Model and Languages for Bioinformatics, AY 2018-19

Second intermediate test.

Available time:
1 hour and half for the three Python exercises.
2 hours if also the Prolog exercise is solved (the Prolog exercise
should be solved only in case you want to pass the full exam,
or improve the result of the first intermediate test)

29/5/2019

FIRST AND FAMILY NAME:

Email:

Exercise 1)
Write a Python function which takes in input two rectangular matrixes M, and M1 of integer values, having the same size NxM, and which returns a boolean value True if and only if there exist one column from M and one from M1 such that the sum of the values in each of the two columns is the same.

For instance if M =[[3,2,1],[4,0,5]] and M1 =[[4,0,1],[6,6,6]] the returned value is True as the sum of the values in column 0 of M gives the value 7, as the sum of the values of column 2 in M1.
If M1= [[4,0,5],[1,1,0]] the returned value is False.

Exercise 2)
Write a Python function, which, given two input binary trees of integers T1 and T2, defined following
class Tree:

```
    def __init__(self, elem=None, left=None, right=None):
        self.elem = elem
        self.left = left
        self.right = right
```

returns a boolean value True if and only if all values in T1 appear at least twice in T2.
The value False is returned otherwise.
So, for instance if T1= Tree(11,Tree(9,Tree(2))),
T2= Tree(9,Tree(2),Tree(9,Tree(2,Tree(11),Tree(5)),Tree(11))), returns True.

Exercise 3)
Consider the module “re” for defining Python regular expressions.
Write a Python function which, given a string S and a positive integer N,
checks if S contains occurrences of a substring which begins with one initial
substring ‘xx’, contains the substring ‘yy’, ends with a substring 'zz'
and has length >N.
The function has to print all occurrences of such substrings in S.
For instance if s = ‘abxxa1yydfczzbxxbbbyyxxzzcaaa12cccyy’, and N==8, it will print
the two following substrings: ‘xxa1yydfczz’ and ‘xxbbbyyxxzz’.

Exercise 4) (optional, in Prolog language)

Consider a binary tree of integers represented by the notation
we have seen in class. Namely, the constant nil represents an empty tree,
while a non empty tree is represented by the term t(N,Tree1,Tree2),
where “t” is a ternary function, N is the root (an integer value),
and Tree1 and Tree2 represents (inductively) the left and right subtrees.
Write a Prolog program NoccurrencesInTree(T,N,L),
which given one binary search tree T
and a positive integer N, returns the list L of the values which belong to T,
and which occur exactly N times in T.

So, for instance
NoccurrencesInTree(t(-1,t(3,nil,nil),t(4,nil,t(-1,nil,t(3,nil,nil)))),2,[-1,3]) is true.