**Problem Definition**

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. A k-ary tree is a tree in which each node has no more than k children. It is also sometimes known as a k-way tree, an N-ary tree, or an M-ary tree. A binary tree is the special case where k=2. Having two children left and right is the standard form of binary tree. Any k-ary tree can be represented as a binary tree where in each node the left pointer points to the first child and the right pointer points to the next sibling. Such a tree has been given the name left-child, right-sibling binary tree (LC-RS binary tree).

Above mentioned is the backbond concept behind the Animal –guessing game. In this game, the player thinks of an animal and the computer tries to guess which animal the player is imagining by asking questions which can be answered `yes' or `no'. To begin each round of the game, you think of an animal. For example, suppose you are thinking of a dog. The computer will generate the question at the root node “Are you a mammal?" as the first question. The user has to press 'y' or 'n' (1 or 0) as the response. Based on the response the computer will generate the second question. The game continues until the leaf node is reached. At the leaf node the computer will generate the guess (dog, mouse, trout or sparrow in our example).

**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. I have created one data structure named Node, which consist of Node left Child, Node right Child, id and a question, this indicates that every node in the binary tree will have left Child, right Child, id and one question as an information.

The code is given below.

This Structure is being implemented in the next class “AnimalFinder”.

*\*\*  
 \* Created by asmita on 4/25/2016.  
 \*/***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 main program “AnimalFInder” that implemented the structure that we have crated above is explained below.

**Creation of the rootNode.**

**public class** AnimalFinder {  
 *//This function create the root node by passing question as a information in the root node and its id*

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

**Creation and addition of the left child to the node for the given id.**

**public void** aadLeftNode(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**){  
 aadLeftNode(rootNode.**leftChild**, parentId, id, question);  
 }  
 **if**(rootNode.**rightChild**!=**null**){  
 aadLeftNode(rootNode.**rightChild**,parentId,id,question);  
 }  
 }  
}

**Creation and addition of the right child to the node for the given id.**

*//This function create and add the rightChild with the given id***public void** aadRightNode(Node rootNode, **int** parentId, **int** id, String question){  
 **if**(rootNode.**id**==parentId){  
 Node newNode =**new** Node(question,id);  
 rootNode.**rightChild**=newNode;  
 **return**;  
 }  
 **else**{  
 **if**(rootNode.**leftChild**!=**null**){  
 aadRightNode(rootNode.**leftChild**,parentId,id,question);  
 }  
 **if**(rootNode.**rightChild**!=**null**){  
 aadRightNode(rootNode.**rightChild**,parentId,id,question);  
 }  
 }  
}

**Main class**

Here the object addNode has been created so that we can call all the functions that are used in our programs. Here respective values to the parameter ( id to the nodes, questions/names(as a information) are added here below.

**public static void** main(String[] args) {  
 AnimalFinder addNode= **new** AnimalFinder();  
 Node rootN=addNode.aadRootNode(**"Are you a mammel"**, 0);(  
 addNode.aadLeftNode(rootN,0,1,**"Are you bigger than cat"**);  
 addNode.aadLeftNode(rootN,1,3,**"DOG"**);  
 addNode.aadRightNode(rootN, 1, 4, **"Mouse"**);  
 addNode.aadRightNode(rootN,0,2,**"do you live under water?"**);  
 addNode.aadRightNode(rootN,2,5,**"Trout"**);  
 addNode.aadRightNode(rootN,2,6,**"Sparrow"**);  
 System.***out***.println(**"Press 1 for yes and 0 for no"**);  
 *playGame*(rootN);//function called  
  
}

**Function to distinguish leafNode**

In this function

**public static boolean** isLeafNode(Node node){  
 **boolean** isLeaf=**false**;  
 **if**(node.**leftChild**==**null**&&node.**rightChild**==**null**){  
 isLeaf=**true**;  
 }  
 **return** isLeaf;  
}

**Function that implements the game**

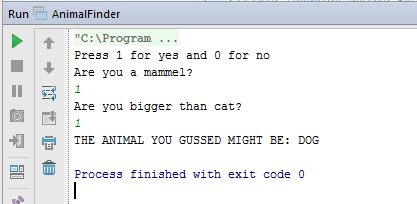
Here the 1st while loop runs until the break is initiated. If else loop is executed on the basis of the id provided to it. If id==1 it goes to the left node and if 0 then it moves to the right node. The final answer is obtained from the leafNode.

**public static void** playGame(Node node){

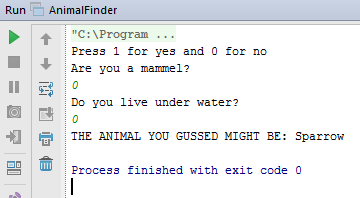
**while**(**true**){  
 Scanner input= **new** Scanner(System.***in***);  
 System.***out***.println(node.**question**);  
 **int** answer= input.nextInt();  
 **if**(answer==1){  
 node=node.**leftChild**;  
 }  
 **else if**(answer==0){  
 node=node.**rightChild**;  
 }  
 **else**{  
 System.***out***.println(**"Invalid Entery"**);  
 }  
 **if**(*isLeafNode*(node)==**true**){  
 System.***out***.println(**"The animal you gussed is"**+ node.**question**);  
 **break**;  
 }  
 }  
}

**Output**

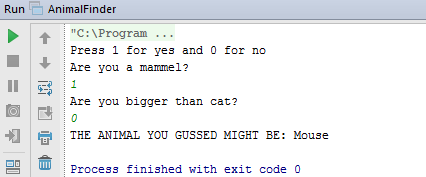
**Type 1: If we are mammal and bigger than cat.**



**Type 2: If we are not mammal and do not live under water.**



**Type 3: If we are mammal and smaller than dog.**



**Analysis**

Once the program is executed we are asked to press 1 for ‘yes’ and 0 for ‘no’. The output or the final result is totally based on value we have assign to it i.e ‘yes’ or ‘no’. The first question asked is “Are you a mammal?, which lies on the root node. After that program runs further by analyzing our ‘yes’ or ‘no’ answer. If we press ‘1’ it traverse the left child else if 0 it traverse the right child.

This is how overall program is being executed which guesses the different animal by traversing left and right node with the help of binary tree technique.