American University of Armenia, CSE
CS121 Data Structures B
Spring 2022

Homework Assignment 4

Due Date: Sunday, March 27 by 23:59 electronically on Moodle

Please solve the programming tasks using Java, following good coding practices (details are on Moodle). Don't forget to submit the corresponding assignment report.

1. **(15 points)** Write a method that, given an array list of lower case Latin letters sorted in non-increasing order, adds the letters that are missing. Similarly, write a method that does the same thing for a linked positional list. Both methods should use no additional space. Illustrate the work of both methods in a program. Specify and compare the execution times of the two methods.

2. **(20 points)** Write a class `SLLList` that implements the List ADT using a SLL as the underlying container. Make SLL's *Node*, *header* and *size* protected. Specify and discuss the execution times of each of the List ADT methods in this implementation.

3. **(15 points)** Extend your answer in question 2 with an efficient iterator that enumerates only the elements at indices divisible by 3 of a `SLLList`.

4. **(20 points)** Write a class `IntLinkedPositionalList` that represents a positional list of **ints**. Your class should extend the `LinkedPositionalList<Integer>` class. Make *header* protected. Add an iterator that enumerates the contents of a `IntLinkedPositionalList` in non-decreasing order of number of digits. What is the running time of your iterator class public methods? What is their space complexity? Briefly justify your answers.

5. **(25 points)** The task is to implement various sorting algorithms for a positional list of characters.

   (a) Implement a method *selectionSort* that given a Character positional list PL, sorts it using *selection sort.* You are **not allowed** to create any arrays or list objects. You are not allowed to swap the values (elements) at the positions. At each iteration, you should remove the position with smallest value and insert its value before the current position.

   (b) Implement a method *insertionSort* that given the first position of a Character positional list PL, sorts it using *insertion sort.* You are **not allowed** to create any arrays or list objects. You are not allowed to swap the values (elements) at the positions. At each iteration, you should remove the current position (if it is not in its correct order at the moment) and insert its value into its correct position within the sorted portion.

   (c) Implement a method *bubbleSort* that given a Character positional list PL, sorts it using *bubble sort.* You are **not allowed** to create any arrays or list objects. You are not allowed to swap the values (elements) at the positions.