import java.util.\*;  
  
//This is our binary search tree class with the Node class and all related methods inside the BST class  
class BST {  
 Node root;  
 BST (){  
 root = null;  
 }  
 //This method will take the key and value in the argument and create a new node  
 public void addNode(String key, String value){  
 Node newNode = new Node(key, value);  
 if (root == null) {  
 root = newNode;  
 return;  
 }  
  
 Node currentNode = root;  
 /\*This part judges whether the node is smaller or larger than the  
 current node and moves or creates the node accordingly\*/  
 while (true){  
 if (key.compareTo(currentNode.key) < 0){  
 if (currentNode.left == null){  
 currentNode.left = newNode;  
 return;  
 } else  
 currentNode = currentNode.left;  
 } else {  
 if (currentNode.right == null){  
 currentNode.right = newNode;  
 return;  
 } else  
 currentNode = currentNode.right;  
 }  
 }  
 }  
 class Node {  
 String key;  
 String value;  
 Node left,right;  
  
 Node (String key, String value){  
 this.key = key;  
 this.value = value;  
 left = null;  
 right = null;  
 }  
 }  
}  
  
public class Main {  
 /\*This method sorts the array alphabetically according to their capitals,  
 and keeps the associated state connected to the capital as they are sorted\*/  
 public static void sortByCapitals(String [][] arr){  
 int n = arr[1].length;  
 for (int i = 0; i < n - 1; i++){  
 for (int j = 0; j < n - 1 - i; j++){  
 if (arr[1][j].compareTo(arr[1][j+1]) > 0) {  
 String capTemp = arr[1][j];  
 arr[1][j] = arr[1][j+1];  
 arr[1][j+1] = capTemp;  
 //These additional lines are here to ensure the state moves accordingly with its sorted capital  
 String stateTemp = arr[0][j];  
 arr[0][j] = arr[0][j+1];  
 arr[0][j+1] = stateTemp;  
 }  
 }  
 }  
 }  
 //Only the row with the capitals is printed  
 public static void printCapitals(String [][] arr){  
 for (int i = 0; i < 50; i++){  
 System.*out*.println(arr[1][i]);  
 }  
 }  
 /\*Both rows of the array are printed.  
 I could have done 2 for-loops, but since there were only 2 rows this seemed easier\*/  
 public static void printBoth(String [][] arr){  
 for (int i = 0; i < 50; i++){  
 System.*out*.println(arr[0][i] + ": " + arr[1][i]);  
 }  
 }  
 public static void main(String[] args) {  
 Scanner scanner = new Scanner(System.*in*);  
 int correctCount = 0;  
 BST treeBST = new BST();  
 // Here is our array with 50 states in row [0] and 50 capitals in row [1]  
 String [][] list = {  
 {"Montana", "Nebraska", "Nevada", "New Hampshire", "New Jersey", "New Mexico", "New York", "North Carolina", "North Dakota", "Ohio", "Oklahoma", "Oregon", "Pennsylvania", "Rhode Island", "South Carolina", "South Dakota", "Tennessee", "Texas", "Utah", "Vermont", "Virginia", "Washington", "West Virginia", "Wisconsin", "Wyoming", "Missouri", "Mississippi", "Minnesota", "Michigan", "Massachusetts", "Maryland", "Maine", "Louisiana", "Kentucky", "Kansas", "Iowa", "Indiana", "Illinois", "Idaho", "Hawaii", "Georgia", "Florida", "Delaware", "Connecticut", "Colorado", "California", "Arkansas", "Arizona", "Alaska", "Alabama"},  
 {"Helena", "Lincoln", "Carson City", "Concord", "Trenton", "Santa Fe", "Albany", "Raleigh", "Bismarck", "Columbus", "Oklahoma City", "Salem", "Harrisburg", "Providence", "Columbia", "Pierre", "Nashville", "Austin", "Salt Lake City", "Montpelier", "Richmond", "Olympia", "Charleston", "Madison", "Cheyenne", "Jefferson City", "Jackson", "Saint Paul", "Lansing", "Boston", "Annapolis", "Augusta", "Baton Rouge", "Frankfort", "Topeka", "Des Moines", "Indianapolis", "Springfield", "Boise", "Honolulu", "Atlanta", "Tallahassee", "Dover", "Hartford", "Denver", "Sacramento", "Little Rock", "Phoenix", "Juneau", "Montgomery"}  
 };  
 //This calls our previous methods and iterates through our array by printing the state and the capital  
 System.*out*.println("List of States and connected Capitals: ");  
 *printBoth*(list);  
 System.*out*.println(" ");  
  
 //Sorts and prints only the capitals  
 *sortByCapitals*(list);  
 System.*out*.println("List of Capitals in alphabetical order: ");  
 *printCapitals*(list);  
 System.*out*.println(" ");  
  
 //Converts the 2D array into a hashmap  
 System.*out*.println("HashMap printed: ");  
 HashMap<String, String> cityStates = new HashMap<String, String>();  
 for (int i = 0; i < 50; i++)  
 cityStates.put(list[0][i], list[1][i]);  
 System.*out*.println(cityStates);  
 System.*out*.println(" ");  
  
 //Converts the hashmap into a tree map, sorted via the states instead of capitals  
 System.*out*.println("TreeMap printed: ");  
 TreeMap<String, String> cityStatesTree = new TreeMap<String,String>(cityStates);  
 System.*out*.println(cityStatesTree);  
  
 //This creates a BST from the 2D array  
 for (int i = 0; i < 50; i++){  
 treeBST.addNode(list[0][i], list[1][i]);  
 }  
  
 //This iterates through our states and allows an input following each state printed  
 System.*out*.println(" ");  
 System.*out*.println("What is the capital of the following states: ");  
 for (int i = 0; i < 50; i++){  
 System.*out*.println(list[0][i]);  
 System.*out*.println("Enter the Capital or type \"quit\" to quit: ");  
 String answerAtttempt = scanner.next();  
 if (answerAtttempt.equalsIgnoreCase("quit"))  
 break;  
 String answerRight = list [1][i];  
 //This checks to see if the input matches the array's capital and increases our counter if true  
 if (answerAtttempt.equalsIgnoreCase(answerRight)) {  
 System.*out*.println("Correct!");  
 correctCount++;  
 } else  
 System.*out*.println("Incorrect.");  
 }  
 System.*out*.println(" ");  
 System.*out*.println("Number of correct: " + correctCount);  
  
 System.*out*.println(" ");  
  
 //This allows user to enter a state to receive the appropriate capital as output  
 boolean loop = true;  
 try {  
 while (loop) {  
 System.*out*.println("Please enter a state and I will tell you the capital: ");  
 System.*out*.println("Type \"quit\" if you want to terminate.");  
 String input = scanner.next();  
 if (input.equalsIgnoreCase("quit"))  
 loop = false;  
 if (cityStatesTree.containsKey(input))  
 System.*out*.println(cityStatesTree.get(input));  
 }  
 }catch (Exception e) {  
 System.*out*.println("Please try again");  
 }  
 }  
}