Algorithms and Data Structures, Academic Year 2016/2017

International Bologna Master in Bioinformatics

June 27, 2017

Please complete the following exercises by applying the concepts that have been illustrated to you during the classes. The score associated with each exercise and the expected time for completion is reported in the first line. It is possible to keep paper version of notes, but no electronic device is allowed.

Exercise 0 (2 points): write your name and surname in the first row of all the sheets you use.

Name:__________________________________ Surname:_______________________________

Exercise 1 (35 points, 60 minutes): Given a sequence S of n integers, stored into an array, one wants to find the length of the *Longest Increasing Subsequence* (*LIS*) of S (e.g. if S = 9, 15, 3, 6, 4, 2, 5, 10, 3, then the solution is 4, because the *LIS* is 3, 4, 5, 10). Solve the problem by a *dynamic programming* algorithm, defining first the recurrence relation giving the optimal sub-structure property, and then writing its pseudo-code and analyzing its complexity. (*Suggestion*: define $D[i]$ as the length of the *LIS* ending with element $S[i]$ and $P[i]$ as the position of the element preceding $S[i]$ in such an *LIS*). Finally, run (by hand) the algorithm on the sequence given above as an example.

(use additional sheets for this exercise, including the back of this sheet)

Name:________________________________ Surname:_____________________________

Exercise 2 (18 points, 20 minutes): given the following sequences of visited nodes of a generic **binary** tree whose nodes are all **distinct**, write the tree itself in the space below. An “x” is a placeholder for an unknown number. Note: the solution is not unique.

pre-order visit: x 30 6 7 32 x x 10 3
in-order visit: x x x 13 x 16 21 3 10
post-order visit: 7 6 30 x 15 x x x 13

Name:__________________________________ Surname:_______________________________

Exercise 4 (25 points, 15 minutes): Consider the following function and assume that it is always called with **a** being an array of integers, **0 <= l <= u**, and **u <= size(a) + 1**. Also assume **k** to be a positive integer.
Compute the time and space complexity of **f** as a function of **k**. Then determine the mathematical expression implemented.

Hint: to understand what the function does, imagine to call it initially with **l = 0** and **u = size(a) + 1**.
Hint: observe that for certain values of **u,l,k**, not all elements of the array **a** are accessed.

```
function f(int a[], int l, int u)
begin
    int res := 1
    int step := (u – l) / k   // this is integer division, rounded below; e.g. 11 / 3 = 3
    if u - l < k then
        for j := l to u – 1 do
            res := res * a[j]
        end
    else
        for j := 0 to k - 1 do
            res := res * f(a, l + step * j, l + step * (j+1))
        end
    return res
end
```

Name:___________________________________ Surname:________________________________

Exercise 5 (20 points, 25 minutes): Consider the string "m a m m a". Write (by hand) its corresponding:

- Suffix trie;
- Suffix tree;
- Suffix array;
- Burrows-Wheeler transform;
- LF mapping.