CSCI 232 Program 1

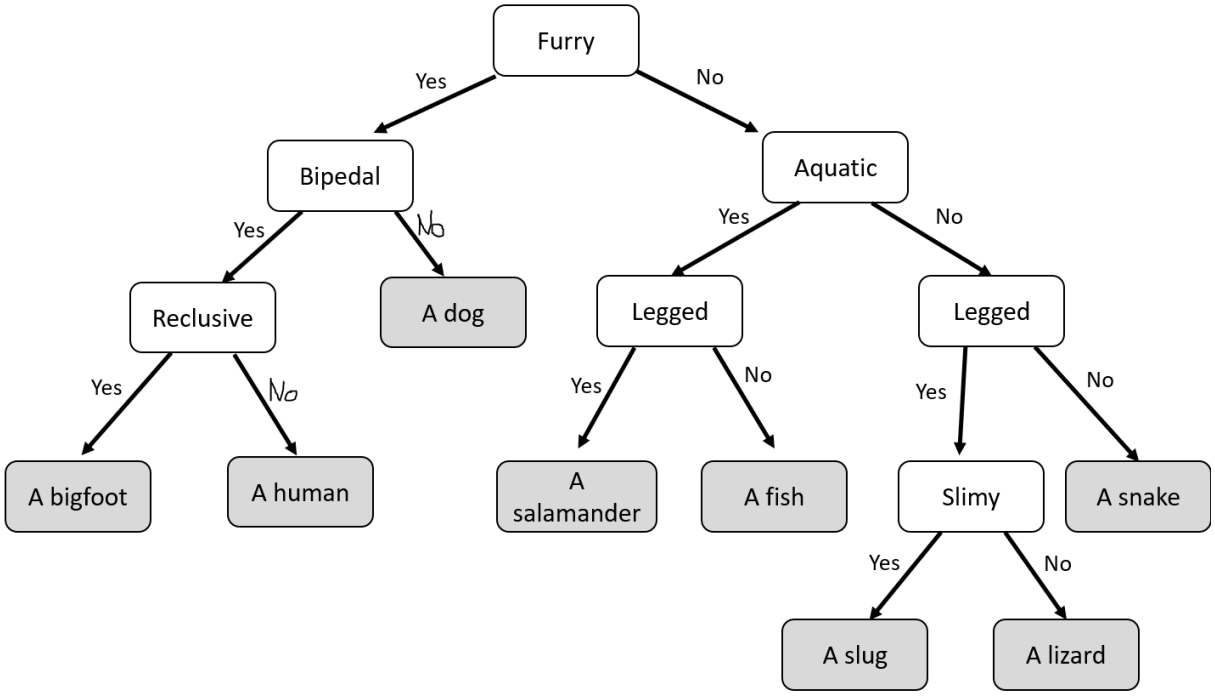
Due Tuesday May 30th @ 11:59 PM. Please submit this assignment (.java files) to the appropriate dropbox on D2L

Background and Instructions

In this program, you will build an application that uses a Binary Search Tree (BST) to identify animals. If the program is unable to identify the animal, it will gather information from the user so it can remember that animal in the future.

Animals can be classified by their characteristics (it is in the water, it doesn't have legs, it has gills, it is a fish, etc). You can imagine representing these characteristics in a tree-like structure where internal nodes are characteristics (furry, bipedal, gilled) and leaf nodes are animals (dog, human, fish). Because each characteristic can be answered with "yes" or "no", we can represent this classification system as a binary tree (each internal node has two children).

You will read in this BST from a file, and then add nodes as you discover more animals. Conceptually, this is what the tree will look like (white nodes are internal, grey nodes are leaf nodes).



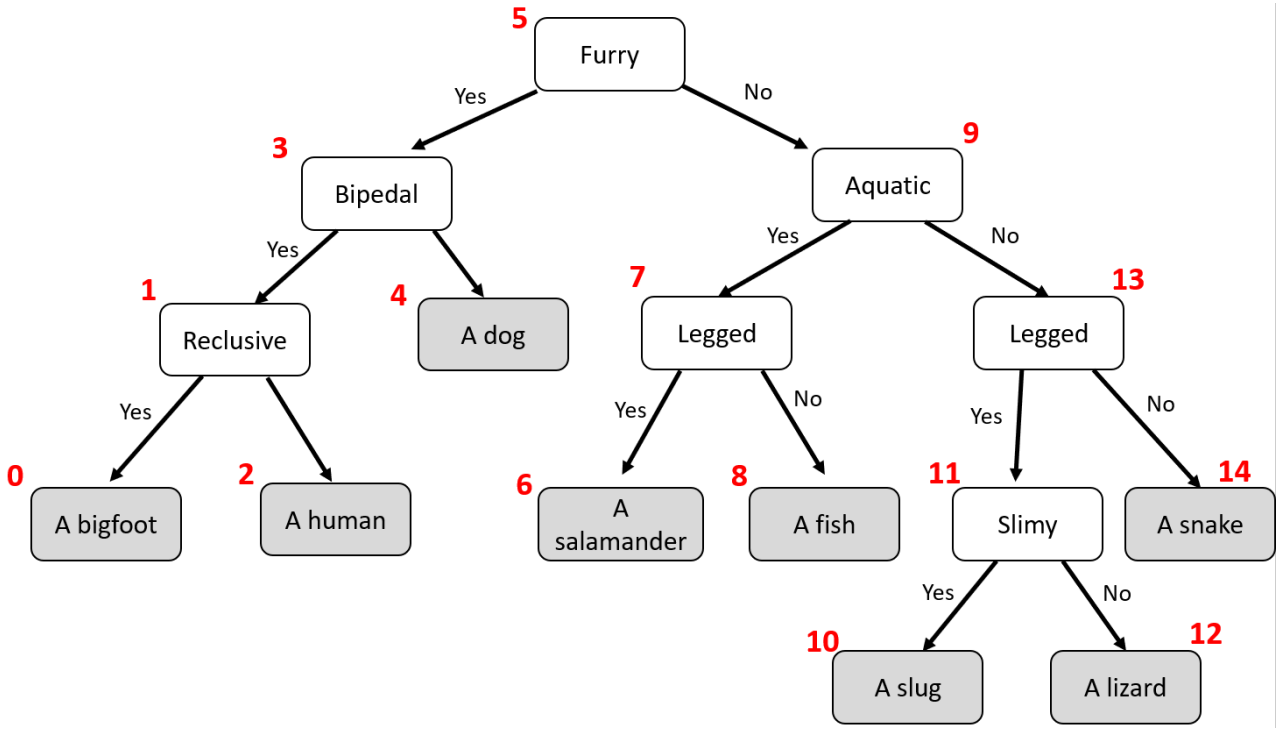
Part I: Reading BST from input file: `loadBST()`

We will write this code in class on Tuesday May 23rd. The first step is to write the body of the **`loadBST()`** method. This method constructs the baseline identification tree (pictured above), which will be created from an input file that has this information. Here is how the input file looks

(https://www.cs.montana.edu/pearsall/classes/summer2023/232/programs/tree.txt):

```
5,furry
3,bipedal
9,aquatic
1,reclusive
4,a dog
7,legged
13,legs
0,a bigfoot
2,a human
6,a salamander
8,a fish
11,slimy
14,a snake
10,a Slug
12,a Lizard
```

This is a breadth-first output of nodes from the BST. Each node has a Node number, and the prompt (for internal nodes) or animal (for leaf nodes). The Node numbers were derived from an in-order traversal of nodes in the tree. Because our application is a BST, we can use these node numbers to place the nodes in their correct spots. Notice that when node numbers are applied, you can see how this is a BST. When we build this tree using our BST **`insert()`** method, and because we have node numbers that go along with each animal/prompt, our **`insert()`** method will always build the same tree with the correct structure.



You will use the BST **`insert()`** method that we wrote in class (you may need to make slight modifications) and insert each line from the file into the tree, creating each Node object properly. After you think you have the **`loadBST()`** method working properly, you can run the Progam1Demo class, which calls a breadth-first method print out. If you got the **`loadBST()`** method working properly, you should see the following output:

```
furry 5
bipedal 3
aquatic 9
reclusive 1
a dog 4
legged 7
legs 13
a bigfoot 0
a human 2
a salamander 6
a fish 8
slimy 11
a snake 14
a Slug 10
a Lizzard 12
```

Once you have confirmed the tree is loaded properly, you can comment out the **`tree.breadthFirst()`** method call.

Part II: Using Tree to identify an animal: identify()

Next, you will fill in the body of the **`identify()`** method. The user will then try to identify an animal by answering queries from your application. If your application eventually settles on the correct animal, great, if not, it will inquire as to the differentiating trait between the animal being identified and the leaf it had settled on. NOTE: You must provide the information that I provide in my output in this circumstance (e.g. “I don’t know any furry, not squeaky, bipedal animals that aren’t a human.”). This amounts to your position in the classification tree. It will then create a new internal node and new leaf node to represent that trait as well as the new animal. See sample output below to see how the output should look when identifying an animal.

Sample Output

Sample output 1: Identifying a known animal.

```
Do you have another animal to identify? (Y/N) > Y
Is this animal furry?
N
Is this animal aquatic?
Y
Is this animal legged?
N
Hmmm... I think I know.
Is it a fish?
Y
Good! All done.
Do you have another animal to identify? (Y/N) >
```

Sample output 2: Identifying another known animal.

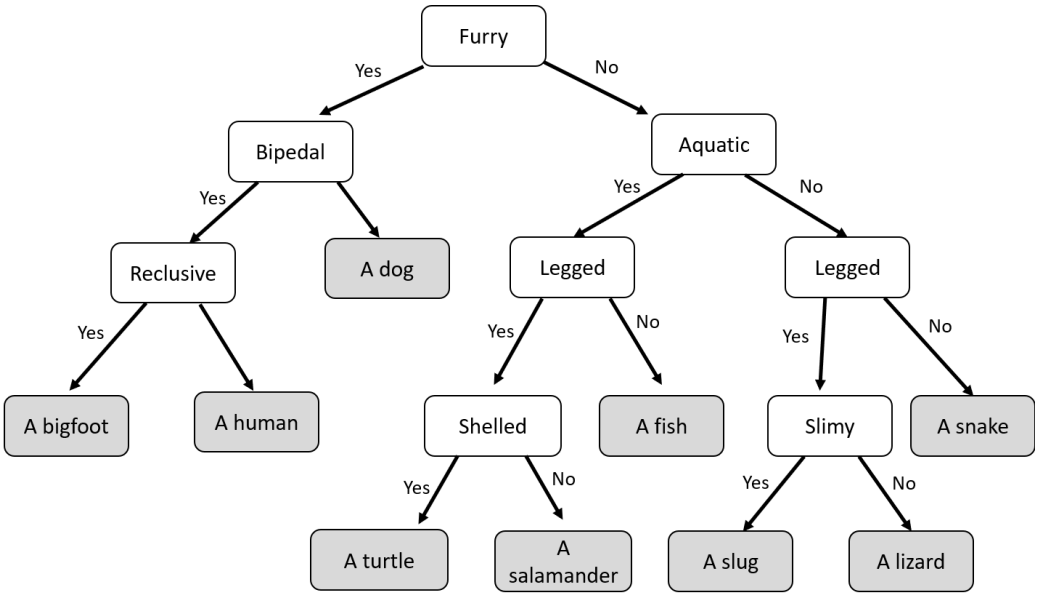
```
Do you have another animal to identify? (Y/N) > Y
Is this animal furry?
Y
Is this animal bipedal?
Y
Is this animal reclusive?
Y
Hmmm... I think I know.
Is it a bigfoot?
Y
Good! All done.
Do you have another animal to identify? (Y/N) >
```

Sample output 3 BST encounters an animal it does not know, and then stores its information.

```
Do you have another animal to identify? (Y/N) > Y
Is this animal furry?
N
Is this animal aquatic?
Y
Is this animal legged?
Y
Hmmm... I think I know.
Is it a salamander?
N
I was wrong...
I don't know any animals that are not furry  aquatic  legged
What is the new animal? >
Turtle
What characteristic does Turtle have that a salamander does not? > Shelled

Do you have another animal to identify? (Y/N) > Y
Is this animal furry?
N
Is this animal aquatic?
Y
Is this animal legged?
Y
Is this animal Shelled?
Y
Hmmm... I think I know.
Is it Turtle?
Y
Good! All done.
Do you have another animal to identify? (Y/N) >
```

In the turtle example, the new, updated identification tree looks like this:



Starting Code:

Below is some code that you should download and use to get you started. The code that you write for this assignment will mostly be in AnimalTree.Java, but you are welcome to also modify Node.Java and Program1Demo.Java. You can also modify tree.txt if you want to modify the given BST (but be careful)

- Program1Demo.Java (https://www.cs.montana.edu/pearsall/classes/summer2023/232/programs/Program1Demo.java)
- Node.Java (https://www.cs.montana.edu/pearsall/classes/summer2023/232/programs/Node.java)
- AnimalTree.Java (https://www.cs.montana.edu/pearsall/classes/summer2023/232/programs/AnimalTree.java)
- tree.txt (https://www.cs.montana.edu/pearsall/classes/summer2023/232/programs/tree.txt)

Grading

• Tree is loaded correctly from file (**`loadBST()`**) - **30 points**

• Tree can identify animals correctly (**`identify()`**) - **30 points**

• If tree cannot identify the animal, it will ask for the new characteristic, and remember that animal in future iterations (**`identify()`**) – **30 points**

• Each method written by the user has a comment describing what the method does- **10 points**

NOTE: If your code does not compile, correctness cannot be verified, and you won't receive any points for your code. Turn in code that compiles!