Assignment 6 --- Binary Search Trees

This assignment has two parts; the first part covers binary trees where the goal is to construct and visualize a new binary tree implementation using Bridges BinTreeElement. The second covers binary search trees in which you will be asked to complete the BST class .

Part I

Overview

The primary goals of this program are as follows:

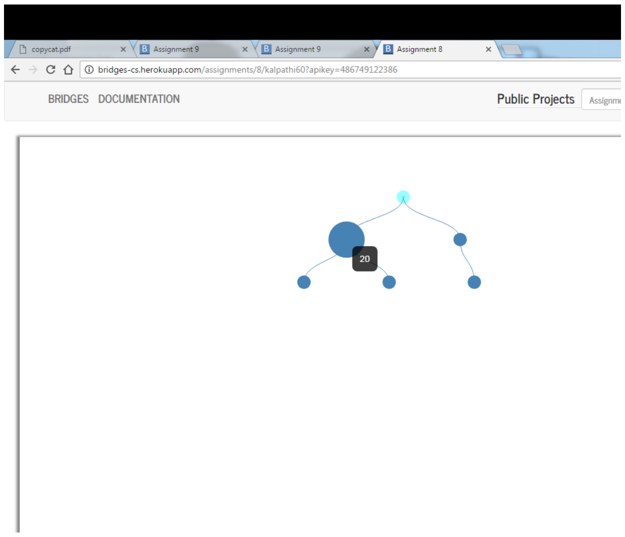
(1) Create a **simpleBinaryTree** class that implements Binary Tree using Bridges BinTreeElement objects.
(2) Create a main method to demonstrate the Binary Tree data structure. Assign visual properties to the elements of the Binary Tree.

The **Binary Tree implementation** will be similar to the List implementation we worked on earlier but you will use BinTreeElement to construct the nodes and setLeft and setRight to connect the nodes the nodes to each other and build the tree.

The **Tester** should create an instance of the Binary Tree class. The binary tree will store 6 integer values. Make sure you use the integer values as the labels for the nodes. The binary tree should be visualized with the root element featuring different visual properties. Visualizing the data structure and adding visual properties to binary tree elements can be accomplished easily with the Bridges API.

Deliverables –

Your program should generate will be something like this:



Part II

BST Assignment–

Overview

The primary goals of this program are as follows:

(1) Complete the BST class using Bridges *BinTreeElement* objects.

(2) Create a Tester class to demonstrate the Binary Tree data structure. Assign visual properties to the elements of the Binary Tree.

The **Binary Tree implementation (BST class)** will implement the Dictionary interface and extends the comparable class. You will complete the implementations of the methods in the Dictionary interface.

The **Tester** should create an instance of the Binary Tree class. The binary tree will store a number of string words. The binary tree should be visualized with the root element featuring different visual properties. Visualizing the data structure and adding visual properties to binary tree elements can be accomplished easily with the Bridges API.

Using the provided BST class skeleton do the following:

--- Import *bridges.base.BSTElement library* to allow you to use BSTElement objects in this class

--- Don't forget to parameterize the SLelement objects with the generic type arguments, Key, E. You can examine the documentation for BSTElement nodes.

--- Create the following methods in the BST class:

```
private BSTElement<Key,E> inserthelp(BSTElement<Key,E> rt, Key k, E e)
private BSTElement<Key,E> getmin(BSTElement<Key,E> rt)
private BSTElement<Key,E> deletemin(BSTElement<Key,E> rt)
private void inOrderTraversal (BSTElement<Key,E> rt)
private void postOrderTraversal (BSTElement<Key,E> rt )
private void preOrderTraversal (BSTElement<Key,E> rt)
```

Tester –

--- Create a Tester class with a main method

--- Import *bridges.connect.Bridges* to allow you to create a Bridges object

--- Initialize a Bridges object with the assignment number, your username, and your API key.

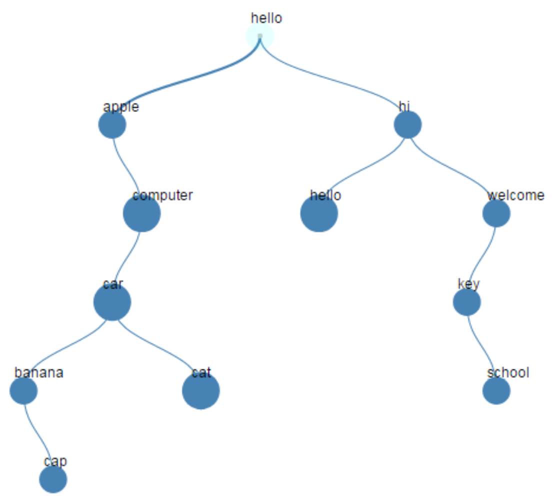
--- Create an instance of your BST class and parameterize it to hold elements and keys

--- Insert 13 new elements to your BST.

--- Call Bridges' setDataStructure method passing in the head of your list and then call Bridges' visualize method.

Deliverables –

The final Bridges assignment your program should generate will be something like this:



Scoring Rubric

Part I: 80 points

--- Up to 10 points for appropriate documentation and comments
--- Up to 10 points for correctly adding 6 elements to the list
--- Up to 20 points for setting the labels for all the added elements
--- Up to 20 points for setting new distinct visual properties for the root and the smallest element
--- Up to 20 points for visualizing the BST data structure

Part II: 120 points

--- Up to 10 points for appropriate documentation and comments
--- Up to 10 points for correctly adding 13 elements to the list
--- Up to 45 points for correctly implementing the traversal methods
--- Up to 20 points for implementing the insert method
--- Up to 10 points for correctly implementing the remove min method
--- Up to 10 points for correctly implementing the get minimum method
--- Up to 5 points for filling this survey: https://unccpsych.az1.qualtrics.com/jfe/form/SV_dcK4tmEmIQ68hDv

Total points available: 200