Northeastern University
Khoury College of Computer
and Information Sciences

Plagus – Plagiarism Detector For Source Code

CS5500 – Group 6

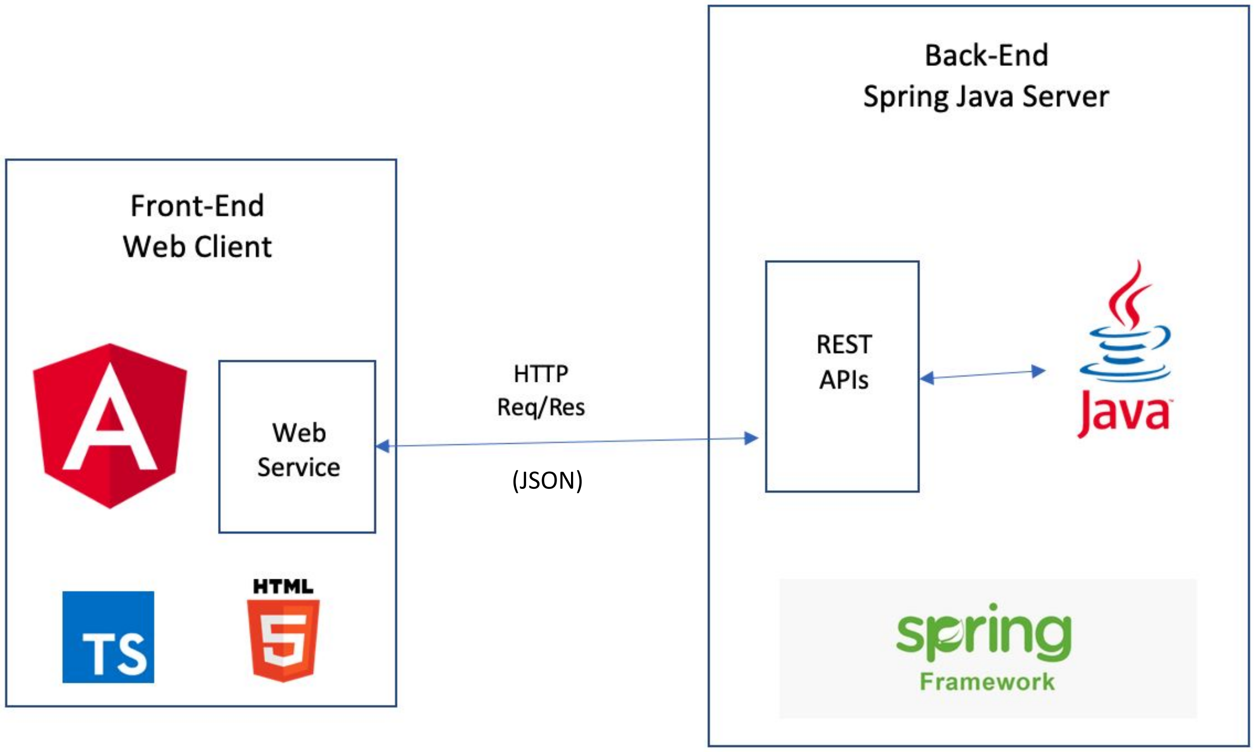
Hassan Khan
Samuel Raphael
Yichuan Philip Ma
Pierre-Alexandre Mousset

Overview of Plagus

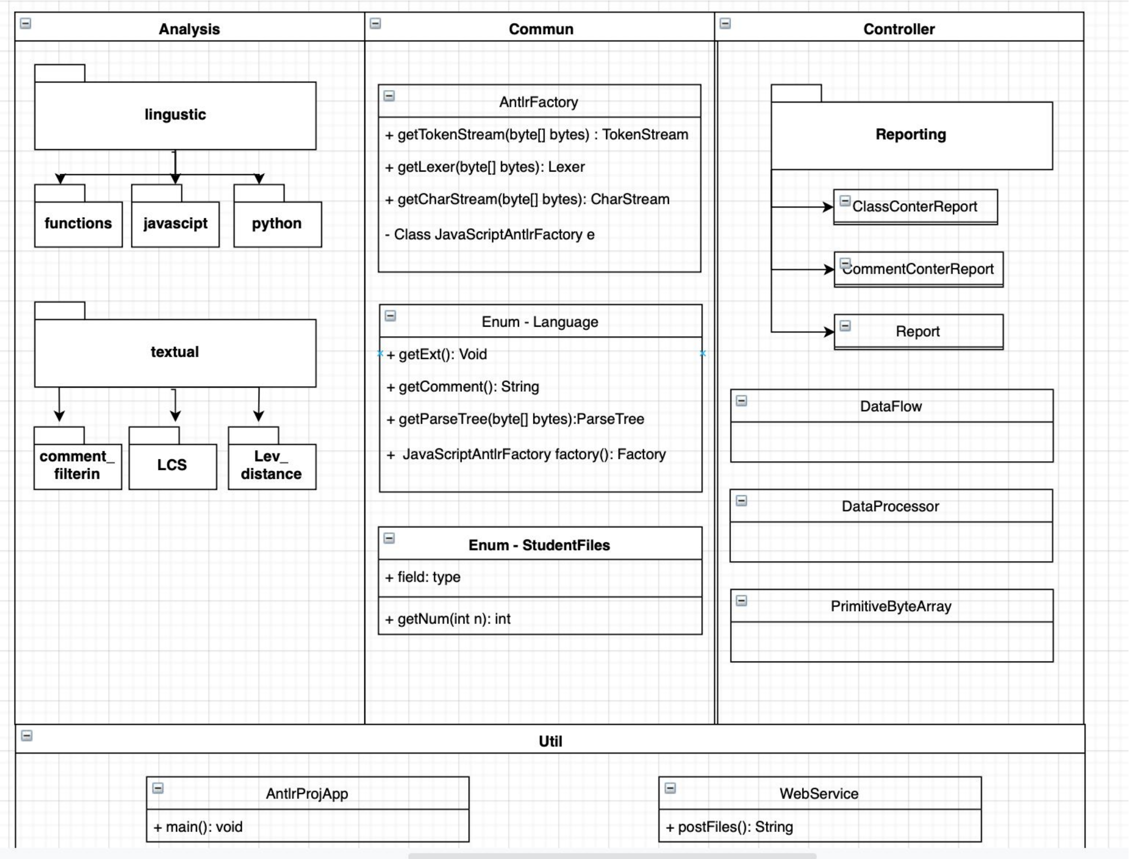
- Detect plagiarism in Python 3 and JavaScript source files from two students
- Use of Spring Framework for the server that handles requests from the front-end
- Use of Angular for the front-end UI

Live Demo

System Architecture



System Architecture - Backend



Analysis

lingustic

functions

javascipt

python

textual

comment_ filterin

LCS

Lev_ distance

Commun

AntlrFactory

+ getTokenStream(byte[] bytes) : TokenStream

+ getLexer(byte[] bytes): Lexer

+ getCharStream(byte[] bytes): CharStream

- Class JavaScriptAntlrFactory e

Enum - Language

+ getExt(): Void

+ getComment(): String

+ getParseTree(byte[] bytes):ParseTree

+ JavaScriptAntlrFactory factory(): Factory

Enum - StudentFiles

+ field: type

+ getNum(int n): int

Controller

Reporting

ClassConterReport

CommentConterReport

Report

DataFlow

DataProcessor

PrimitiveByteArray

Util

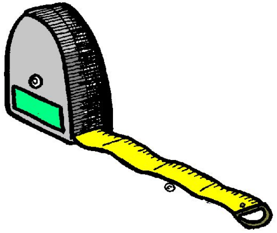
AntlrProjApp

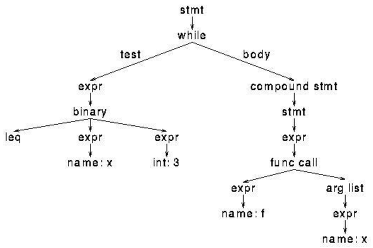
+ main(): void

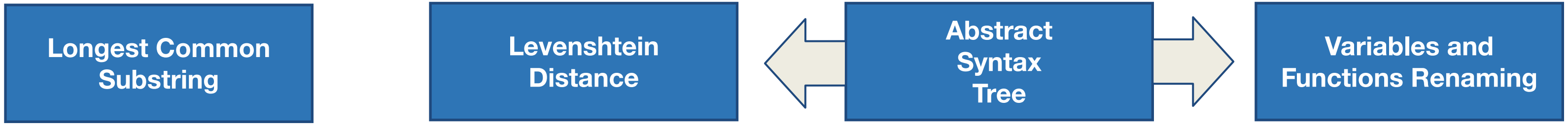
WebService

+ postFiles(): String

Algorithms - Overview







Longest Common Substring

Levenshtein Distance

Abstract Syntax Tree

Variables and Functions Renaming

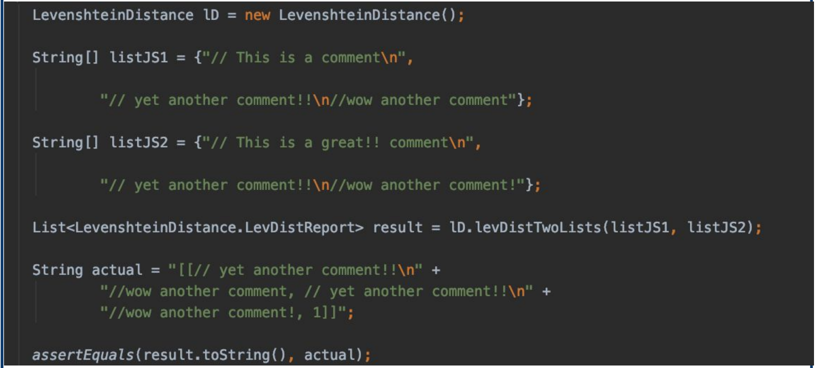
I- Levenshtein Distance

- Why?
    - We used Levenshtein distance to detect closeness between two strings
    - Levenshtein distance is the number of different characters between the two strings
    - The lower the distance, the more similar the two strings are
    - A distance of 0 means that two strings are completely identical

- Where we used it:
    - Functions
    - Comments

Example

File A



```
LevenshteinDistance lD = new LevenshteinDistance();

String[] listJS1 = {"// This is a comment\n",

        "// yet another comment!!\n//wow another comment"};

String[] listJS2 = {"// This is a great!! comment\n",

        "// yet another comment!!\n//wow another comment!"};

List<LevenshteinDistance.LevDistReport> result = lD.levDistTwoLists(listJS1, listJS2);

String actual = "[[// yet another comment!!\n" +
        "//wow another comment, // yet another comment!!\n" +
        "//wow another comment!, 1]]";

assertEquals(result.toString(), actual);
```

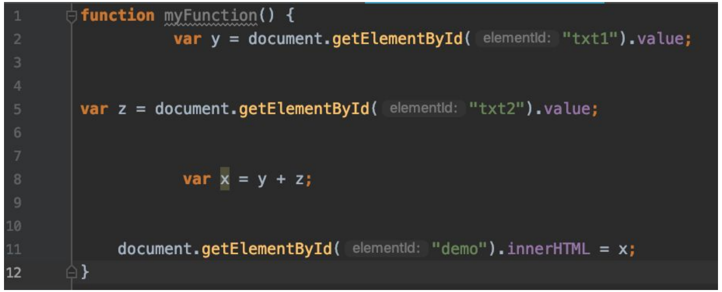
2 - LCS

- Why
  - A source code file is lexed into tokens
  - We are applying the LCS on a list of Tokens

- How we implemented it
  - Filter by using getType() method for each token: white spaces, new lines, string value…

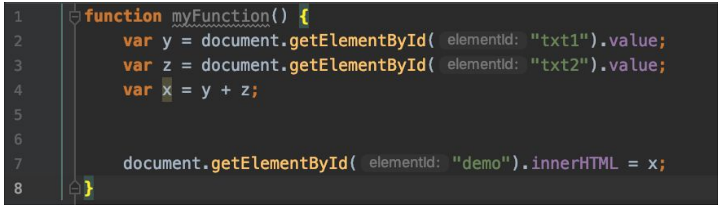
Example

File A



```
1   function myFunction() {
2            var y = document.getElementById( elementId: "txt1").value;
3
4
5   var z = document.getElementById( elementId: "txt2").value;
6
7
8             var x = y + z;
9
10
11      document.getElementById( elementId: "demo").innerHTML = x;
12  }
```

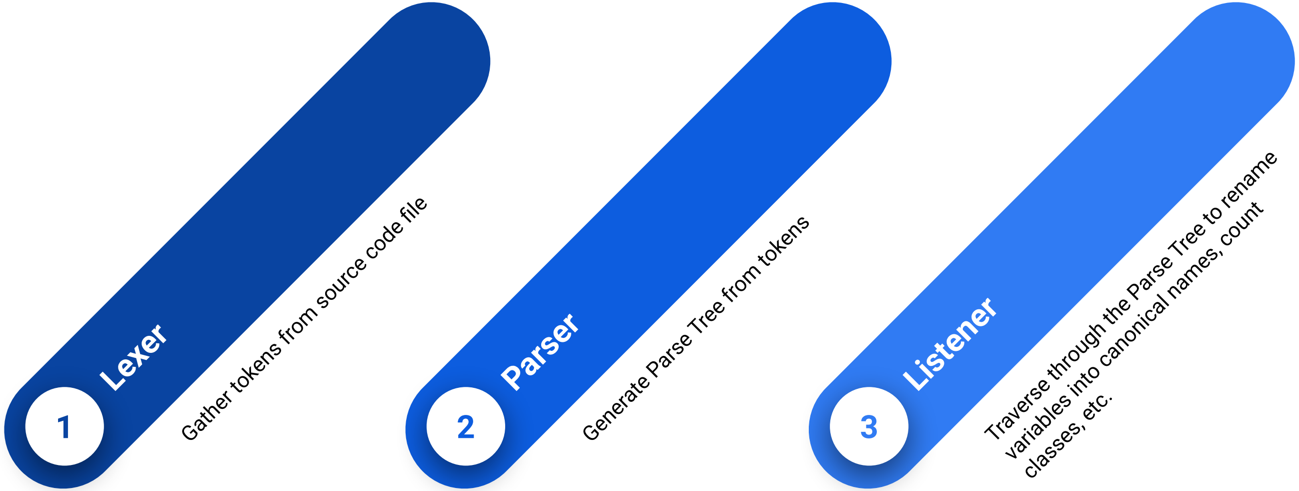
File B



```
1  function myFunction() {
2    var y = document.getElementById( elementId: "txt1").value;
3    var z = document.getElementById( elementId: "txt2").value;
4    var x = y + z;
5
6
7    document.getElementById( elementId: "demo").innerHTML = x;
8  }
```

File A - results	**File B - results**
Starting Line: 1	Starting Line: 1
StartingIndexOnLine: 0	StartingIndexOnLine: 0
Ending Line: 12	Ending Line: 8
StartingIndexOnLine: 0	StartingIndexOnLine: 0

3 - AST



1 Lexer

Gather tokens from source code file

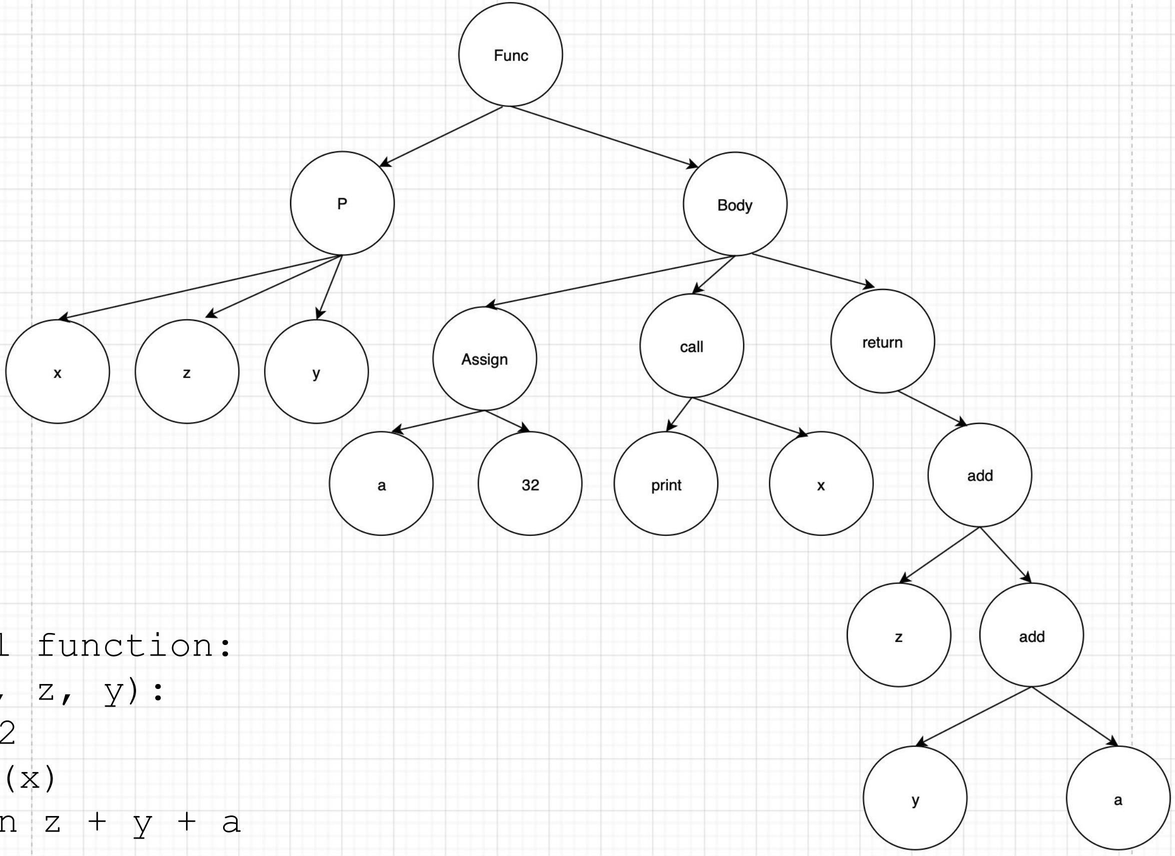
2 Parser

Generate Parse Tree from tokens

3 Listener

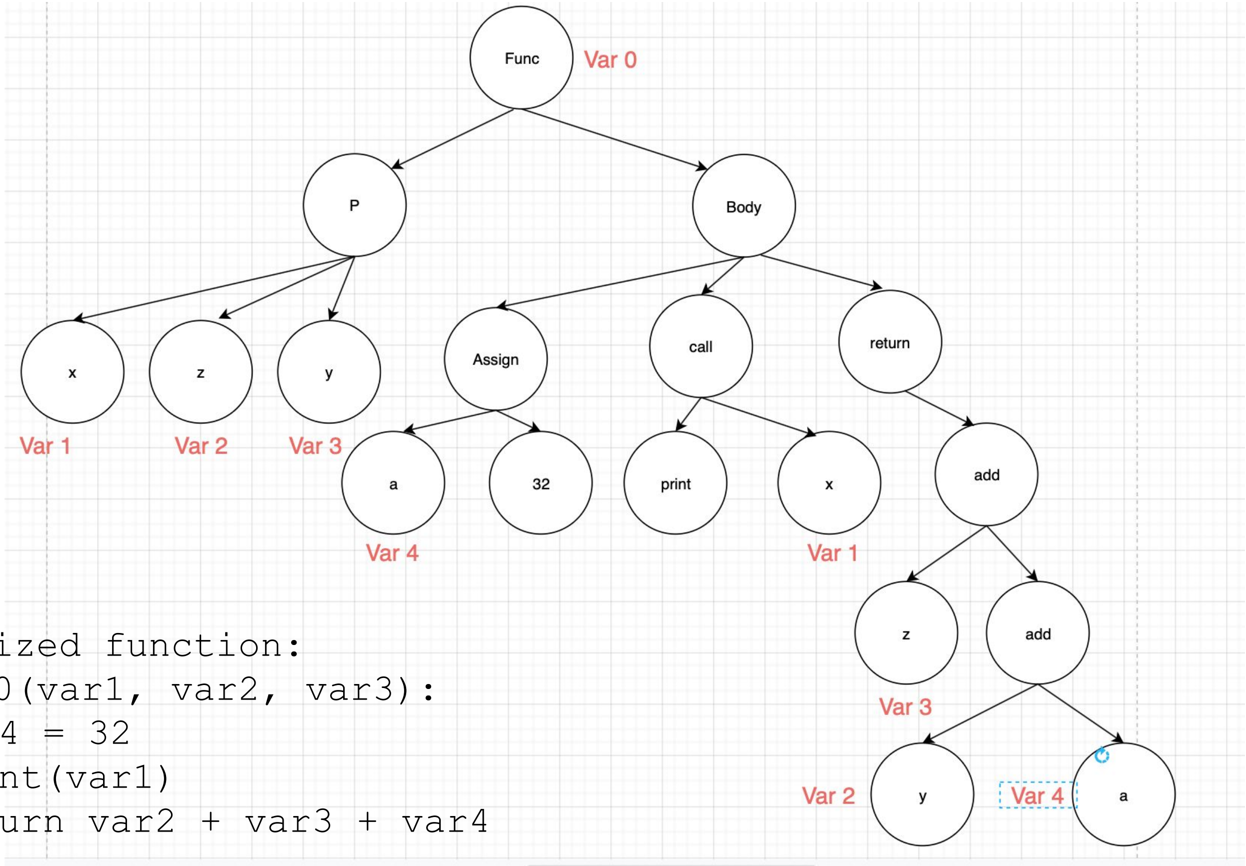
Traverse through the Parse Tree to rename variables into canonical names, count classes, etc.

3 - AST Example (1/2)



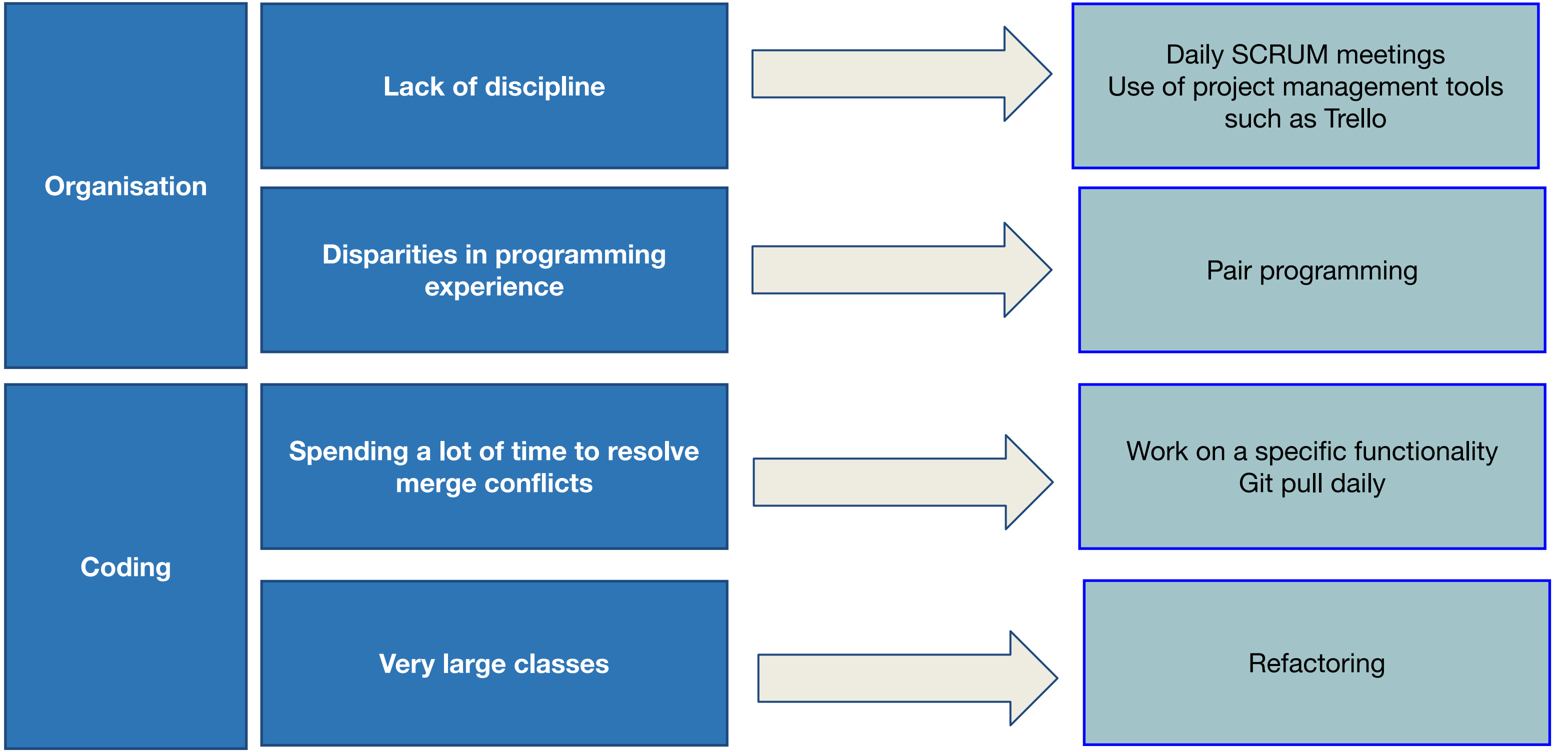
```
# Original function:
def foo(x, z, y):
    a = 32
    print(x)
    return z + y + a
```

3 - AST Example (2/2)



```
# Canonized function:
def var0(var1, var2, var3):
    var4 = 32
    print(var1)
    return var2 + var3 + var4
```

Software Engineering practices



Organisation	Lack of discipline	Daily SCRUM meetings Use of project management tools such as Trello
	Disparities in programming experience	Pair programming
Coding	Spending a lot of time to resolve merge conflicts	Work on a specific functionality Git pull daily
	Very large classes	Refactoring

Things to improve

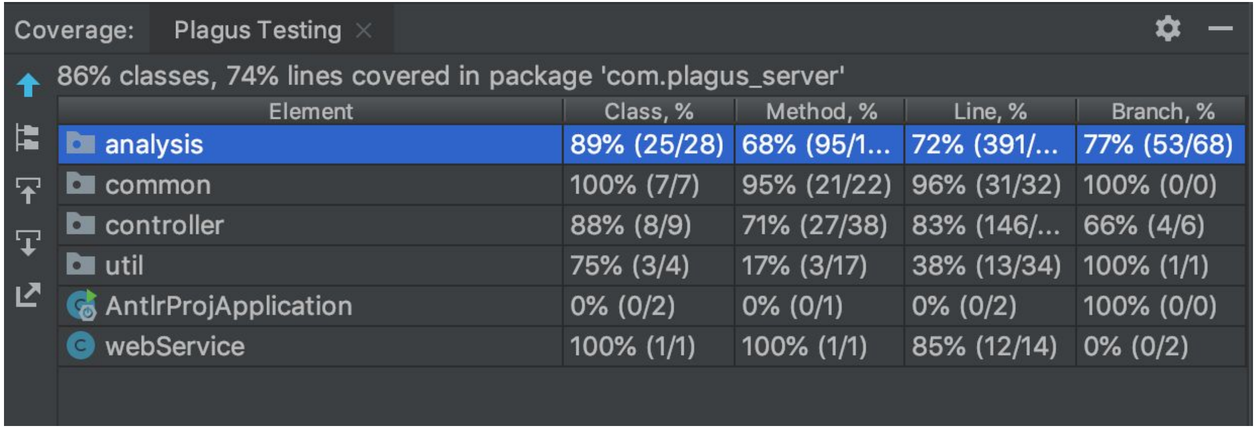
Testing

Documentation

Starting earlier

Communication between teammates!

Testing



Coverage: Plagus Testing

86% classes, 74% lines covered in package 'com.plagus_server'

Element	Class, %	Method, %	Line, %	Branch, %
analysis	89% (25/28)	68% (95/1...	72% (391/...	77% (53/68)
common	100% (7/7)	95% (21/22)	96% (31/32)	100% (0/0)
controller	88% (8/9)	71% (27/38)	83% (146/...	66% (4/6)
util	75% (3/4)	17% (3/17)	38% (13/34)	100% (1/1)
AntlrProjApplication	0% (0/2)	0% (0/1)	0% (0/2)	100% (0/0)
webService	100% (1/1)	100% (1/1)	85% (12/14)	0% (0/2)

Thank You from Plagus!

Any questions?

Hassan Khan
Samuel Raphael
Yichuan Philip Ma
Pierre-Alexandre Mousset