Assignment No. 7: Multi-way Trees

Transformations between different representations

Allocated time: 2 hours

Implementation

You are required to implement **correctly** and **efficiently** *linear time* transformations between three different representations for a multi-way tree:

R1: *parent representation*: for each key you are given the parent key, in a vector.
R2: *multi-way tree representation:* for each node you have the key and a vector of children nodes
R3: *binary tree representation:* for each node, you have the key, and two pointers: one to the first child node, and one to the brother on the right (i.e. the next brother node)

Also, you are required to write a *pretty print* procedure on R3, which performs a preorder traversal on the binary representation and outputs the tree in a friendly manner (see the image on the next page for an example).

Therefore, you are given as input a multi-way tree in the *parent* representation (*R1*). You are required to implement *T1*, which transforms the tree to a *multi-way* representation (*R2*), then *T2*, which transforms from the *multi-way* representation to the *binary* representation (*R3*). Then, on the *binary* representation, you are asked to write a pretty print procedure (using a pre-order traversal).

You should be able to design the necessary data structures by yourselves. You may use intermediate structures (i.e. additional memory).

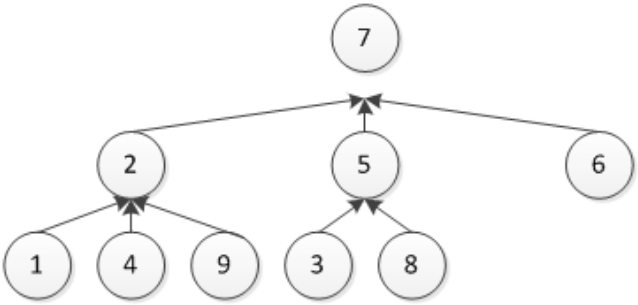
Evaluation

You should run your algorithms on a sample input tree (you may use the one in the example provided on the next page). Output (in a readable manner) the tree in each of the three representations (for R1 simply print the parent vector; for R3 it is enough to call the pretty print procedure).

Explain what data structures you employed for the *R2* and *R3* representations.
You should assess the efficiency of your methods: i.e. do your transformations run in *O(n)*?
Also, explain the necessity for any additional memory employed by your algorithms.

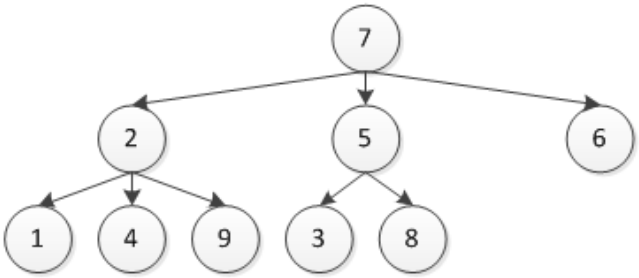
Input (R1): Π = {2, 7, 5, 2, 7, 7, -1, 5, 2}

1 2 3 4 5 6 7 8 9



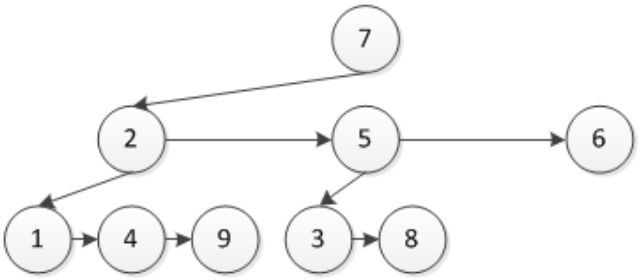
R1: parent representation

T1:parent -> multi-way



R2: multi-way representation

T2: multi-way -> binary



R3: binary representation

PP: prrety_print (binary)

```
7
    2
        1
        4
        9
5
        3
        8
    6
```

Pretty print