binary file generated by the compiler, and * is a wildcard denoting one of {a,b,c,d}).

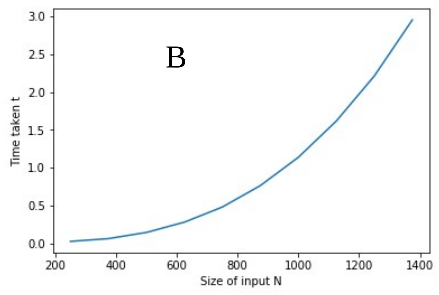
The raw output was then modified using regular expressions and a stream editor, specifically `sed -i `s/find/replace/` *.txt` where again *.txt denotes the path and name of the file being edited (as this file existed in the same directory, there was no need to supply a path). Additionally, “find” and “replace” are placeholders for the patterns we wanted to search for.

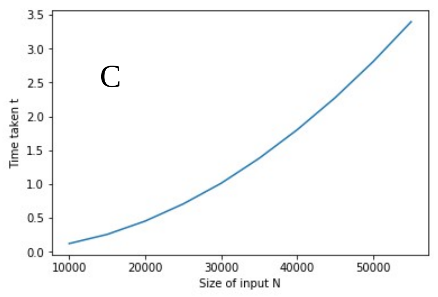
This modification was sufficient to clean the data. A manual inspection confirmed that all changes had been performed as expected. The extensions were then changed to .csv to facilitate the next part of the process, which was writing the following ten lines of code:

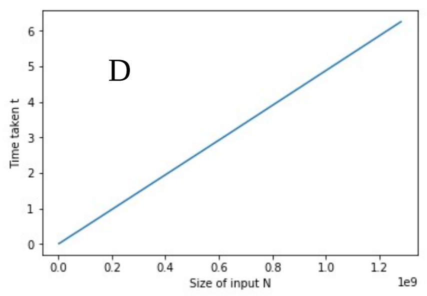
```python
import numpy as np
import matplotlib.pyplot as plt

def graph_results(file_name):
    data = np.genfromtxt(file_name, delimiter=',',
                         names=['size','time'])
    plt.plot(data['size'], data['time'])
    plt.xlabel('Size of input N')
    plt.ylabel('Time taken t')
    return
```

This basic script, written in python 3, loads the data from a file and generates a graph of it. We can see from the below output that the asymptotic behavior of the complexity matches our predictions. These estimates were conjectured based on the heuristic that the complexity was equal to the number of nested loops (provided that the indices were updated appropriately). These conjectures were in turn proven by a mathematical analysis which derived an exact expression for the number of instructions executed for any $n \in \mathbb{N}$


A
Time taken t
Size of input N
1e9


B
Time taken t
Size of input N


C
Time taken t
Size of input N


D
Time taken t
Size of input N
1e9