Comparing files with Python

By Víctor Moreno

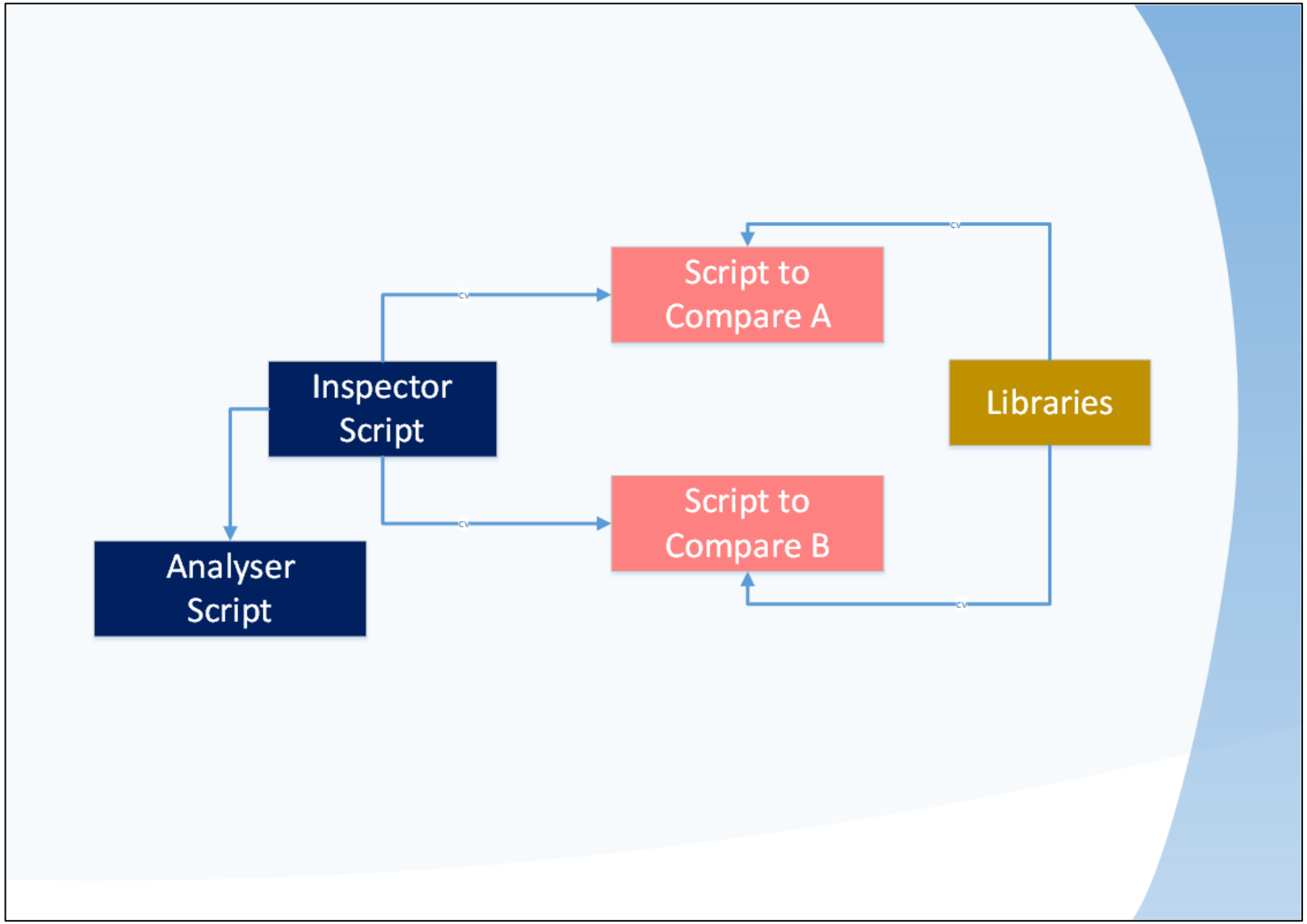
About Project

Goal

The goal of this Project is build a script in Python that allows to compare two files and show the percentage of similarity between them.

With this functionality we can have a visibility for scripts working and doing similar tasks.

Architecture



Architecture

Inspector

Is the script that makes the comparison between files. Receive the keywords and path files.

Analyser

Is the script that contains the class "Analyser" with his corresponding methods for compare text between files.

Scripts to Compare (A and B)

They are the two files for compare the percentage of similarity.

Libraries

They are other classes for the scripts to compare, this libraries are used simulating real scripts written in Python.

Scripts to compare

We have six scripts to compare, they are:

- ValidateActions.py
  - This script allow us know if exist permission to alter objets inside databases.
- ValidateFields.py
  - This script validate fields in tables and it return values for each result.
- ValidateLoads.py
  - This script send workloads for tables (thousands of records).
- ValidateLoads_2.py
  - This script its similar than ValidateLoads.py
- ValidateStress.py
  - This script simulate connections for the server (thousands of connections).
- ValidateStress_2.py
  - This script its similar than ValidateStress.py

Libraries

We have six scripts as libraries, they are:

- DatabaseConnect.py
  - This script give us methods for connect with databases.
- FunctionalTesting.py
  - This script give us methods for validate fields in tables.
- LoadTesting.py
  - This script give us methods for send, get, delete and update workloads.
- SecurityTesting.py
  - This script give us methods for know if users can alter objects in the database.
- StressTesting.py
  - This script give us methods for make connections of user to databases.

Libraries

- Class SecurityTesting.py
  - Methods
    - ValidateLogin
    - ValidateUserState
    - ValidateCreateOfObjects
    - ValidateReadObjects
    - ValidateUpdateObjects
    - ValidateDeleteObjects
- Class FunctionalTesting.py
  - Methods
    - AllowNulls
    - CharacterLenght
    - RangeOfValues
    - ContainsSpecialCharacters
    - ContainsNumbers
    - ContainsLetters
- Class DatabaseConnect.py
  - Methods
    - ConnectDatabase
    - CloseConnection
- Class StressTesting.py
  - Methods
    - SimulateConnection
    - CloseConnection
- Class LoadTesting.py
  - Methods
    - SendLoadToTable
    - GetLoadFromTable
    - DeleteLoadFromTable
    - UpdateLoadFromTable

Algorithm

Keyword	Script A	Script B	Percentage
Function	5 times	5 times	100%
Change	3 times	2 times	66.66%
Property	0 times	0 times	0 %
(“vmoreno”, “Pass123”, “CloudServer”)	1 times	8 times	12.5 %

Similarity = (100 + 66.66 + 0 + 12.5) / 3
Similarity = 179.16 / 3
Similarity = 59.72 %

Keyword “Property” is excluded because never script contains the keyword

How to run Inspector Script

Inspector.py C:\Python27\MyPythonScripts\OracleProject\TestingScripts

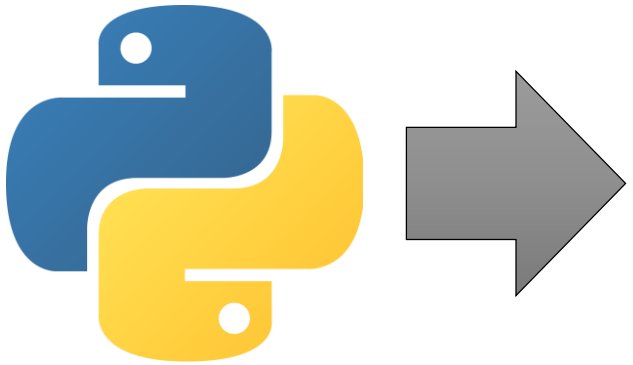
```
1   from DatabaseLibraries.Analyzer import *
2
3   KeywordsList = (
4                   "ValidateLogin",
5                   "ValidateUserState",
6                   "SimulateConnection",
7                   "CloseConnection",
8                   #"SendLoadToTable",
9                   #"GetLoadFromTable",
10                  #"DeleteLoadFromTable",
11                  #"UpdateLoadFromTable",
12                  )
13
14  PathFiles ={
15              #"ValidateActions"   : "MyPythonScripts/OracleProject/TestingScripts/ValidateActions.py",
16              #"ValidateFields"    : "MyPythonScripts/OracleProject/TestingScripts/ValidateFields.py",
17              #"Script ValidateLoads"     : "MyPythonScripts/OracleProject/TestingScripts/ValidateLoads.py",
18              #"Script ValidateLoads2"   : "MyPythonScripts/OracleProject/TestingScripts/ValidateLoads_2.py",
19              "ValidateStress"     :   "MyPythonScripts/OracleProject/TestingScripts/ValidateStress.py",
20              "ValidateStress_2"   : "MyPythonScripts/OracleProject/TestingScripts/ValidateStress_2.py",
21             }
22
```

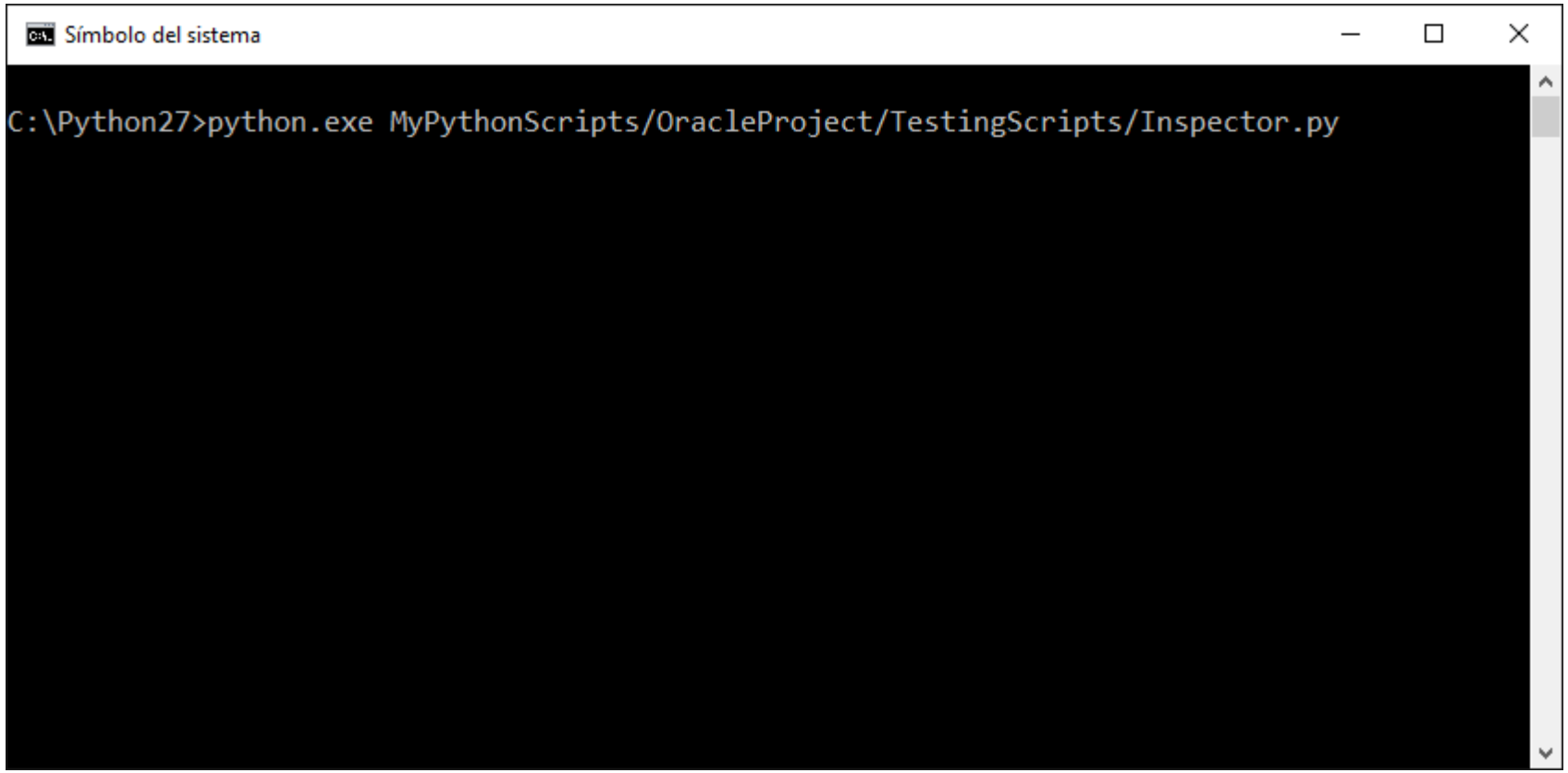
1) Define keywords

Open Inspector.py file and edit the script according to need

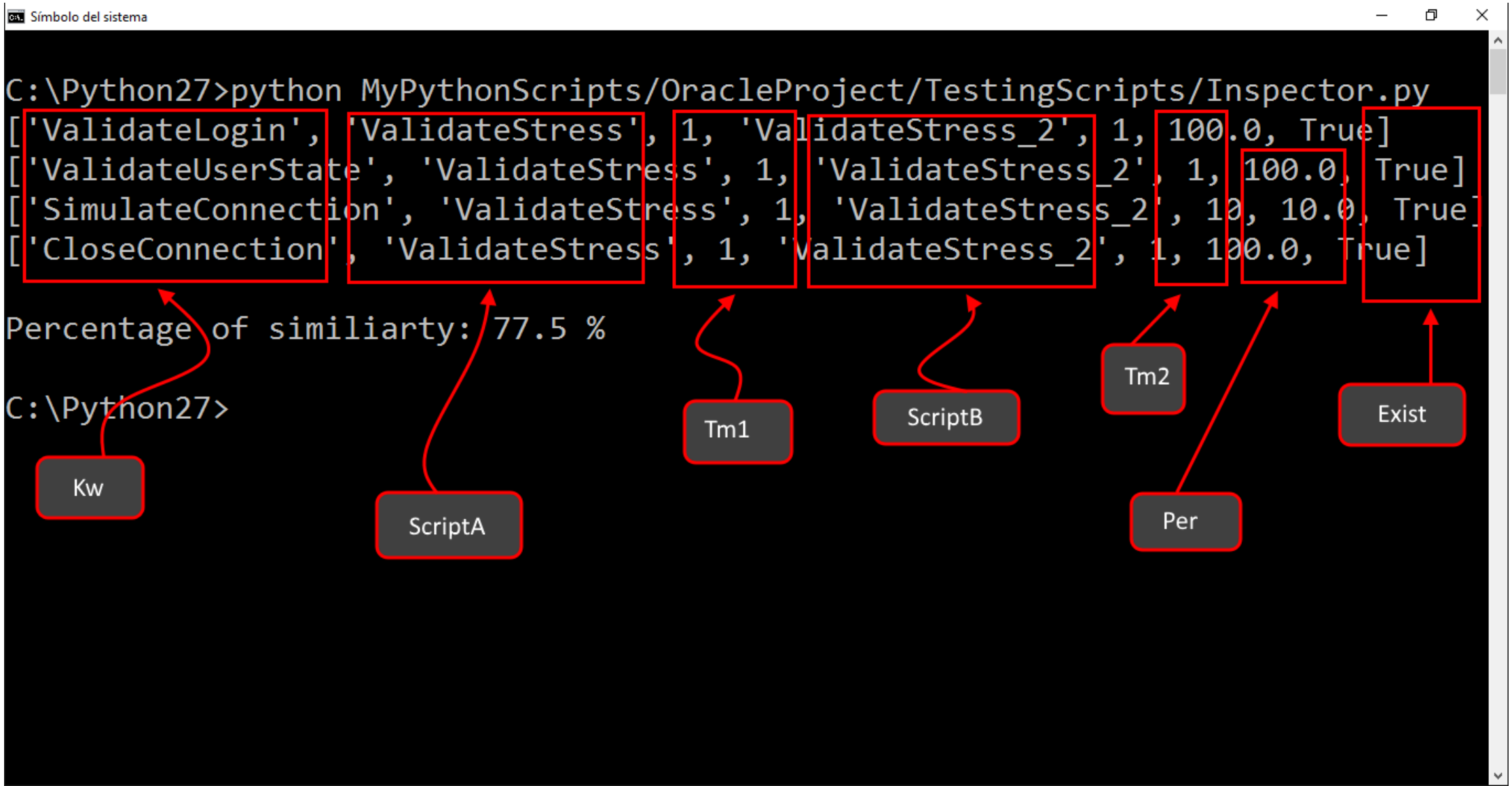
2) Define path files (only two)

How to run Inspector Script





Showing results



Showing results

At the screen we see several results, the way to interpret its as follows:

[Kw, ScriptA, Tm1, ScriptB, Tm2, Per, Exist]

- **Kw:** It's the keyword to search.
- **ScriptA:** It's the first script to compare.
- **Tm1:** Total matches found at Script A.
- **ScriptB:** It's the second script to compare.
- **Tm2:** Total matches found at Script B.
- **Per:** Represents the percentage of similarity based in the keyword.
- **Exist:** Indicates if exist keywords between the two scripts.

Percentage of Similarity

This calculation is based as follows:

Similarity %	=	$\frac{\text{Sum of Percentage of similarity for each keyword}}{\text{Number of keywords coinciding between scripts}}$

Notes

This was my first Project at Python. I have never worked with this language. I think its a easy programming language with potential and versatility.