

Exercise 2.2: Suppose your friend sent you the following file, can you construct the original binary tree that it represents?

```
treeFile.txt
1 01A01B01C1D
2
```

// Draw the original tree here:

Exercise 2.3: Using the pseudocode from 2.1, write a function that prints the translated Tree.

```
huff_tree.h
1 #pragma once
2 class Tree{
3     public:
4         struct Node {
5             char value;
6             Node* left;
7             Node* right;
8             Node(char value = 0, Node *left = NULL,
9                 Node *right = NULL):
10                value(value), left(left), right(right) {}
11         };
12         Node* root;
13         void translate(const Node* subRoot) const;
14     };
15
```

```
huff_tree.cpp
1 #include "huff_tree.h"
2 void Tree::translate(const Node* subRoot) const {
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17 }
```

In the programming part of this lab, you will:

- Complete the implementation of the HuffmanTree class:
- Implement the buildTree() function
- Implement the writeTree() and readTree() functions for writing and reading a binary tree from a file.
- Have fun encoding and decoding!

As your TA and CAs, we're here to help with your programming for the rest of this lab section! 😊