CS2223: Algorithms
D-Term, 2015

Homework I

Teams: To be done individually

Due date: 03/27/2015 (1:50 PM)

Submission: Electronic submission only

General Instructions

- ***Python Code vs. Pseudocode:*** Each question will explicitly state whether the deliverable is pseudocode or a Python program that the TAs will run to give you a grade.

- ***Programming Language:*** If a question asks you to write a program, then use the Python language.

- ***Submissions:*** The submission of Homework 1 must be done electronically through the blackboard site for CS2223 available from myWPI. Login to myWPI, go to CS2223 under "My Courses", then go to "Assignments", and submit your homework under "HW1". All programs plus your report (.doc or .pdf) should be zipped into a single file, and that is the file to submit.

Question 1 (Test the Order of Growth) [24 Points]

To observe the importance of algorithm efficiency and the order of growth, you are required to perform the following experiment. Assume an input of size 5,000 (denoted as N)

(***a) Linear function (O(N):*** Write a function that loops over the input items (a single loop) and print the elapsed time taken by the function in seconds.

(b) Quadratic function (O(N^2)): Write a function that loops over the input items with two-levels of nesting and print the elapsed time taken by the function in seconds. The function should have two nested loops (outer and inner) where each one starts from 0 to N.

(c) Cubic function (O(N^3)): Write a function that loops over the input items with three-levels of nesting and print the elapsed time taken by the function in seconds. The function should have three nested loops where each one starts from 0 to N.

Hint: **The actual code inside the loop is not important. So make it simple and define a variable in the beginning of your program, and then increment this variable inside the inner most loop (The example below is for two-level nesting).**

```
Sum = 0
for i in xrange(N):
     for j in xrange(N):
          Sum += j
```

Deliverables of Question 1

(1) [3 x 6 points = 18 points] A single Python program, named as "question1.py". The program should contain the three functions mentioned above. The program takes one argument with values:
-- 1 (calls the linear function),
-- 2 (calls the quadratic function),
-- 3 (calls the cubic function).

(2) [3 x 2 points = 6 points] In your report, write down the time printed by each function.

Question 2 (Linear Time Sorting) [25 Points]

The sorting algorithms discussed in class so far (i.e., the Bubble sort and Insertion sort) have $O(N^2)$ worst-case complexity. That is, given an input of size N, these algorithms may need $O(N^2)$ comparisons to sort the input list.

In this question, you are given an array of unsorted values of size 100 (the list has 100 elements), and each value is drawn randomly from the range of [0…1000] inclusive. You are required to design an algorithm to sort the given list in a linear time (that is O(N) worst-case performance).

Hint: **Make use of the fact that the range of the values is known in advance (i.e., from 1 to 1000)**

Deliverables of Question 2

(1) [10 points] In the report, write down a pseudocode of the algorithm and write down any assumptions you have (Only pseudocode is needed not a Python code).

(2) [5 points] Write a proof showing that your algorithm is correct.

(3) [5 points] Analyze your algorithm and state its Best-Case and its Worst-Case time complexity.

(4) [5 points] Analyze your algorithm and state its Best-Case and its Worst-Case space complexity.

Question 3 (Variation of Binary Search) [25 Points]

Write a variation of Binary Search where instead of choosing the middle element each time to compare with, you will choose the 1/3 element each time to compare with and then decide to move left or right.

***Hint:* In binary search, we select the middle element between index m (smaller) and n (larger) as floor of (m + (n-m)/2) = (m+n)/2. In your algorithm, your 1/3 element between index m and n is computed as floor of (m+ (n-m)/3)**

Deliverables of Question 3

(1) [10 points] Write a Python program to implement the algorithm described above. Your program should create in the beginning an array that stores only the even numbers between 1 and 200 (E.g., 2, 4, 6, …200) in ascending order.

(2) The program should take one argument which is the key K to be searched for. The program should print: (a) Whether K is found or not, (b) The number of comparisons done by the algorithm, and (c) The position at which K is found (if K exists).

(3) [15 points] In your report, analyze the above algorithm and state its Best-Case and its Worst-Case time complexity.

Question 4 (Order Of Growth) [26 Points]

Put these functions in order. Put the following functions in a list by increasing order of growth. That means, in your final list, if *f* comes before *g*, then f $\in$ O(g). If f $\in$ Θ(g), either one can come first.

$n \log_2 n$	$12\sqrt{n}$	$1/n$	$n^{\log_2 n}$
$100n^2 + 6n$	$n^{0.51}$	$n^2 - 324$	$50n^{0.5}$
$2n^3$	3^n	$2^{32}n$	$\log_2 n$

Deliverables of Question 4

(1) [10 points] In the report, write the functions in their increasing order of growth.

(2) [4 points] State which functions fall into the same Θ order (that is, have the same order).

(3) [12 points] For each pair of functions in (2) for which f = Θ(g), write a proof to show that f = Θ(g).

Bonus Question (Insertion Sort on Linked List) [10 Bonus Points]

Write a program to sort a linked list using insertion sort. Please go to the following webpage, and read the details:
https://leetcode.com/problems/insertion-sort-list/
Note: in the drop menu for programming languages, select "Python". Start working on your code in the webpage, and feel free to try out submissions. Feedbacks on the correctness of your code will be given by the webpage.

Deliverables of Bonus Question

Write a Pyhon program to implement the algorithm described above. Your program should be able to pass all test cases in https://leetcode.com/problems/insertion-sort-list/ by trying to submit your code on the webpage. You can try submitting as many times as you want on leetcode webpage.

If your program can pass all the tests on leetcode, please use this code as your solution for this bonus question and submit it together with other questions in HW1.