Advanced Programming Techniques

Aurel Vlaicu University Arad

Practical Session #5

Exercise 1. Find if in a sequence of characters the parantheses match. What data structure could you use in this case? What implementation option do you prefer? What is the complexity of the proposed solution?

Exercise 2. Check if two strings are anagrams of each other. You may choose one option below. Which one of the options below is more efficient?

- sorting the strings and comparing them
- count the frequency of each character in the input strings

Exercise 3. You are given a string and you are asked to reverse the order of the words given in the string. Words are separated using dots. The output should have the same structure.

What data structure presented in the course may be useful here?

You may use lists in Python in your implementation. The procedure `.append()` allows you to add words to the end of the list.

Exercise 4. You are given an unsorted sequence of n elements. Check if there exist two elements with a given sum. Challenge: solve the problem in linear time.

Exercise 5. (a) Write an algorithm for finding a peak in a one dimensional array. (A peak verifies $A[i-1] \leq A[i] \geq A[i+1]$. Implement the efficient solution proposed in the course.

(b) Write an algorithm for finding a peak in a two dimensional $n \times n$ array. Propose a solution in $O(n \log n)$ time

Exercise 6. (a) Write an algorithm for finding the maximum subsequence sum in an array: find $1 \leq i \leq j \leq n$ such that

$$A[i] + ... + A[j]$$

is maximal. Your solution should be **sub-quadratic** (more efficient than the double loop).

(b) Write a code which solves the problem posed in the course: find the best buy/sell moments in a given time series of stock prices.

Exercise 7. Given a set of activities, along with the starting and finishing time of each activity (positive integers), find the maximum number of activities performed by a single person assuming that a person can only work on a single activity at a time.

Exercise 8. Given a set of positive integers and an integer k, check if there is a non-empty subset that sums to k.