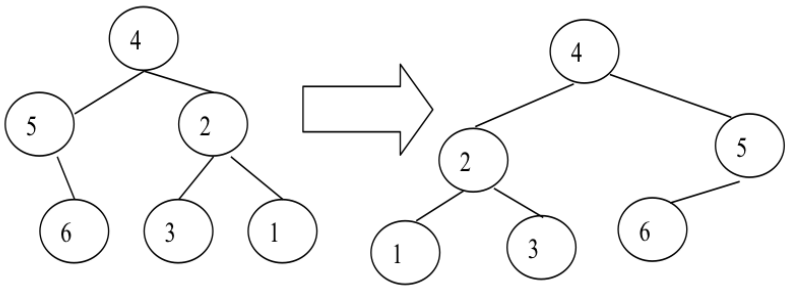
Binary Trees

---

1. Write a recursive function mirrorTree() that will modify a binary tree so that the resulting tree is a mirror image of the original structure.



You should not create any intermediate or temporary trees. The function accepts a single parameter: a pointer to the root note of the binary tree to be mirrored.

```
void mirrorTree(BTNode *node);
```

2. Write a C function `printSmallerValues()` that accepts a pointer to the root node of a binary tree and prints all integers stored in the tree that are smaller than a given value `m`. The function prototype is given as follows:

```
void printSmallerValues(BTNode *node, int m);
```

3. Write a function smallestValue() that returns the smallest value stored in a given tree. The function accepts a pointer to the root of the given tree. You should determine the correct function prototype.

4. Write a recursive function hasGreatGrandchild() that prints the values stored in all nodes of a binary tree that have at least one great-grandchild. The function accepts a single parameter: a pointer to the root note of the binary tree.

```
int hasGreatGrandchild(BTNode *node);
```

Hint: Determine the common property shared by nodes with great-grandchild nodes, and write a recursive function that computes that property.