DEERWALK INSTITUTE OF TECHNOLOGY

LAB 1: ANIMAL GUESSING GAME (IMPLEMENTATION OF BINARY SEARCH TREE)

(ARTIFICIAL INTELIGENCE)

SUBMITTED BY:
NAME: SUSHIL AWALE
PROGRAM: B.SC.CSIT (FIFTH SEM)
ROLL NO.: 0540
SECTION: A
DATE: 23 FEB 2018

SUBMITTED TO:

BIRODH RIJAL

KATHMANDU, NEPAL

2018

PROBLEM

Write a program in any high level programming language to implement an animal guessing game.

BACKGROUND

Animal guessing game is a toy problem in the field of AI. The game works as follows:

1. The program asks the user to think of an animal which it tries to guess.
2. The program asks the user a series of yes/no questions and user has to reply.
3. The questions vary on the basis of user's reply.
4. Finally, the program makes a guess and asks the user for confirmation.
5. If the guess is wrong, the program asks the user what he/she was thinking of and also asks what appropriate question would help to distinguish the animal.
6. The program stores this information in its memory and uses it the next user plays the game.

METHODOLOGY

The following program was written to implement the game. The program is written in Kotlin and uses a Binary Search Tree consisting of Node data structure.

The Node consists of the data (question/guess answer) and two pointers to children nodes. If the user answers with a 'YES' the tree traverses to the right, otherwise left. Another node pointer, named *currentNode* tracks the current state of the game and if the guess is wrong, the program learns from the user and stores a new node in the current node.

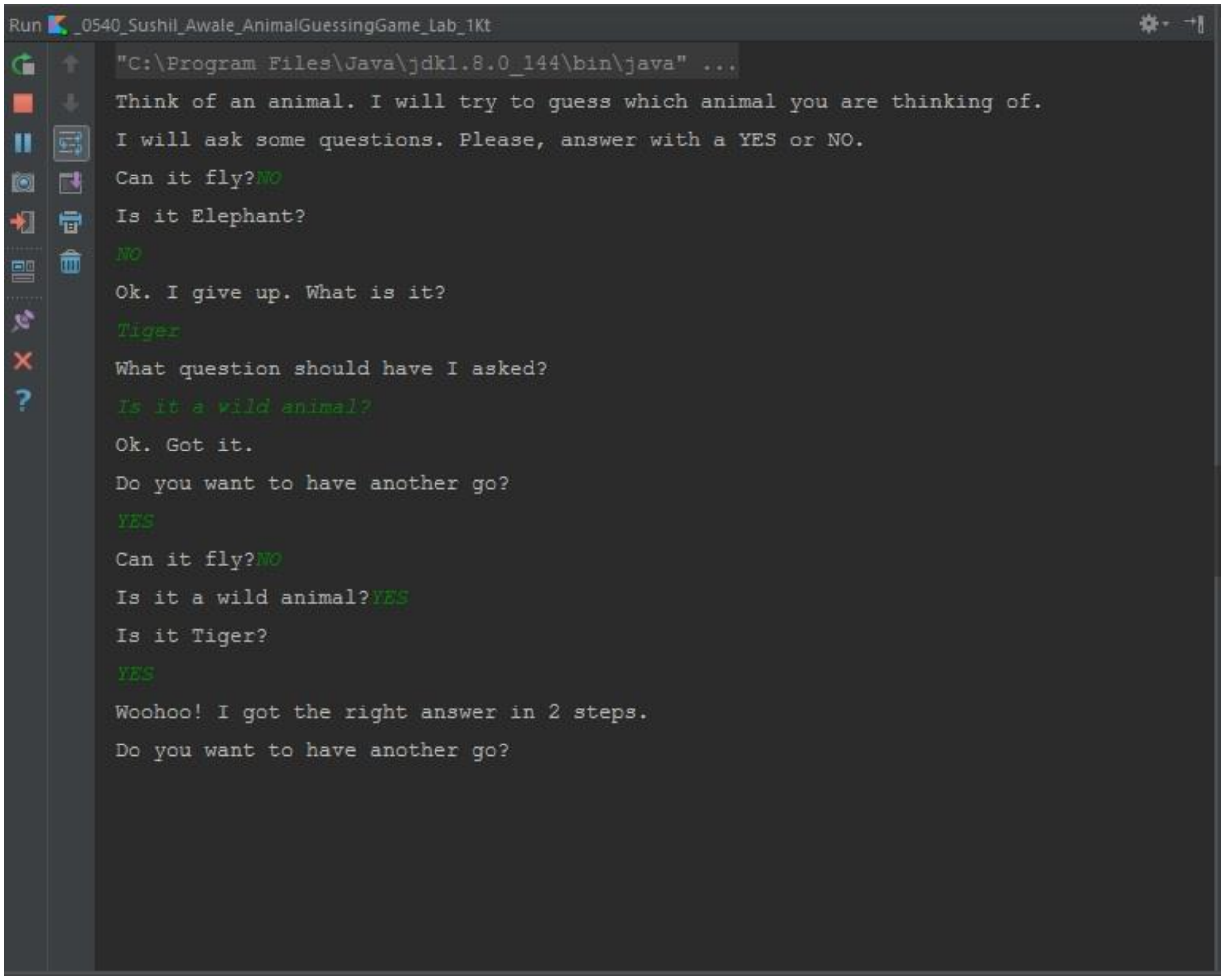
After the end of the game, the user can opt to replay the game.

PROGRAM CODE

```
/* Node data structure for binary search tree */
class Node (var data: String, var leftChild: Node? = null, var rightChild:
Node? = null)

class BinarySearchTree {

    var rootNode = Node("") /* Initialize root node */
    var currentNode = Node("") /* A pointer variable to the current node*/

    /* Initialize the Binary Search Tree with initial data */
    fun initialize(data: String, yesAnswer: String, noAnswer: String) {
        rootNode = Node(data, Node(noAnswer), Node(yesAnswer)) /* Create
new node and set it as root node */
    }

    /* Function to insert children node for left and right node of current
node */
    fun insertGrandChildren(currentNode: Node, data: String, userAnswer:
String) {
        currentNode.leftChild = Node(currentNode.data) /* Shift the
current answer to a new left node */
        currentNode.rightChild = Node(userAnswer) /* Store the user answer
*/
        currentNode.data = data /* Replace answer with new hint given by
user */
    }
}

class AnimalGuessing {

    var BST = BinarySearchTree() /* Initialze binary search tree */
    var counter: Int = 0 /* Initialize counter to calculate number of
steps */

    fun play() {
        BST.currentNode = BST.rootNode /* Each time the user plays the
game, the pointer points to the initial node */
        counter = 0 /* Set counter to zero for new game */
        while (BST.currentNode.leftChild != null ||
BST.currentNode.rightChild != null){ /*  Traverse until a child node is
null. */
            askQuestion()
            var userResponse: String = readLine()!!
            when (userResponse.toUpperCase()) {
                "YES" -> BST.currentNode = BST.currentNode.rightChild!! /*
Traverse right */
                "NO" -> BST.currentNode = BST.currentNode.leftChild!! /*
Traverse left */
                else -> println("My creator only designed me to understand
a YES or NO. :( ")
            }
            counter++ /* Increment counter */
        }
            guess()
    }

    private fun guess() { /* This function makes a guess when the pointer
reaches the last node */
        println("Is it " + BST.currentNode.data + "?") /* Make a guess */

        var userConfirmation: String = readLine()!!
        if (userConfirmation.toUpperCase() == "YES") {
            println ("Woohoo! I got the right answer in $counter steps. ")
            replay()
        } else if (userConfirmation.toUpperCase() == "NO") {
            learn()
        }
    }

    private fun replay() {
        println ("Do you want to have another go?")

        var userResponse: String = readLine()!!
        when (userResponse.toUpperCase()) {
            "YES" -> play()
            "NO" -> println ("Bye! Thanks for teaching me.")
            else -> println ("My creator only designed me to understand a
YES or NO. :( ")
        }
    }

    private fun askQuestion() = print(BST.currentNode.data)

    private fun learn() {
        println ("Ok. I give up. What is it?")
        var userAnswer: String = readLine()!!

        println ("What question should have I asked?")
        var newHint: String = readLine()!!

        BST.insertGrandChildren(BST.currentNode, newHint, userAnswer) /*
Insert newly learned information */
        println("Ok. Got it.")
        replay()
    }
}

fun main (args : Array<String>) {

    println ("Think of an animal. I will try to guess which animal you are
thinking of.")
    println ("I will ask some questions. Please, answer with a YES or
NO.")

    var game = AnimalGuessing() /* Initialize the game */
    game.BST.initialize("Can it fly?", "Pigeon", "Elephant") /* Input
```

```
initial data */
    game.play() /* Play game */
}
```

OUTPUT



Run _0540_Sushil_Awale_AnimalGuessingGame_Lab_1Kt

```
"C:\Program Files\Java\jdk1.8.0_144\bin\java" ...
Think of an animal. I will try to guess which animal you are thinking of.
I will ask some questions. Please, answer with a YES or NO.
Can it fly?NO
Is it Elephant?
NO
Ok. I give up. What is it?
Tiger
What question should have I asked?
Is it a wild animal?
Ok. Got it.
Do you want to have another go?
YES
Can it fly?NO
Is it a wild animal?YES
Is it Tiger?
YES
Woohoo! I got the right answer in 2 steps.
Do you want to have another go?
```

LIMITATION

The above program has the following limitations:

1. The program can only read ‘YES’ or ‘NO’ value while traversing the tree. However, it can read other text when learning from the user.
2. The program cannot differentiate between duplicate data (both question and answer.)
3. The program does not store new information in secondary storage. Hence, it will not retain the information when terminated.