1 Assignment No. 6: Multi-way Trees

Transforms between different representations **Allocated time:** 2 hours

1.1 Implementation

1. You are required to implement **correctly** and **efficiently** *iterative* and *recursive* binary tree traversal. You may find any necessary information and pseudo-code in your course and seminar notes.

2. Moreover, the **correct** and **efficient** implementation of *linear* complexity algorithms is required for transforming multi-way trees between the following representations:

    1. **R1**: *Parent representation*: for each index, the value in the vector represents the parent's index, e.g.: $\Pi = \{2, 7, 5, 2, 7, 7, -1, 5, 2\}$
    2. **R2**: *Multi-way tree representation*: each node contains the key and a vector of child nodes.
    3. **R3**: *Binary representation*: each node contains the key and two pointers, one to the first child and the second to the right sibling (e.g., the next sibling).

Therefore, you need to define transformation **T1** from the *parent representation* (**R1)** to the *multi-way tree representation* (**R2**), and then the transformation **T2** from the *multi-way tree representation* (**R2**) to the *binary representation* (**R3**). For all representations (**R1**, **R2**, **R3**), you need to implement the Pretty Print (**PP**) display (see page 2).

Define the data structures. You can use intermediate structures (e.g., additional memory).

1.2 Requirements

1.2.1 Implementation of *iterative* and *recursive* binary tree traversal in *O(n)* and *with constant additional memory* (3p)

You will have to prove your algorithm(s) work on a small-sized input.

1.2.2 Implementation of transforms between different representations

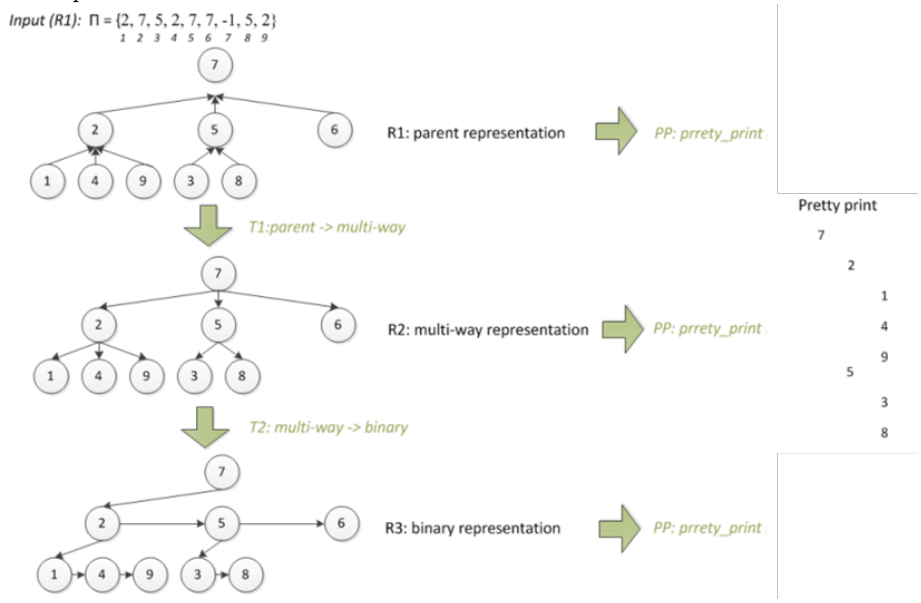
1.2.3 Correct implementation for Pretty-print for R1 (2p)

1.2.4 Correct implementation for *T1 (from R1 to R2)* and pretty-print for *R2* (1p) + *T1* in linear time (1p)

1.2.5 Correct implementation for *T2 (from R2 to R3)* and pretty-print for *R3* (2p) + *T2* in linear time (1p)

The correctness of the algorithms should be demonstrated using the example $\Pi = \{2, 7, 5, 2, 7, 7, -1, 5, 2\}$.

Use Pretty Print for all three representations. *Each representation (R1,R2,R3) should have a pretty print of its own with a different implementation but with the same print.*



Analyse the time and space efficiency of the two transformations. Did you achieve *O(n)*? Did you use additional memory?