Binary Tree Lab
Potential Applications for Use in AI

Lab Objectives

- Be able to write a node system for a binary tree that holds multiple variables
- Increment variables within the nodes based on criteria
- Access a system that returns randomized individuals to fill a population
- Be able to collect data from multiple nodes in the tree

Introduction

With the ongoing pandemic, I felt it would be appropriate to make a program that would explore how data may be organized from all the data being collected. The inspiration for this is the potential for artificial intelligence applications. The main idea being how might an artificial intelligence system use or manipulate this data. It definitely seems like the field of predictive analytics would benefit greatly from artificial intelligence.

For my lab, I explored implementing a binary tree. Each of the nodes represented an age. This seemed like a logical way of using a tree as the number of different ages is finite and the root can be the median age. For simplicity, I limited age from 1 too 100 using 50 as my root. Each of the nodes contains information on how many times this node has been created that represents a population size. Upon creation, nodes can also be marked as high risk. This will increment a counter of high-risk individuals in the population. High risk is determined by preexisting conditions and age.

To represent the use in predictive analytics I have collected data like the US population, the population of are groups within the US population, and how many adult Americans have preexisting conditions. This is implemented to give randomized results based on these criteria. While the data it provides is inaccurate for any real-world application due to it not taking many factors into account, I feel it effectively demonstrates how you might use similar data in a system to predict expected high-risk individuals within a smaller or larger population.

Task #1 Create Node Class

1. Copy the files *CalculateRiskTester.java*, *Creator.java*, and *BinaryTree.java*. *Creator.java* and *CalculateRiskTester.java* are complete and do not need to be modified.
2. Create a new class called Node. The defining value a node will have is the age of the persons contained within. In addition to age, we will need to keep track of population size, how many of the population are high risk, and the left and right children Nodes of the object.
3. Create a constructor that takes an integer representing age. Initialize this integer to age.
4. We will also need various methods. We need getters for age, population, and high-risk population. Setters are also needed for population and high risk. Setters will simply increment their population. Finally, we will need getters and setters for the left and right child nodes.

Task #2 Create a system to fill the population with nodes

1. Create a constructor in the BinaryTree class. The constructor does not need any parameters and will need to set the root to a new node with age 50.
2. Next, we need a way to add both new nodes into the system and to increase the population counts of preexisting nodes. This is where we will connect to the creator class to generate an individual. This individual's information will be used in creating or incrementing nodes appropriately.
3. If a generated individual has a risk level of one then increment the population in addition to the high-risk population.

Task #3 Create a find method

1. In order to collect specific pieces of data, we need a way to search through the tree and pull out certain nodes based on age.
2. To do this we will create a method the takes an integer parameter and returns a node. Due to creating the tree with lower numbers to the left and high to the right, we will be searching the tree in the same way.
3. If we are unable to find the node we want to return a Node with an age value of -1 so we do not throw a NullPointerException. Otherwise, return the located Node.

Test Data

Due to the randomized data of the lab we are looking for numbers that will have high similarity when using a large sample size.

In testing, I found that in a sample size of 10,000,000 typically about 37,000 were at high risk.

It is important to check the low values as well to make sure that we have no errors. If you had a population of 50 people such as a small community that you wanted to model this program should still output results. While they would vary greatly between program iterations and average over time would help to normalize the data to be more in line with expected results.

References

https://en.wikipedia.org/wiki/Binary_tree
https://www.cdc.gov/coronavirus/2019-ncov/need-extra-precautions/people-at-higher-risk.html
https://www.census.gov/quickfacts/fact/table/US/AGE775218
https://www.kff.org/coronavirus-covid-19/issue-brief/how-many-adults-are-at-risk-of-serious-illness-if-infected-with-coronavirus/

Outcome

Working with binary trees can be rather difficult as there are a lot of moving parts but they can be rather rewarding. As for my personal interest moving forward, during this project, I have definitely had a growing interest in what similar solutions to similar problems might look like. For applications such as artificial intelligence, there does seem to be great potential in that field. I

definitely still have a lot to learn and while my method for implementing this could be improved, it still felt encouraging to complete it.