**CSE220 Lab Quiz 5**

**Monday 11 AM Slot**

Tentative Solutions and Rubrics

1.2 Set A

| class BTNode {  int energy;  BTNode left, right;  public BTNode(int energy) {  this.energy = energy;  this.left = null;  this.right = null;  }  }  public class Main {  public static void main(String[] args) {  // Tree Construction  BTNode root = new BTNode(20);  BTNode n1 = new BTNode(10);  BTNode n2 = new BTNode(25);  root.left = n1;  root.right = n2;  BTNode n3 = new BTNode(5);  BTNode n4 = new BTNode(15);  n1.left = n3;  n1.right = n4;  BTNode n5 = new BTNode(35);  n2.right = n5;  BTNode n6 = new BTNode(30);  BTNode n7 = new BTNode(40);  n5.left = n6;  n5.right = n7;    // Testing  System.out.println(calculate\_energy(root,30));  System.out.println(calculate\_energy(root,34));  }  public static String calculate\_energy(BTNode root, int energy) {  int totalEnergy = 1;  BTNode current = root;  while (current != null) {  totalEnergy \*= current.energy;  if (current.energy == energy) {  return "Total energy: " + (totalEnergy);  } else if (energy < current.energy) {  current = current.left;  } else {  current = current.right;  }  }  return "route does not exist";  }  } |
| --- |

1.3 Set B

| class BTNode {  int tax;  String city;  BTNode left, right;  public BTNode(int tax, String city) {  this.tax = tax;  this.city = city;  this.left = null;  this.right = null;  }  }  public class Main {  public static void main(String[] args) {  // Tree Construction  BTNode khulna = new BTNode(20, "Khulna");  BTNode ruppur = new BTNode(10, "Ruppur");  BTNode bhanga = new BTNode(25, "Bhanga");  khulna.left = ruppur;  khulna.right = bhanga;  BTNode pabna = new BTNode(5, "Pabna");  BTNode bogra = new BTNode(15, "Bogra");  ruppur.left = pabna;  ruppur.right = bogra;  BTNode bhulta = new BTNode(35, "Bhulta");  bhanga.right = bhulta;  BTNode rupganj = new BTNode(30, "Rupganj");  BTNode sylhet = new BTNode(40, "Sylhet");  bhulta.left = rupganj;  bhulta.right = sylhet;  // Testing  System.out.println(crossingTax(khulna, "Sylhet",40));  System.out.println(crossingTax(khulna, "Dhaka",34));  }  public static String crossingTax(BTNode root, String targetCity, int targetCityTax) {  int totalTax = 0;  BTNode current = root;  while (current != null) {  totalTax += current.tax;  if (current.city.equals(targetCity)) {  return "Total Tax: " + (totalTax - root.tax) + " tk";  } else if (targetCityTax < current.tax) {  current = current.left;  } else {  current = current.right;  }  }  return "route does not exist";  }  } |
| --- |

1.4 Marking Scheme

| **SL** | **Points to meet** | **Marks (15)** |
| --- | --- | --- |
| 1 | Construct the Node class | 2.5 |
| 2 | Construct the BST | 2.5 |
| 3 | Defining the function with correct parameters | 1 |
| 4 | Right return condition | 1.5 |
| 5 | Correct Recursive calls | 3 |
| 6. | Correct Calculation (summation for set A and correct conditions for set B ) | 3 |
| 7. | Correct Output statements | 1.5 |
| **Total** | | **10** |