DEERWALK INSTITUTE OF TECHNOLOGY

LAB 1: ANIMAL GUESSING GAME IMPLEMENTING BINARY SEARCH TREE

(ARTIFICIAL INTELLIGENCE)

SUBMITTED BY:
NAME: Suraj Prasai
PROGRAM: B.Sc. CSIT (SEM V)
ROLL NO.: 0537
SECTION: A
DATE: 23rd February, 2018

SUBMITTED TO:

BIRODH RIJAL

KATHMANDU, NEPAL

2018

PROBLEM STATEMENT

Implement animal guessing game using Binary Search Tree in any programming language.

DESCRIPTION

Animal Guessing game is game in which the user thinks of some animal as their choice. The game then asks certain questions to the user multiple questions stored in the nodes of its binary tree. The nodes are propagated by the yes/no answer provided by the user. At the end node, the game displays the value in the node and asks user for confirmation. If the guess is correct, it provides the option to play again else it will ask the user about the animal they have thought of along with a distinct feature that recognizes that animal. The new animal is now added in our game.

METHODOLOGY

To implement the animal game, a simple program was written in JAVA. The source code is as below:

```
        rightChild = new Node(rightQuestion);           //set right node of root
        rightChild.setLeft(new Node(answer3));
        rightChild.setRight(new Node(answer4));
        root.setRight(rightChild);

        return root;
}


public static void playGame (Node guess)
{

    while (!isLeaf(guess)) {
        counter++;
        if ( askQuestion(guess.getValue()))        //if user selects yes as the value stored in node, guess is set as left else as right
        {
            guess = guess.getLeft();
        }else
        {
            guess = guess.getRight();
        }
    }
    System.out.println("My Guess is " +guess.getValue());       //leaf node reached after n iteration
    if(askQuestion("Am i right? "))
    {

        System.out.println("I Guesses it in "+counter+" tries !!");
        System.out.println("You see I am very Intelligent. :) ");
    }else
    {
        addAnimal(guess);           //add new Animal the user have thought of ,as it is not currently stored
    }

}


public static Boolean isLeaf(Node leaf)            //check if node is root i.e have no root
{
    if(leaf.right == null && leaf.left == null)
    {
        return true;
    }else
    {
        return false;
    }
}

public static Boolean askQuestion(String question)      //receives yes/no answer from user by asking certain question
{
    System.out.println(question);
    String playAgain = sc.nextLine();

    if(playAgain.equalsIgnoreCase( anotherString: "y") || playAgain.equalsIgnoreCase( anotherString: "yes"))
    {
        return true;
    }else if (playAgain.equalsIgnoreCase( anotherString: "n") || playAgain.equalsIgnoreCase( anotherString: "no"))
    {
        return false;
    }else
    {
```

```
    }else
    {
        System.out.println("Please enter a valid choice. ( Y / N ) : ");
        askQuestion(question);
    }
    return true;
}

public static void addAnimal(Node newNode)
{
    String guessAnimal = newNode.getValue();
    String newAnimal;
    String animalFeature;


    System.out.println("Sorry! i Failed to recognize you. ");
    System.out.println("Please help me register you in my Memory");
    System.out.println("Can you tell me who you are? ");
    newAnimal = sc.nextLine();
    System.out.println("What makes u unique from others? ( A Yes / NO feature perhaps ): ");
    animalFeature = sc.nextLine();

    newNode.setValue("Do you " +animalFeature);         //replace current value by new animal's feature

    if(askQuestion("Do you " +animalFeature + "?: "))
    {
        newNode.setLeft(new Node(newAnimal));               //add animal in left node if feature is true for it
        newNode.setRight(new Node(guessAnimal));            //current value replaced above is placed in right node
    }else
    {
        newNode.setLeft(new Node(guessAnimal));
        newNode.setRight(new Node(newAnimal));
    }


}

public static class Node {                      //class to create nodes in BST
     String value;                          //Question stored in the node
     Node left;                             // If answer to question is yes, next question stored in Left node
     Node right;                            //If value is no then stored in Right node

    public Node(String value) {         //Constructor for Node
        this.value = value;             //Set value as value passed as argument
        left= null;
        right = null;           //initialize left and right nodes as null
    }

    public String getValue() {
        return value;
    }

    public void setValue(String value) {
        this.value = value;
    }
```

```
    public Node getLeft() {
        return left;
    }

    public void setLeft(Node left) {
        this.left = left;
    }

    public Node getRight() {
        return right;
    }

    public void setRight(Node right) {
        this.right = right;
    }
}
}
```

EXPLANATION:

1. Initially a Binary Search Tree class was created.

2. To make nodes of the BST, the blueprint inner class Node was created.

3. A counter to count number of iteration taken to guess user choice is defined.

4. The main method displays welcome information and initializes root of BST with initial values.

5. Main calls the method playGame(node) repeatedly if user wants to play again and again.

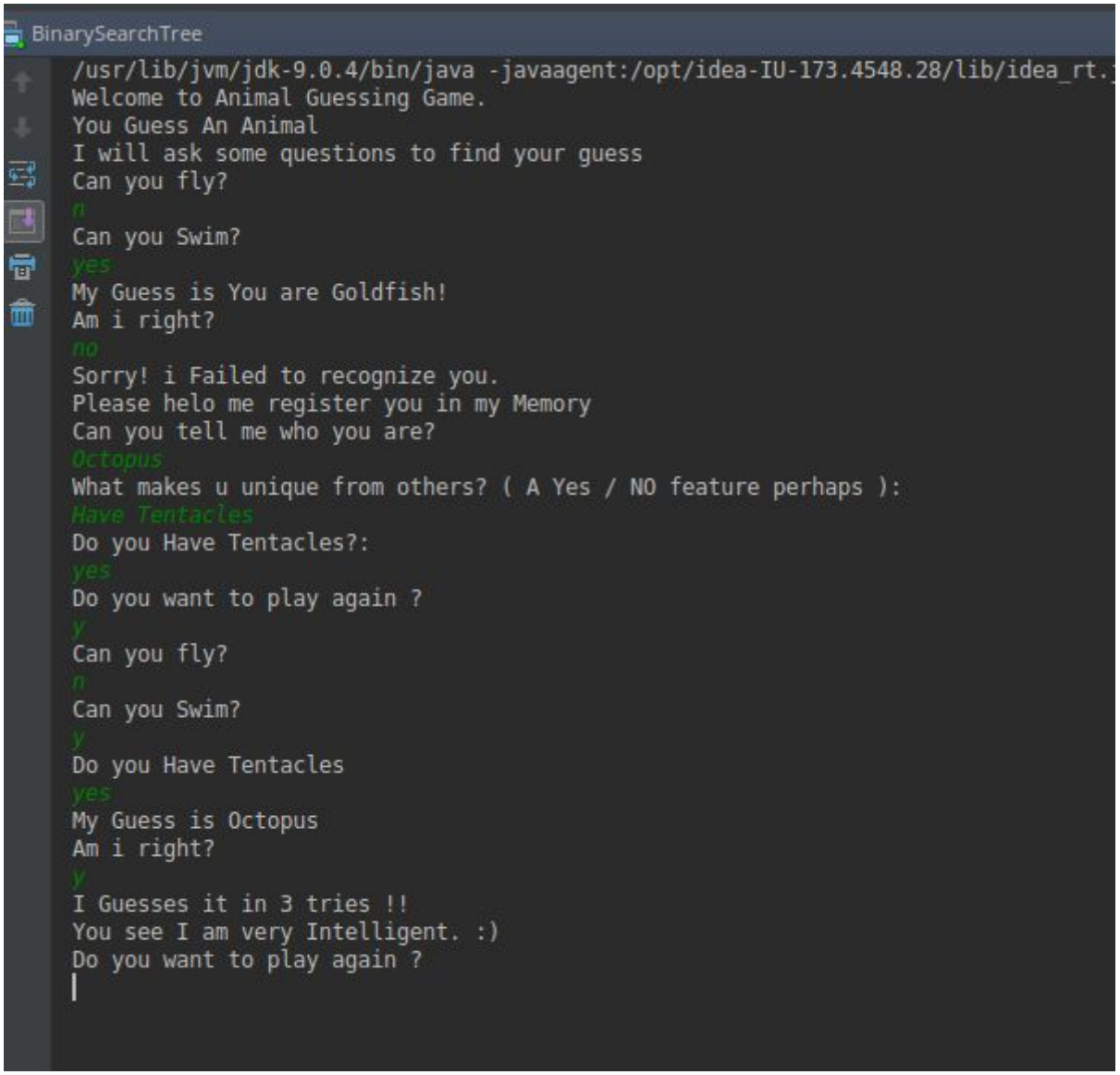
6. In playGame(node), the user is asked question on root to the user, determining the answer, it sets node as left value of node if user answers “yes” else it sets node as right. It iterates until leaf node is reached. It then asks question on the leaf node.

7. If it is correct answer, program ends and user can play again.

8. If the answer is not correct, it asks user for their guess. It replaces the leaf node with new animal's feature of user. The previous leaf is set on right value of this node (as a false value to this question)

9. New animal is added onto the BST, the game can successfully predict if it is the same animal the user have thought of.

OUTPUT



LIMITATION

1. Every time program is started, only initial values are setup. Animals added in previous implementations will not be added.

2. At least 2 child of the root must be predefined up to second level.

3. Same animal can be added with new or same feature.

4. It has only 2 child on every level, more iterations needed to find unique animals(n-ary tree could be used).