# Binary Trees.

For this assignment you have to write a program that takes 3 command line arguments as follows:

1. The filename for the input file
2. The filename for the output file
3. A command to execute

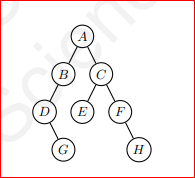
An example could be (given our program is called trees):

trees input.dat output.dat height

In the input file there will be given a tree as in the first lab example:

ABD\*G\*\*\*CE\*\*F\*H\*\*

Which would result in the following tree:



Besides reading and constructing the tree, you have to implement the next functions:

* **height** : prints the height of the tree **10p**
* **preorder**: prints the nodes of the tree in preorder (recursively print: currentNode\_key, left\_node, right\_node) **10p**
* **inorder**: prints the nodes of the tree in order(recursively print: left\_node, currentNode\_key, right\_node) **5p**
* **postorder**: prints the nodes in postorder ( left\_node, right\_node, currentNode\_key) **5p**
* **leaves**: prints the number of leaves in the three **10p**
* **swap**: will swap the children of a node whose key is given in the second line of the output file and print the nodes **inorder. 10p.**

**EX input:**

**ABD\*G\*\*\*CE\*\*F\*H\*\***

**G**

* **swap children of node G**

**THE FUNCTIONS UP UNTIL NOW ARE MANDATORY IN ORDER TO PASS THIS ASSIGNMENT.**

**Optional functions:**

* **list:** after building the tree, transform it into a doubly linked list and print the list **10p**
* **postfix:** in this case, the input file will represent an operation written in postfix form and you have to print the infix form of the operation on the first line, including parentheses for the order of operations, and on the second line the result of the operation (Supported operations: +, -, /, \*). **15p**

**EX: input**

**1 2 + 4 3 5 6 + - \* +**

**Output:**

**( ( ( ( 6 + 5 ) – 3 ) \* 4 ) + ( 2 + 1 ) )**

**35**

**No warnings 10p**

**No mem leaks 10p**

**No cppcheck warnings/errors 5p**

**Example of use cases:**

|  |  |  |
| --- | --- | --- |
| **ARGUMENTS** | **Input file** | **Output file** |
| **Input.dat output.dat height** | ABD\*G\*\*\*CE\*\*F\*H\*\* | **4** |
| **Input.dat output.dat preorder** | ABD\*G\*\*\*CE\*\*F\*H\*\* | **ABDGCEFH** |
| **Input.dat output.dat inorder** | ABD\*G\*\*\*CE\*\*F\*H\*\* | **DGBAECFH** |
| **Input.dat output.dat postorder** | ABD\*G\*\*\*CE\*\*F\*H\*\* | **GDBEHFCA** |
| **Input.dat output.dat leaves** | ABD\*G\*\*\*CE\*\*F\*H\*\* | **3** |
| **Input.dat output.dat swap** | ABD\*G\*\*\*CE\*\*F\*H\*\*  D | **GDBAECFH** |
| **Input.dat output.dat list** | ABD\*G\*\*\*CE\*\*F\*H\*\* | **ABDGCEFH** |
| **Input.dat output.dat postfix** | 1 2 + 4 3 5 6 + - \* + | **( ( ( ( 6 + 5 ) – 3 ) \* 4 ) + ( 2 + 1 ) )**  **35** |