package newproject;

/\*\*

 \* parser!

 \*

 \*/

import com.github.javaparser.JavaParser;

import com.github.javaparser.ast.CompilationUnit;

import com.github.javaparser.ast.Node;

import java.io.FileInputStream;

import java.io.IOException;

import java.util.HashMap;

import java.util.Map;

public class App {

    public static void main(String[] args) {

        try {

            FileInputStream file = new FileInputStream("C:\\Users\\B-TECH\\Documents\\newproj\\src\\test\\java\\newproject\\STUDENT.java");

            CompilationUnit cu = new JavaParser().parse(file).getResult().orElseThrow(() -> new RuntimeException("Unable to parse the file"));

            // Print AST recursively

            printAST(cu, 0);

            // Get token counts

            Map<String, Integer> tokenCounts = getTokenCounts(cu);

            System.out.println("\nToken Counts:");

            tokenCounts.forEach((token, count) -> System.out.println(token + ": " + count));

            // Calculate total token count

            int totalTokens = getTotalTokenCount(tokenCounts);

            System.out.println("\nTotal Tokens: " + totalTokens);

        } catch (IOException e) {

            e.printStackTrace();

        }

    }

    // Function to print AST recursively

    private static void printAST(Node node, int depth) {

        System.out.println(" ".repeat(depth \* 2) + node.getClass().getSimpleName());

        node.getChildNodes().forEach(child -> printAST(child, depth + 1));

    }

    // Function to get token counts

    private static Map<String, Integer> getTokenCounts(CompilationUnit cu) {

    Map<String, Integer> tokenCounts = new HashMap<>();

    cu.getTokenRange().ifPresent(range -> range.forEach(token -> {

        String tokenType = token.getCategory().name();

            tokenCounts.put(tokenType, tokenCounts.getOrDefault(tokenType, 0) + 1);

        }));

        return tokenCounts;

    }

    // Function to calculate total token count

    private static int getTotalTokenCount(Map<String, Integer> tokenCounts) {

        return tokenCounts.values().stream().mapToInt(Integer::intValue).sum();

    }

}