CS302

Assignment 5: Hungry for (Red) Apples

## Description

So...what do you think? You're fired. What? But this idea was tested in a state of the art simulation. Well then it was a terrible simulation, get out!

Well that was an unfortunate situation for Jerry. But maybe he might be able to win them over because for some reason, there is an apple tree (a binary tree), with a set of nodes (apples), and we want to be able to retrieve all the red apples in the fastest time possible. This has to be in the simulation, why would this even be an issue for this company? Anyway, let's take a look at the class definition

```
class binTree
{
public:
  struct binTreeNode
  {
    bool apple;
    binTreeNode * left;
    binTreeNode * right;
  };

  binTree();
  binTree(std::vector<bool>);
  ~binTree();
  int minTime();
private:
  void destroyTree(binTreeNode*);
  void buildTree(binTreeNode * r, std::vector<bool>, int);
  int minTime(binTreeNode * r, int);
  binTreeNode * root;
};
```

Each member of the class contains/performs the following

- struct binTreeNode - each element in the binary tree, contains a pointer to the left and right child, and the bool apple field, for a node if this field contains true then this node is a red apple, if this field contains false then this node is not a red apple
- binTreeNode * root - the root pointer for our binary tree

- `binTree::binTree()` - default constructor that sets the root with `NULL`
- `binTree::binTree(std::vector<bool> apples)` - constructor that takes in a vector of apples, each element is `true` or `false` denoting a red or non red apple, you first allocate a node to the root, and set its apple field with the value stored in apples[0], the root is a leaf at this moment so make sure you set the rest of the fields of the root node as well, then you call the function buildTree and pass in the root node, the vector of apples, and the initial index 0, then the tree will be magically built
- `void binTree::buildTree(binTreeNode * r, std::vector<bool> apples, int i)` - this function is where all the magic happens that builds the tree from the array given, this function needs to compute the left index and right index given index `i`, and then possibly build a left and right child, assign the value from apples vector to the left and right child node, and the recursively build the left and right side
- `void binTree::destroyTree(binTreeNode * r)` - function that deallocates a binary tree rooted at node `r`, deallocates the left subtree, then the right subtree, then deallocates `r`
- `binTree::~binTree()` - deallocates the binary tree, this would call `destroyTree(root)`, and then this function magically deallocates the binary tree
- `int binTree::minTime()` - function that simply contains `return minTime(root, 0)`
- `int binTree::minTime(binTreeNode * r, int time)` - function that counts the minimal amount of moves needed to retrieve all the red apples in the binary tree staring from the root. Every forward move to a child causes the time to increment by 1, every movement back to its parent also increments time by 1, this might be a bit tricky to figure out, but if a path leads to a leaf where no red apples where found along the way, then you do not want to maintain those time counts.

## Contents of Main

Main is given to you, so you must not modify the public functions or else main will not work, do not upload your copy of main, there will be a main.cpp on code grade that will be linked to your binTree files.

## Specifications

- No memory leaks
- You may add non member functions or private functions but no public member functions added

## Sample Run

```
$ ./a.out

Enter apple tree file: tree01.txt
Minimum Time Needed: 6

$ ./a.out

Enter apple tree file: tree02.txt
Minimum Time Needed: 12

$ ./a.out

Enter apple tree file: tree03.txt
Minimum Time Needed: 10
```

```
$ ./a.out

Enter apple tree file: tree04.txt
Minimum Time Needed: 0
```

## Submission

Submit binTree.h (and possibly binTree.cpp if you have your implementation separated) to code grade by the deadline

## References

- Supplemental Video https://youtu.be/URKxiSdYDYg
- Link to the top image can be found at $https : //www.inverse.com/article/34066 - rick - and - morty - jerry - hungry - for - apples - ad - jerry - horse - hospital$