Data structures and algorithms (2018/19) Written exam 11. svečana 2019

This written exam must be taken individually. Any and all literature may be used while taking this test. In your answers be precise, and: (i) answer the questions *as they were asked*; and (ii) answer *all* tasks – if you will be answering to all tasks you might get bonus points.

Time: 75 minutes.

We wish you a lot of success - veliko uspeha!

NALOGA	TOČK	OD TOČK	NALOGA	TOČK	OD TOČK
1			3		
2			4		

IME IN PRIIMEK: ______________________

ŠTUDENTSKA ŠTEVILKA: ______________________

DATUM: ______________________

PODPIS: ______________________

1. naloga: Basics. Let us define a function `contained(p, t)`, which finds if all characters of string `p` are contained in string `t` in the same order. In addition, we define a function `notIntersection(p, t)`, which returns all characters that are not contained in `p` and in `t` at the same time. Moreover, for both functions let $|p| = m$ and $|t| = n$. Suppose that we have two arrays $\mathtt{t} = [\mathtt{p}, \mathtt{o}, \mathtt{s}, \mathtt{t}, \mathtt{a}, \mathtt{j}, \mathtt{a}]$ and $\mathtt{p} = [\mathtt{s}, \mathtt{a}, \mathtt{t}]$, then `contained(`$p$`, `$t$`)` returns FALSE and `notIntersection(`p`, `t`)` returns $\{\mathtt{p}, \mathtt{o}, \mathtt{j}\}$.

VPRAŠANJA:

A) (i.) Write down the function `contained(p, t)` for an arbitrary p and an arbitrary t. (ii.) What is the time complexity of your algorithm? Justify your answer.

B) (i.) Write down the function `notIntersection(p, t)` for an arbitrary p and an arbitrary t. (ii.) What is the time complexity of your algorithm? Justify your answer.

   HINT: Consider that characters of strings p and t can be arbitrary numbers.

C) Suppose that characters in both strings are from a finite set $\Sigma = \{a_1, a_2, ..., a_s\}$, where $s \ll n, m$. (i.) What does your function `notIntersection(p, t)` look like now? (ii.) What is the time and space complexity of your new function?

2. naloga: Tree walks. Our friend Peter Puzzle has heard that we have different tree walks: a) a preorder tree walk prints the root before the values in subtrees; b) an inorder tree walk prints the root of a subtree between printing the values in its subtrees[1]; and c) a postorder tree walk prints the root after the values in its subtrees.

VPRAŠANJA:

A) Peter is wondering if an inorder tree walk and a postorder tree walk can be the same on a binary tree. (i.) Write down a binary tree on ten vertices that has the same inorder and postorder tree walk. (ii.) Now, Peter is wondering what is the maximal possible number of vertices in a binary tree that has the same inorder and postorder tree walk. Find the answer and justify it.

B) Besides the described walks, Peter has found the definition of a tree walk by layers: first the root is visited, then its children, then grandchildren and so on. (i.) For your tree from the first part of the question of this problem write

[1] In the case of k-ary trees the root is even visited multiple times.

down the vertices by layers. (ii.) What do you notice? (iii.) Write down an algorithm that prints a k-ary tree by layers.

C) (i.) What is the time complexity of your algorithm? Justify your answer. (ii.) Describe a binary tree that has the same inorder tree walk, preorder tree walk and tree walk by layers. Justify your answer.

3. naloga: In Butale, residents decided to paint the fence around their village church. They marked the starting point of the fence and then opened a call for every resident from Butale to tell from which meter to which meter the fence should be painted. So, Luka Kratkohlačnica proposed to paint between 4 and 8 meters, Gregor Copatka between 7 and 12 meters, and Benda Cigan between 22 and 33 meters. Hence, there were two stripes of paint, namely, between 4 and 12 meters and between 22 and 33.

VPRAŠANJA:

A) In the end, the community proposed the following stripes for painting: 68 and 76, 55 and 52, 17 and 79, 36 and 56, 47 and 38, 29 and 18, 20 and 70, 7 and 5, 55 and 49, 83 and 26, 88 and 10, 66 and 19, 5 and 2, 50 and 82, 80 and 82, 75 and 39, 10 and 9, 85 and 15, 39 and 12. What is the overall length of the painted fence?

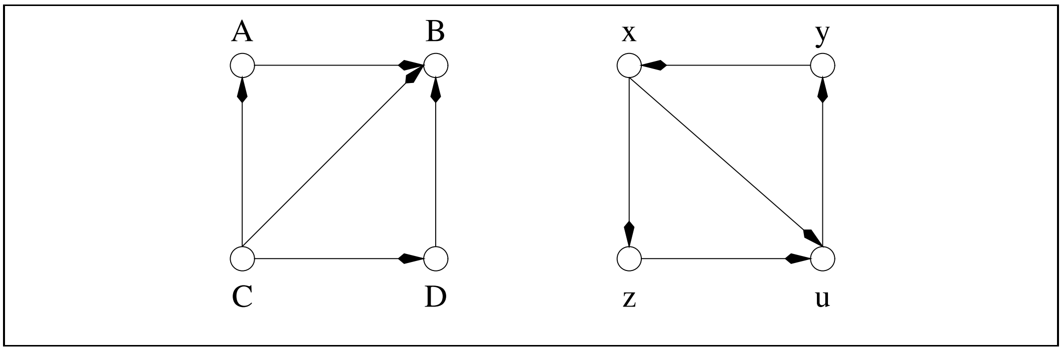
B) Write down the algorithm which takes a number n, representing the number of residents, and n pairs of numbers, and finally returns the length of the painted fence.

C) What is the time and space complexity of your algorithm? Justify your answer.

4. naloga: Graphs. At the lectures we considered the definition of a *topological sorting of a directed graph*. That is, a directed graph $G(V, E)$ can be sorted topologically, if we can order its vertices $v \in V$ along a horizontal line so that all directed edges $(u, v) \in E$ go from left to right. In Figure 1, we have two directed graphs $G_L(V_L, E_L)$ (the left graph) and $G_D(V_D, E_D)$ (the right graph).

VPRAŠANJA:

A) (i.) Represent $G_L(V_L, E_L)$ and $G_D(V_D, E_D)$ first by adjacency matrix and then by adjacency list (ii.) For $G_L(V_L, E_L)$ and for $G_D(V_D, E_D)$ find out, whether it can be sorted topologically or not. If the answer is affirmative, write down the corresponding topological order, otherwise explain why you can not sort it topologically.



Slika 1: Directed graphs.

B) Suppose that we have some directed graph $G(V, E)$. (i.) Write down an algorithm that finds whether $G(V, E)$ can be sorted topologically or not. (ii.) What is the time and space complexity of your algorithm? Justify your answer.

HINT: If your algorithm is recursive, note that the complexity depends on the recursion depth.

C) Suppose that a directed graph $G(V, E)$ can not be sorted topologically. The later can be changed if we reverse a direction of some edge or if we remove some edge – both described operations are called *fixing*. (i.) Write down an algorithm which uses as small number of fixings as possible to change the graph $G(V, E)$ so that it can be sorted topologically. (ii.) Justify the correctness of your algorithm. (iii.) What is the time and space complexity of your algorithm? Justify your answer.