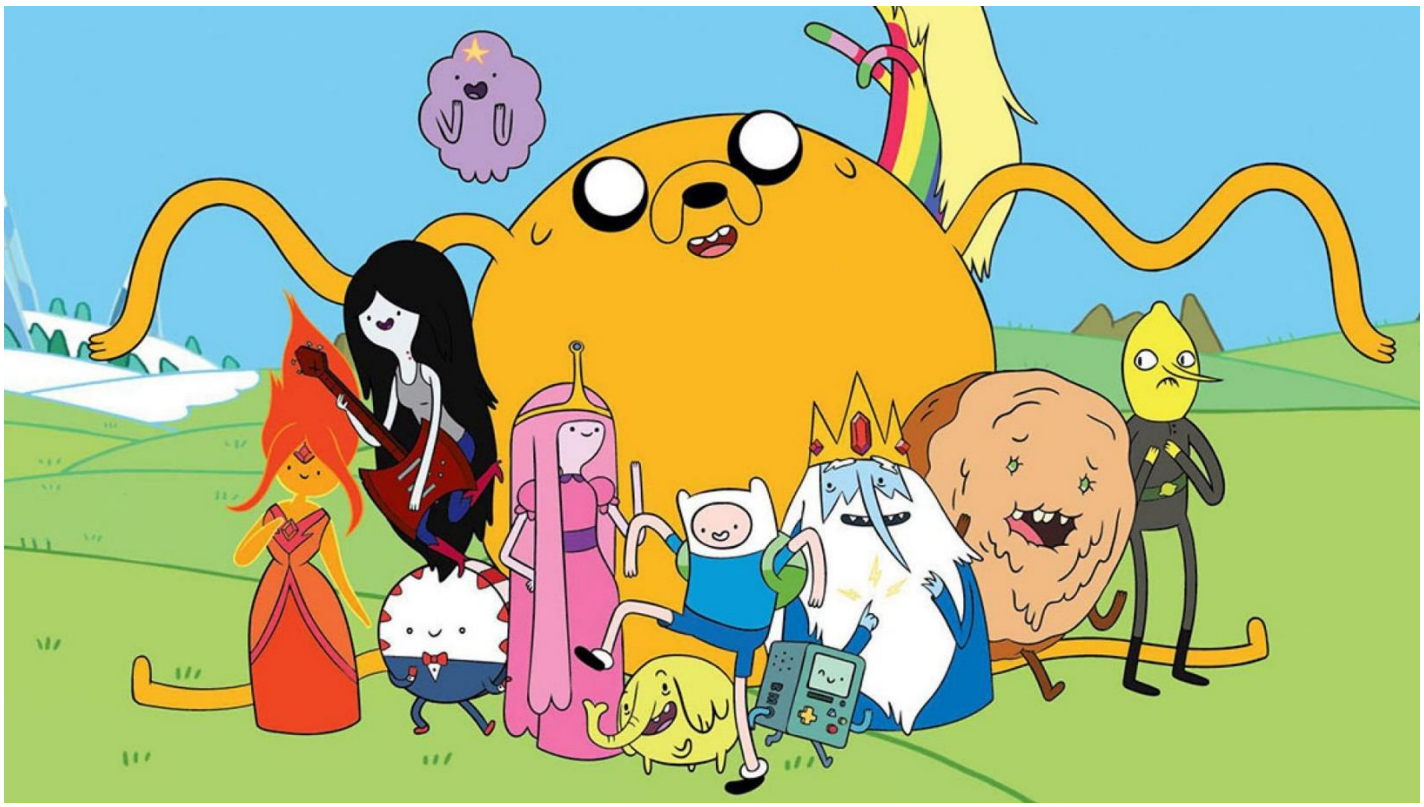
LAB 10 / CSC1310

BINARY TREES

ADVENTURE TIME CAST



Write your own version of a class template that will create a binary tree that can hold values of any data type.

Then, design a class for the Adventure Time cast called **AT_Cast** class (name the file **AT_Cast.h**) that holds the Cast ID Number and name of a cast member.

Next, use the binary tree class template to implement a binary tree whose nodes hold an instance of the AT_CAST class. The nodes should be sorted on the CAST ID number. Print out the cast (sorted by cast ID number) in three different ways – in-order traversal, pre-order traversal, and then post-order traversal.

AT_CAST.H

AT_Cast class should hold the following (private) Adventure Time information:

- **Cast ID Number**: an integer
- **Cast Member Name**: a string

This class should also implement the following public functions:

- Constructor (setting the id & name)
- setID
- setName
- getID
- getName
- overloaded < operator (code & explanation below)
- overloaded << operator (code & explanation below)

OVERLOADED OPERATORS & FRIEND FUNCTIONS

Overloading operators is where you can make an operator behave differently than its normal use (you are overloading it's normal way of behaving with your own way). Operator overloading is similar to function overloading, where you have many versions of the same function differentiated by their parameter lists.

In this program, you are going to have to compare if an AT_Cast object is smaller than another AT_Cast object in order to insert a node in the correct place in the BinaryTree class.

Normally, you can't use relational operators to compare two entire objects. However, we can overload the less-than '<' operator to say that for this class, when we use the less-than '<' operator, we want it to check if the castID attribute for this object is smaller than the castID of the for the object we are comparing to. This operator will then return true or false.

The following function should be a public function of the AT_Cast class:

```
bool operator< (const AT_Cast& member)
{
    return this->castID < member.castID;
}
```

You also want to be able to print out an AT_Cast object using the << operator in the BinaryTree class. Normally you can't print out an object with just the stream insertion operator '<<'. So, we have to overload this operator. What we want to do is print out both the Cast ID Number and the cast member's name when we print out the object.Because we have to directly access the private attributes of the ostream class with this function, we have to make it a friend function. A "friend" function of a class (marked by the keyword friend) is a function defined outside the class. The arguments of that class has unrestricted access to all the class members (private & public).

Below is the function that should be a public function of the AT_Cast class to accomplish this:

```
friend ostream &operator << (ostream &strm, AT_Cast &member)
{
    strm << "Cast ID Number: " << member.castID << "\tName: " << member.name << endl;
    return strm;
}
```

DRIVER.CPP

Test the binary tree in a **driver** by inserting nodes with the information in the table below. Then, print out the nodes in the tree three different ways.

Cast ID Number	Name
1021	Finn
1057	Jake
2486	Ice King
3769	Princess Bubblegum
1017	Lumpy Space Princess
1275	Cinnamon Bun
1899	Peppermint Butler
4218	Marceline
1214	BMO

BINARYTREE.H

This should be a template class which contains the following private & public members:

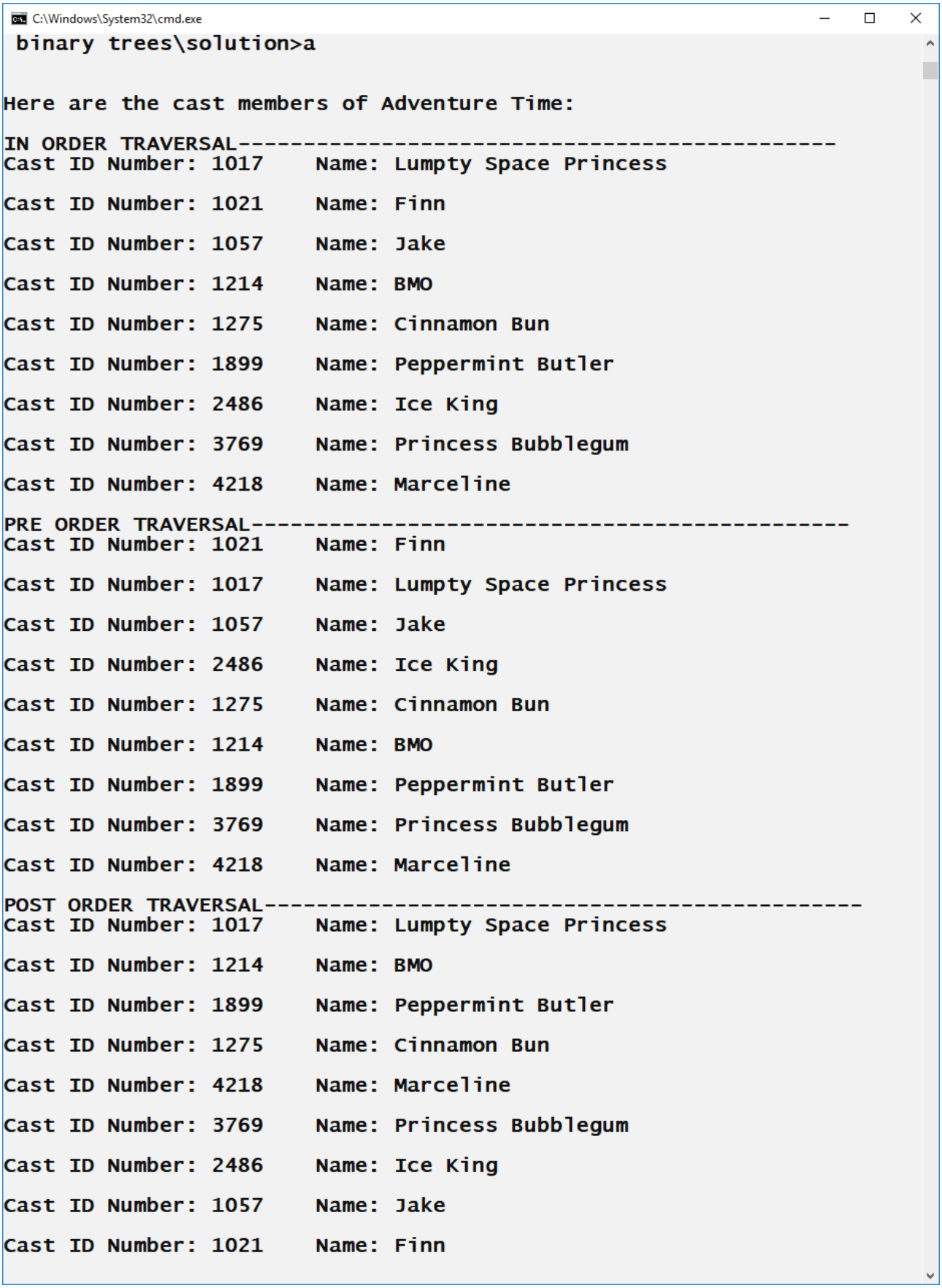
PRIVATE ATTRIBUTES & FUNCTIONS

- TreeNode structure containing a value of the template type, a pointer to the left TreeNode and a pointer to the right TreeNode
- A pointer to the root TreeNode
- Private functions:
    - Insert
    - destroySubTree
    - displayInOrder
    - displayPreOrder
    - displayPostOrder

PUBLIC FUNCTIONS

- Constructor
- Destructor (which should call destroySubTree)
- insertNode (which should call the insert function)
- displayInOrder (should call the displayInOrder private function)
- displayPreOrder (should call the displayPreOrder private function)
- displayPostOrder (should call the displayPostOrder private function)

SAMPLE OUTPUT



WHAT TO TURN IN

Please put **driver.cpp, BinaryTree.h, & AT_Cast.h** in a zipped folder and upload to ilearn submission folder.