Mathematics u34605 (algorithms and data structures) Michaelmas 2020

October 6, 2020

1 First assignment, roughly 2 weeks to complete

Use the `submit-work` program to submit your C program (nothing else).

Write a program which takes a list of binary trees like

```
7 0 3 4 2 1 6 5
20 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19
```

Data is given as a sequence of node-counts followed by the *inorder rankings* indexed by *postorder rankings*.

Thus, in the first tree, the root, which comes last in postorder, has inorder rank 5. Unless some mistake has been made, the first tree is the same as shown in Section 6 of the notes.

Sample data is in the file `examples-p2i` on the web page.

Your output should list the nodes, *and also* calculate and print the preorder, inorder, and postorder rank. For example:

```
 0 3 4 2 1 6 5
Tree 7 nodes
Node  0 lchild  1 rchild  6  pre:  0 in:  5 post:  6
Node  1 lchild  2 rchild  3  pre:  1 in:  1 post:  4
Node  2 lchild  - rchild  -  pre:  2 in:  0 post:  0
Node  3 lchild  - rchild  4  pre:  3 in:  2 post:  3
Node  4 lchild  5 rchild  -  pre:  4 in:  4 post:  2
Node  5 lchild  - rchild  -  pre:  5 in:  3 post:  1
Node  6 lchild  - rchild  -  pre:  6 in:  6 post:  5

-----------------
 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19
etcetera
```

Implementation suggestions.

This structure is enough to do the assignment:

```
typedef struct NODE { struct NODE * lchild, * rchild;
        int pre_rank, in_rank, post_rank ; } NODE;
```

Nodes can be created by

```
(NODE*) calloc (1, sizeof (NODE) );
```

Following the web notes, you should write a recursive function to create a subtree, actually returning a pointer to NODE, which takes arguments size of subtree, minimum rank in inorder, maximum rank in postorder, and array giving inorder ranks indexed by postorder rank. If size is ≤ 0 it returns NULL.

Also write a recursive routine to install the ranks. It should use three 'quasi-reference' pointer-to-int variables pre, in, post; when called from main() they should be initialised to zero, and they should be updated during the installation process.

Finally, you need a recursive routine to print details of the tree constructed, as shown above. Note that the nodes are identified by their preorder ranks.