**DEERWALK INSTITUTE OF TECHNOLOGY**

**Lab 1**

**Artificial Intelligence**

**Submitted to:**

**Birod Rijal**

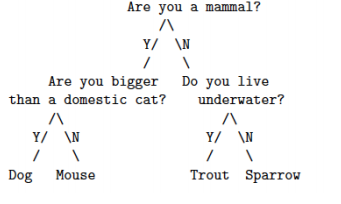
**Submitted By:**

**Umesh Dhungana**

**0342**

# Problem Definition:

This is the program to implement an ‘Animal Guessing Game’. This is the game where user (player) thinks of an animal and computer tries to make guess of an animal that the player is thinking about by asking some questions that can be answered by ‘yes’ or ‘no’. For example, if a user is thinking of mouse then the computer will generate the question at the root node “Are you a mammal?” as the first question. Then user chooses either yes or no as an answer. Then again computer generates other questions based on the response. The game continues until the leaf node is reached. At the leaf node, the computer will generate the guess. In this example we consider dog, mouse, trout and sparrow.



# Methodology

The concept of the binary tree is the main logic behind the implementation of this game, where we have created the parent child relationship using the node. A binary tree is a data structure where a node has at most two children, usually referred to as the “left" child and the “right" child. Each node can store some information as well.

## Data Structure

Linked list is the preferred data structure and Java is the programming language that is used to implement the Animal guessing game. So, this problem is solved form object oriented approach. Node object contains the string value or question, its associated id and its left and right node.

There is a root node and if player selects yes, then left child is selected else right child is selected.

**public class** Node {  
 Node **leftChild**,**rightChild**;  
 Integer **id**;  
 String **question**;  
 **public** Node(String question, **int** id) {  
 **this**.**question** = question;  
 **this**.**id** = id;  
 }  
}

The flow of the program is as:

Creation of the rootNode:

**public** Node addRootNode(String question,**int** id)  
{  
 Node rootNode=**new** Node(question,id);  
 **return** rootNode;  
}

There are two functions addLeftNode() and addRightNode() that add nodes to left child and right child respectively.

Creation and addition of the nodes to the left child

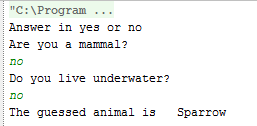
**public void** addLeftNode(Node rootNode, **int** parentId, **int** id, String question) {  
 **if** (rootNode.**id** == parentId) {  
 Node newNode = **new** Node(question, id);  
 rootNode.**leftChild** = newNode;  
 **return**;  
 }  
 **else** {  
 **if** (rootNode.**leftChild** != **null**) {  
 addLeftNode(rootNode.**leftChild**, parentId, id, question);  
 }  
 **if** (rootNode.**rightChild** != **null**) {  
 addLeftNode(rootNode.**rightChild**, parentId, id, question);  
 }  
 }  
}

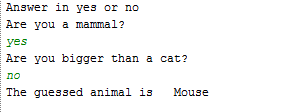
Creation and addition of the nodes to the right child

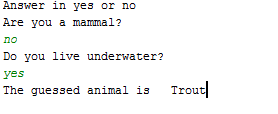
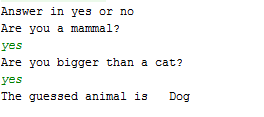
**public void** addRightNode(Node rootNode, **int** parentId, **int** id, String question) {  
 **if** (rootNode.**id** == parentId) {  
 rootNode.**rightChild** = **new** Node(question, id);  
 **return**;  
 } **else** {  
 **if** (rootNode.**leftChild** != **null**) {  
 addRightNode(rootNode.**leftChild**, parentId, id, question);  
 }  
 **if** (rootNode.**rightChild** != **null**)  
 addRightNode(rootNode.**rightChild**, parentId, id, question);  
 }  
}

Then the questions are set accordingly. On the basis of the response given by player the node is selected. The expansion is carried until the leaf node is reached. checkLeaf() is the function to check whether the current node is leaf node or not.

# Output





# Analysis

The question is asked starting with the question in the root node (i.e. “Are you a mammal?”). Then according to the response (yes or no) other questions are asked. If the response is yes left sub tree is traversed else right sub tree is traversed. After reaching to the leaf node we stop and the animal is found.

# Annex

The implementation of this problem using Java is given below:

*/\*\*  
 \* Created by UMESH on 4/23/2016.  
 \*/***import** java.util.Scanner;  
**public class** AnimalGuessing {  
  
 *// Node is a class that contains information about a node.* **public class** Node {  
 Node **leftChild**,**rightChild**;  
 Integer **id**;  
 String **question**;  
 **public** Node(String question, **int** id) {  
 **this**.**question** = question;  
 **this**.**id** = id;  
 }  
 }  
  
 *//This function is used to add root Node* **public** Node addRootNode(String question,**int** id)  
 {  
 Node rootNode=**new** Node(question,id);  
 **return** rootNode;  
 }  
  
 *//This functions creates and add the left child to the node for the given id* **public void** addLeftNode(Node rootNode, **int** parentId, **int** id, String question) {  
 **if** (rootNode.**id** == parentId) {  
 Node newNode = **new** Node(question, id);  
 rootNode.**leftChild** = newNode;  
 **return**;  
 }  
 **else** {  
 **if** (rootNode.**leftChild** != **null**) {  
 addLeftNode(rootNode.**leftChild**, parentId, id, question);  
 }  
 **if** (rootNode.**rightChild** != **null**) {  
 addLeftNode(rootNode.**rightChild**, parentId, id, question);  
 }  
 }  
 }  
  
 *//This functions creates and add the left child to the node for the given id* **public void** addRightNode(Node rootNode, **int** parentId, **int** id, String question) {  
 **if** (rootNode.**id** == parentId) {  
 rootNode.**rightChild** = **new** Node(question, id);  
 **return**;  
 } **else** {  
 **if** (rootNode.**leftChild** != **null**) {  
 addRightNode(rootNode.**leftChild**, parentId, id, question);  
 }  
 **if** (rootNode.**rightChild** != **null**)  
 addRightNode(rootNode.**rightChild**, parentId, id, question);  
 }  
 }  
  
 **public static void** main(String[] args) {  
  
 AnimalGuessing addNode = **new** AnimalGuessing();  
 *//Root node is created as* Node rootNode=addNode.addRootNode(**"Are you a mammal?"**,0);  
  
 *//The nodes are added to the tree as* addNode.addLeftNode(rootNode, 0, 1, **"Are you bigger than a cat?"**);  
 addNode.addLeftNode(rootNode, 1, 3, **"Dog"**);  
 addNode.addRightNode(rootNode, 1, 4, **"Mouse"**);  
  
 addNode.addRightNode(rootNode, 0, 2, **"Do you live underwater?"**);  
 addNode.addLeftNode(rootNode, 2, 5, **"Trout"**);  
 addNode.addRightNode(rootNode, 2, 6, **"Sparrow"**);  
  
 System.***out***.println(**"Answer in yes or no"**);  
 *//rootnode is passed to startGame  
 startGame*(rootNode);  
 }  
  
 *//This function check if the given node is leaf node or not* **public static boolean** checkLeaf(Node node)  
 {  
 **boolean** isLeaf=**false**;  
 **if**(node.**leftChild**==**null** && node.**rightChild**==**null**)  
 {  
 isLeaf=**true**;  
 }  
 **return** isLeaf;  
 }  
  
 **public static void** startGame(Node node) {  
 **while** (**true**) {  
  
 Scanner input = **new** Scanner(System.***in***);  
 System.***out***.println(node.**question**);  
 String answer = input.nextLine().toLowerCase();  
 **if** (answer.equals(**"yes"**)) {  
 node=node.**leftChild**;  
 } **else if** (answer.equals(**"no"**)) {  
 node=node.**rightChild**;  
 } **else** {  
 System.***out***.println(**"Invalid entry! Enter either yes or no only."**);  
 }  
 **if**(*checkLeaf*(node) == **true**)  
 {  
 System.***out***.println(**"The guessed animal is \t"**+node.**question**);  
 **break**;  
 }  
 }  
 }  
}