American University of Armenia, CSE
CS 121 Data Structures A, B, C
Fall 2019

Homework Assignment 2

Due Date: Thursday, September 26 by 23:55 electronically on moodle

Please solve the programming tasks either in Java or C++, following good coding practices (details posted in moodle).

You should submit full tested programs for all questions.

1. **(10 points)** We have an integer array the elements of which are initially increasing up to some index i, and they are decreasing starting from index i. Let's call this index *the peak*. Write an **efficient** method/function that, given such an array, determines the peak index i. Write a program that inputs the elements of an array (note that the array size is not given) and uses your method to output the peak index.

   Give big-Oh estimates for the running times of both the method and the whole program. **Briefly justify your answer.**

2. **(20 points)** We have two arrays: B and N. The B array contains elements, such as `"b3"` and `"b26"` and the N array contains elements like `"n3"` and `"n26"`. The pair `"b4"` and `"n4"` is called a matching pair, while the pair `"b10"` and `"n17"` is not. All the elements in N have a unique match from B. But there is a single extra element e' in B that doesn't have any matches in N. The input is k, where $k = |N|$, the elements of B and the elements of N. Write an **efficient** program that receives this input and prints the value e'.

   Give a big-Oh estimate for the running time of your program. **Briefly justify your answer.**

3. **(20 points)** Write a program that inputs an array arr of n elements and outputs the number of pairs of elements that are **in-order**. We will call a pair of elements $arr[i]$ and $arr[j]$ **in-order** if $i < j$ and $arr[i] < arr[j]$. The running time of your program should be $O(n \log n)$. *Inefficient solutions will receive partial points.*

4. **(15 points)** Write a method/function that, given the head node of a singly linked list of char elements, prints every second element in the list. You need to write a program to test your method/function. **Any code that does not compile will not receive any points.**

5. (20 points) Write a method/function that, given the head node of a singly linked list of integer elements, satisfies all three points below:

(a) modifies the list by removing all the nodes with prime elements,

(b) prints the sizes of the original and modified list,

(c) returns the head of the modified list.

Your method may traverse the list **only once**. You are **not allowed** to create any nodes. Test your method **properly** by using it in a program. **Any code that does not compile will not receive any points.**

6. (a) **(5 points)** Modify the **insertion sort** algorithm by replacing the main loop with recursion. Test your method **properly** by using it in a program.

(b) **(5 points)** Modify the **quick-sort** algorithm to rely on a randomly chosen pivot. Test your method **properly** by using it in a program.

(c) **(5 points)** Give an example input of length 10 on which merge-sort runs in $O(n \log n)$ time, insertion sort runs in $O(n)$ time, and quick-sort (where the pivot is the last element) runs in $O(n^2)$ time to sort in non-decreasing order of elements. Illustrate all three sorting algorithms for that example. Specify the running times of all three algorithms on the reverse of your example.