American University of Armenia
CS 121 Data Structures A

Homework Assignment 4

1. (3 points) Write a Java/C++ function that, given a pointer to the first node of a singly-linked list of characters and its size, returns `true` if the characters in the list form a palindrome and `false` otherwise. Your algorithm should traverse the list **only once** and should use a stack. Test your function on a few examples and discuss the results.

2. (2 points) The main drawback of the array-based queue implementation (from the slides/ textbooks) is its bounded capacity. Still using arrays for storage, modify this implementation to eliminate the drawback. Test your implementation on an example illustrating the improvement and discuss the results. *Hint:* We have discussed a similar technique for [array lists][vectors].

3. (2 points) The main drawback of the array-based stack implementation (from the slides/ textbooks) is its bounded capacity. Give an implementation of the stack ADT using an array list for storage. The array list, in turn, should rely on [dynamic][extendable] arrays. Test your implementation on an example illustrating the improvement and discuss the results.

4. (5 points) Write two Java/C++ implementations for the deque ADT: (1) using arrays with no capacity bounds and (2) using doubly linked lists. Your implementations should be complete, i.e. housekeeping functions, proper error-handling and documentation should be provided. List, explain and compare the running times of the deque operations and the space usage of both implementations.

5. (3 points) Write a Java/C++ program that contains a full implementation of the [positional list][list] ADT using a doubly-linked list and in the main method performs a series of operations on an initially empty [positional list][list] storing integers. Your implementation can be based on the code given in the textbooks and needs to be complete with housekeeping functions, proper error-handling, iteration mechanisms. Explain step-by-step the execution of the sample main method, also discussing running times and space requirements. Explain specific implementation details.

6. (2 points) Using your implementation of the [positional list][list] ADT from Task 5, write a Java/C++ program that first constructs a list L_1 of randomly generated integers, then makes a second list L_2 containing the elements of L_1 that are divisible by 2, then another list L_3 containing the elements of L_2 that are divisible by 3. What is the common property of all the elements in L_3? Your program should rely on iterators. What is the running time of your program as an expression of the length of L_1?