**APP Project**

“AI” Game Using Binary Tree in JAVA

Team Members:

Anouska Jhunjhunwala[RA2211003010637]

Vatsal Sinha[RA2211003010641]

**Input File:  
animal\_game.txt**

Animal Guessing Game

You're asked to guess an animal by answering a series of questions.

200 Is it a mammal? (Y/N)

180 Does it have fur? (Y/N)

160 It's a dog

190 It's a cat

220 Is it a tall animal? (Y/N)

210 It's a giraffe

230 It's a kangaroo

**Source Code:**

**AnimalGuessingGame.java**

import java.io.BufferedReader;

import java.io.FileReader;

import java.io.IOException;

import java.util.Scanner;

class Node {

int data;

String question;

Node yesLink;

Node noLink;

public Node(int data, String question) {

this.data = data;

this.question = question;

this.yesLink = null;

this.noLink = null;

}

}

public class AnimalGuessingGame {

private Node root;

public AnimalGuessingGame(String filename) {

root = loadGame(filename);

}

private Node loadGame(String filename) {

Node currentNode = null;

try {

BufferedReader reader = new BufferedReader(new FileReader(filename));

String line;

Node root = null;

while ((line = reader.readLine()) != null) {

if (line.matches("\\d+ .\*")) {

String[] parts = line.split(" ", 2);

int data = Integer.parseInt(parts[0]);

String question = parts[1].trim();

Node newNode = new Node(data, question);

if (currentNode == null) {

root = newNode;

} else {

if (data % 10 == 0) {

currentNode.yesLink = newNode;

} else {

currentNode.noLink = newNode;

}

}

currentNode = newNode;

} else {

System.out.println(line);

}

}

reader.close();

return root;

} catch (IOException e) {

e.printStackTrace();

return null;

}

}

public void playGame(Scanner scanner) {

if (root == null) {

System.out.println("Game data is missing or corrupted. Please load a valid game file.");

return;

}

Node currentNode = root;

boolean gameFinished = false;

while (!gameFinished) {

System.out.println(currentNode.question);

if (currentNode.yesLink == null && currentNode.noLink == null) {

System.out.println("The animal is: " + currentNode.question);

System.out.println("Do you want to play again? (Y/N)");

String playAgain = getUserAnswer(scanner);

if (playAgain.equalsIgnoreCase("Y") || playAgain.equalsIgnoreCase("Yes")) {

currentNode = root;

} else {

gameFinished = true;

}

} else {

String answer = getUserAnswer(scanner);

if (answer.equalsIgnoreCase("Y") || answer.equalsIgnoreCase("Yes")) {

if (currentNode.yesLink != null) {

currentNode = currentNode.yesLink;

} else {

System.out.println("Invalid answer. Please answer with 'Y' or 'N'.");

}

} else if (answer.equalsIgnoreCase("N") || answer.equalsIgnoreCase("No")) {

if (currentNode.noLink != null) {

currentNode = currentNode.noLink;

} else {

System.out.println("Invalid answer. Please answer with 'Y' or 'N'.");

}

} else {

System.out.println("Invalid answer. Please answer with 'Y' or 'N'.");

}

}

}

}

private String getUserAnswer(Scanner scanner) {

String answer = scanner.nextLine().trim();

return answer;

}

private void displayInOrder(Node node) {

if (node != null) {

displayInOrder(node.yesLink);

System.out.print(node.data + " " + node.question);

if (node.yesLink != null || node.noLink != null) {

System.out.print(" (");

if (node.yesLink != null) {

System.out.print("Yes: " + node.yesLink.question);

}

if (node.noLink != null) {

if (node.yesLink != null) {

System.out.print(", ");

}

System.out.print("No: " + node.noLink.question);

}

System.out.print(")");

}

System.out.println();

displayInOrder(node.noLink);

}

}

private void displayPreOrder(Node node) {

if (node != null) {

System.out.print(node.data + " " + node.question);

if (node.yesLink != null || node.noLink != null) {

System.out.print(" (");

if (node.yesLink != null) {

System.out.print("Yes: " + node.yesLink.question);

}

if (node.noLink != null) {

if (node.yesLink != null) {

System.out.print(", ");

}

System.out.print("No: " + node.noLink.question);

}

System.out.print(")");

}

System.out.println();

displayPreOrder(node.yesLink);

displayPreOrder(node.noLink);

}

}

private void displayPostOrder(Node node) {

if (node != null) {

displayPostOrder(node.yesLink);

displayPostOrder(node.noLink);

System.out.print(node.data + " " + node.question);

if (node.yesLink != null || node.noLink != null) {

System.out.print(" (");

if (node.yesLink != null) {

System.out.print("Yes: " + node.yesLink.question);

}

if (node.noLink != null) {

if (node.yesLink != null) {

System.out.print(", ");

}

System.out.print("No: " + node.noLink.question);

}

System.out.print(")");

}

System.out.println();

}

}

public static void main(String[] args) {

AnimalGuessingGame game = new AnimalGuessingGame("animal\_game.txt");

Scanner scanner = new Scanner(System.in);

while (true) {

System.out.println("Animal Guessing Game");

System.out.println("P Play the game");

System.out.println("L Load another game file");

System.out.println("D Display the binary tree");

System.out.println("H Help information");

System.out.println("X Exit the program");

System.out.print("Your choice: ");

String choice = scanner.nextLine();

if (choice.equalsIgnoreCase("P")) {

game.playGame(scanner);

} else if (choice.equalsIgnoreCase("L")) {

System.out.print("Enter the game file name: ");

String filename = scanner.nextLine();

game = new AnimalGuessingGame(filename);

} else if (choice.equalsIgnoreCase("D")) {

System.out.println("What order do you want to display?");

System.out.println("1. In-order");

System.out.println("2. Pre-order");

System.out.println("3. Post-order");

System.out.println("4. Return to the main menu");

System.out.print("Your choice: ");

String displayChoice = scanner.nextLine();

if (displayChoice.equals("1")) {

System.out.println("In-order Traversal:");

game.displayInOrder(game.root);

} else if (displayChoice.equals("2")) {

System.out.println("Pre-order Traversal:");

game.displayPreOrder(game.root);

} else if (displayChoice.equals("3")) {

System.out.println("Post-order Traversal:");

game.displayPostOrder(game.root);

} else if (displayChoice.equals("4")) {

continue;

} else {

System.out.println("Invalid choice. Please select a valid option.");

}

} else if (choice.equalsIgnoreCase("H")) {

try (BufferedReader br = new BufferedReader(new FileReader("animal\_game.txt"))) {

br.readLine();

String helpInformation = br.readLine();

System.out.println(helpInformation);

} catch (IOException e) {

System.out.println("Error reading help information: " + e.getMessage());

}

} else if (choice.equalsIgnoreCase("X")) {

System.out.println("Exiting the program. Goodbye!");

break;

} else {

System.out.println("Invalid choice. Please enter a valid option.");

}

}

}

}

**Screenshots:**

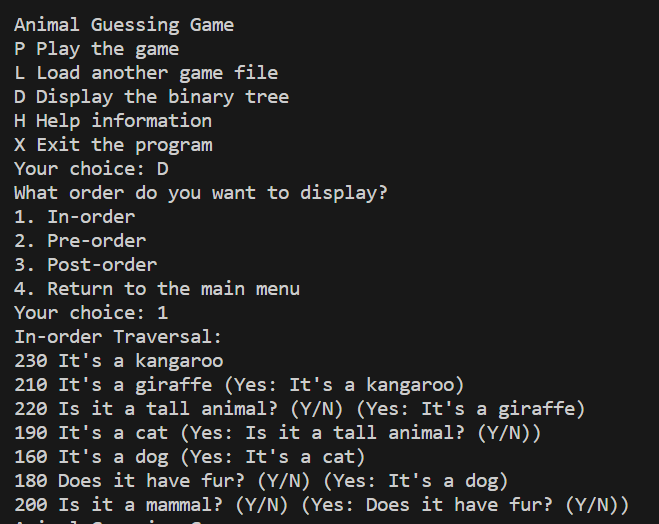


Fig 1. In-sorder Traversal

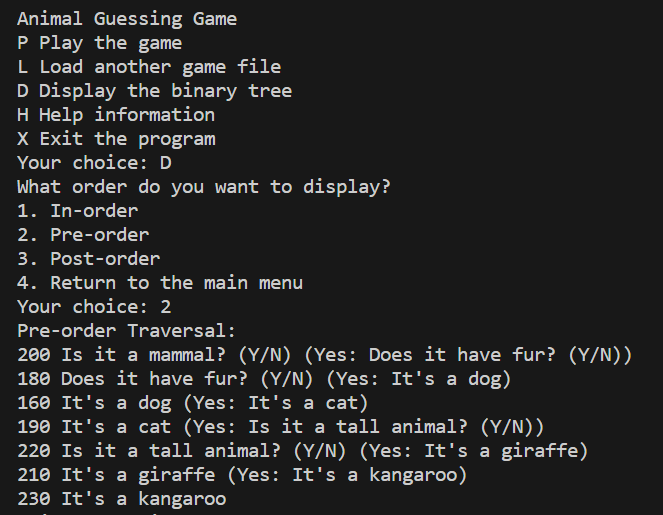


Fig 2. Preorder Traversal

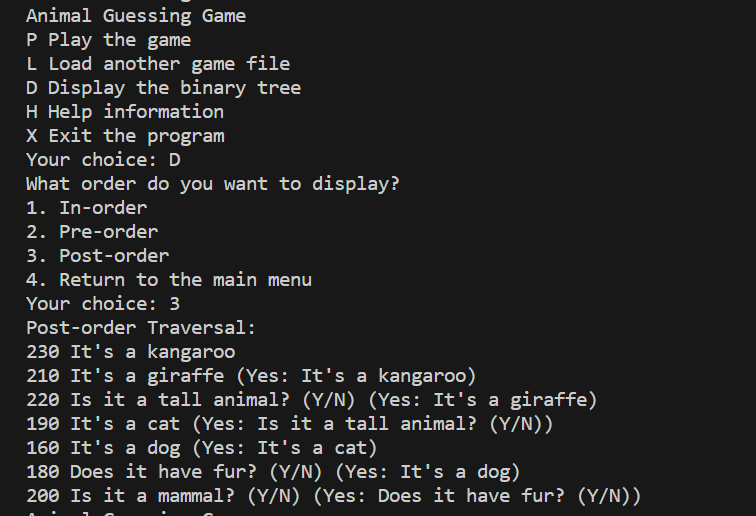


Fig 3. Post-order Traversal

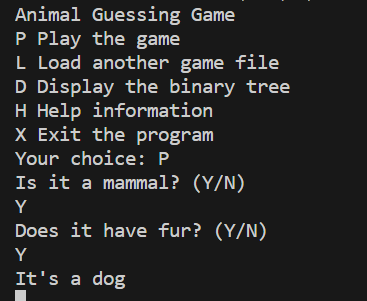


Fig 4. Animal Guessing Game