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

Homework Assignment 1

Due Date: Thursday, September 12 by 23:55 electronically on moodle

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

1. (9 points) For the code below:

```
int count_neg = 0;
int count_pos = 0;
for (int i = 0; i < a.length; i++) {
    if (a[i] < 0)
        count_neg++;
    if (a[i] > 0)
        count_pos++;
}
System.out.println(count_neg);
System.out.println(count_pos);
```

(a) Give the total number of element comparisions (i.e. involving an element of the array) as a function of the length of the array (not big-Oh notation).

(b) Can the algorithm be improved in terms of the number of comparisons? If so, show how and give the total number of element comparisons in the best and worst case analysis for your improved version of the algorithm.

(c) Give the big-Oh estimate of the running time of the original code and the improved version from (b).

Briefly justify your answers for all three parts.

2. (12 points) Order the following functions by asymptotic growth rate.

$5\log n + 48n$	$5n^3$	$3n + 2n\log n$
2^n	$15n$	$10 + \log n$
$8n^2 + 2n^3$	1250×2^{45}	$2n^2$
$\left(\frac{9}{2}\right)^n$	$60\log n + 10n$	$17n^3 + n^4$

3. (19 points) There are three types of chocolate bars with prices X, Y, and Z. Write a **recursive** program that inputs the amount W in dollars, determines and outputs in how many ways you can spend exactly W dollars to buy a set of one or more chocolate bars.

4. (20 points) Write a **recursive** program that given an array of integers, determines and outputs the number of subarrays the sum of the elements of which is 0. Note that a subarray is a **contiguous** portion of the original array. The input to the program is a natural number

$n > 0$, representing the length of the array, and the n elements of the array. *Inefficient solutions will receive partial credit.*

5. (20 points) Write a **recursive** program that inputs a natural number $n > 0$, generates and prints a square shape of asterisks (the symbol *) with n squares of decremental sizes printed within each other. The smallest square should be a single asterisk.

For example, when $n = 4$, your program should print:

```
**************
*            *
* ********** *
* *        * *
* * ****** * *
* * *    * * *
* * * * * * * *
* * *    * * *
* * ****** * *
* *        * *
* ********** *
*            *
**************
```

6. (20 points) There is a square table 4×4 filled with lowercase English letters. You can form words by starting with a letter at a random position and appending it with new letters by going up, down, left or right. The word **can** pass through a position multiple times but it is **not** allowed to leave the boundaries of the table.

Write a **recursive** program that inputs the 4×4 table and a word and prints "YES" or "NO" to indicate if the given word can be formed on the given table.