Assignment #5 (80 Points) – COSC 2336 – Dr. Leonard Brown
Due: December 7, 2018

General Description

There are two parts to this assignment.

Part I (30 Points)

1. Give the Postorder, Inorder, and Preorder traversals of the binary tree on the last page.

2. Give the binary tree that would generate the following traversals:

2.1: Inorder: 2, 3, 1, 5, 6, 7, 4, 9, 8
Preorder: 3, 2, 7, 5, 1, 6, 4, 9, 8

2.2: Inorder: 1, 9, 8, 7, 5, 4, 6, 3, 2
Postorder: 1, 8, 9, 5, 3, 6, 4, 2, 7

3. Beginning with an empty binary search tree, what binary search tree is formed when you insert the following values in the order given.

- Grape, Cherry, Lemon, Orange, Tangerine, Apple, Peach, Melon, Banana
- Tangerine, Peach, Orange, Melon, Lemon, Grape, Cherry, Banana, Apple
- Lemon, Melon, Orange, Peach, Tangerine, Grape, Cherry, Banana, Apple

4. Show how the node containing Lemon would be deleted from each of the binary search trees in Problem #3.

5. If the tree on the last page was implemented using the Node class for a general tree (sibling and child pointers), list the nodes (the data contained in the nodes) that would visited on the path from the root to each of the following:

5.1 Godzilla
5.2 Dr. Watson
5.3 Batman

Part II (50 points)

Write a Java program that plays a mind-reading game with a user. The program should ask a user to think of a fictional character. Then, the program will ask the user to answer a series of yes or no questions about the character. At the end, the program should display the name of the character.

Your program should utilize a binary tree to store the questions. Each internal node of the tree should contain a String representing a yes/no question. Each leaf should contain a String representing a character. The left child of an internal node represents the path taken after a "yes" response. The right child represents the path taken after a "no" response.

The above design allows your program to start at the root and traverse downwards until a leaf is reached. When a leaf is reached, the program should display the String contained in the node and ask if it is the user's character. If the answer is yes, the program has correctly "read" the user's mind. If the answer is no, the program guessed incorrectly. If this occurs, the program should input from the user the correct character, and input a new yes/no question that will distinguish the correct character from the one in the leaf.

For this design to work, your tree must be built logically. Each time an incorrect guess is given by the program, its data tree will grow. Initially, the tree should contain a single node with a default character. When an incorrect guess occurs, the program inputs a new character and a distinguishing question. Two new nodes should be created where one contains the distinguishing question and the other contains the new character. The nodes corresponding to the new character and the incorrect guess should be set as the children of the question node. This question node should be placed in the tree at the location of the incorrect guess.

An example illustrating how the program could work along with a data tree corresponding to the program follows. Initially, the data tree only contains the character "Godzilla". Since this root node is also a leaf, the program should immediately guess the character. If the user was thinking of this character, the program is correct and the game terminates. Most likely, the user was thinking of a different character (such as "Batman"). Because the program was wrong, the program needs to input the correct answer and a distinguishing question. The distinguishing question should be added at the wrong answer's location (in this case, the root), with its left child being the correct answer ("Batman") and the right child being the wrong answer ("Godzilla").

During the second execution, the user thinks of "Dr. Watson". The program starts at the new root ("Is your character human?"). Because the user enters Yes, the program moves to the left child. Since this is a leaf, the program guesses the character. Since this is also a wrong answer, the program should input the correct answer and a distinguishing question. The question is inserted at this location, which the correct and incorrect answers at its leaves.

Examples of a 3rd and 4th execution are also given. At the end of the 4th execution, the binary tree should look as illustrated on the last page.

Sample Program Execution #1

You were thinking of Godzilla.
>**NO**
I was wrong. What is your character?
>**Batman**
Enter a yes/no question distinguishing Batman from Godzilla
>**Is your character human?**

Sample Program Execution #2

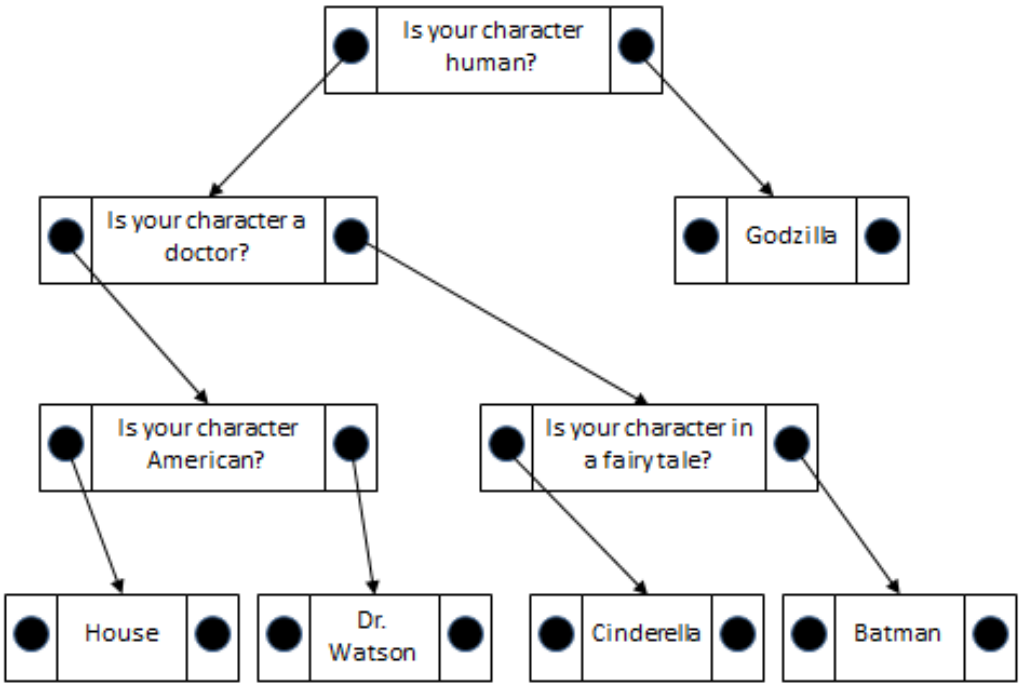
Is your character human?
>**YES**
You were thinking of Batman.
>**NO**
I was wrong. What is your character?
>**Dr. Watson**
Enter a yes/no question distinguishing Dr. Watson from Batman
>**Is your character a doctor?**

Sample Program Execution #3

Is your character human?
>**YES**
Is your character a doctor?
>**YES**
You were thinking of Dr. Watson.
>**NO**
I was wrong. What is your character?
>**House**
Enter a yes/no question distinguishing House from Dr. Watson
>**Is your character American?**

Sample Program Execution #4

Is your character human?
>**YES**
Is your character a doctor?
>**NO**
You were thinking of Batman.
>**NO**
I was wrong. What is your character?
>**Cinderella**
Enter a yes/no question distinguishing Cinderella from Batman
>**Is your character in a fairy tale?**



Grading

Part II of your assignment will be graded based on the following criteria

Implementation of Requirements	20 points
Correct Computation of Input/Output	20 points
Design and Documentation	10 points

Submission

Submit your assignment through Canvas. If your assignment contains multiple files, zip them into a single folder before submitting.

Notes

The same requirements in Assignment #1 regarding the documentation and formatting style apply to this assignment as well.

Points can be deducted from your assignment based on the quality of its presentation. Handwritten assignments will **not** be accepted.