IT314 - Software Engineering
Lab 5 - Static Analysis

Name: Darshan Dobariya
Student ID: 202001183

Static Analysis:
Static analysis is a method of examining the source code of a software program without
executing it. Static analysis can help detect errors, bugs, vulnerabilities, and other quality issues
in the code. Static analysis tools can perform various tasks such as checking syntax, style,
logic, data flow, control flow, and security. Static analysis can improve the reliability,
performance, and maintainability of software by identifying and correcting defects early in the
development process.

Static Analysis Tools:
Static analysis tools are software tools that analyze the source code of a program without
executing it. They can help developers find and fix errors, bugs, vulnerabilities, code smells, and
other quality issues in their code. Static analysis tools can also measure various metrics of the
code, such as complexity, readability, maintainability, test coverage, and documentation. Static
analysis tools can be integrated into the development process as part of the code editor, the
version control system, or the continuous integration pipeline. Some examples of static analysis

tools are SonarQube, PMD, ESLint, and Pylint.

List of tools:

Python:

- Mypy
- Pylint
- Pyflakes
- Pycodestyle (pep8)
- Flake8
- Prospector
- Bandit

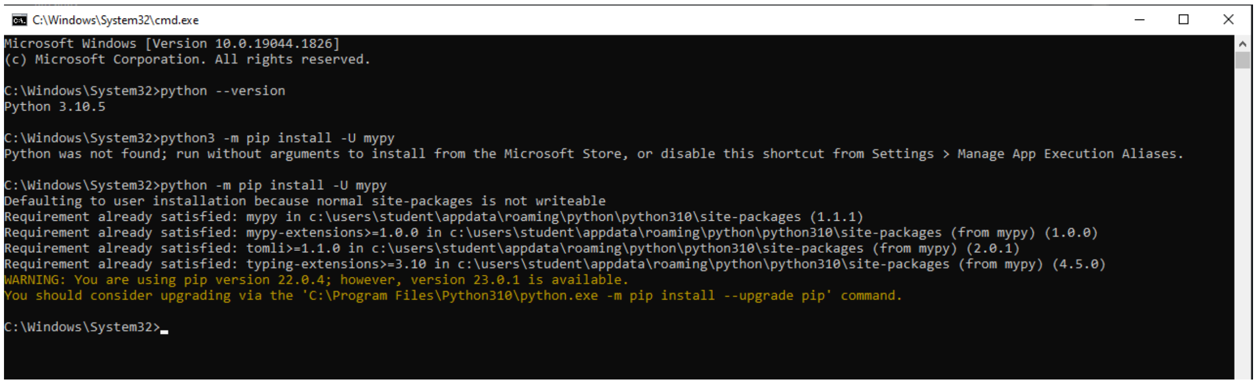
Java:

- FindBugs
- PMD
- Checkstyle
- Error Prone
- Spoon
- Spotbugs

Select the tool of your choice. Select a git repository, use the selected tool and analyze the files
from the selected repository. Submit the tool output and understanding of the errors.

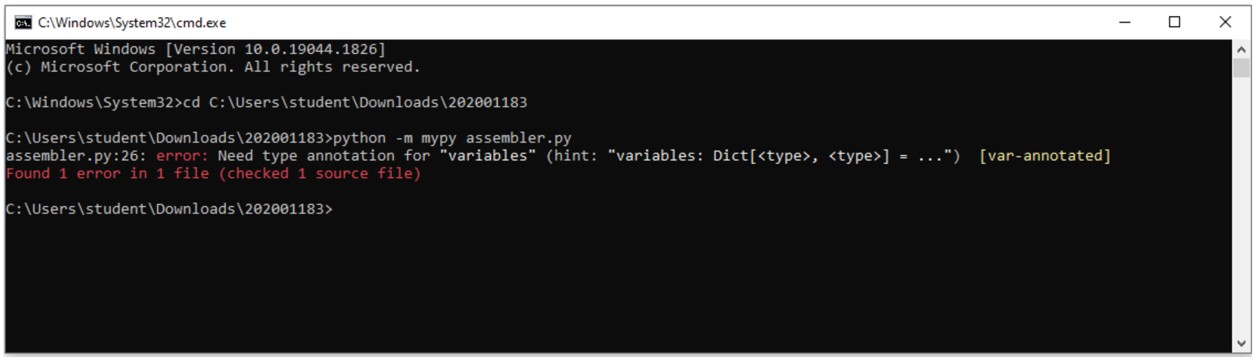
Here given tools i use mypy.

A)Frist i install mypy.



```
C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.19044.1826]
(c) Microsoft Corporation. All rights reserved.

C:\Windows\System32>python --version
Python 3.10.5

C:\Windows\System32>python3 -m pip install -U mypy
Python was not found; run without arguments to install from the Microsoft Store, or disable this shortcut from Settings > Manage App Execution Aliases.

C:\Windows\System32>python -m pip install -U mypy
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: mypy in c:\users\student\appdata\roaming\python\python310\site-packages (1.1.1)
Requirement already satisfied: mypy-extensions>=1.0.0 in c:\users\student\appdata\roaming\python\python310\site-packages (from mypy) (1.0.0)
Requirement already satisfied: tomli>=1.1.0 in c:\users\student\appdata\roaming\python\python310\site-packages (from mypy) (2.0.1)
Requirement already satisfied: typing-extensions>=3.10 in c:\users\student\appdata\roaming\python\python310\site-packages (from mypy) (4.5.0)
WARNING: You are using pip version 22.0.4; however, version 23.0.1 is available.
You should consider upgrading via the 'C:\Program Files\Python310\python.exe -m pip install --upgrade pip' command.

C:\Windows\System32>_
```

B)First file:



```
C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.19044.1826]
(c) Microsoft Corporation. All rights reserved.

C:\Windows\System32>cd C:\Users\student\Downloads\202001183

C:\Users\student\Downloads\202001183>python -m mypy assembler.py
assembler.py:26: error: Need type annotation for "variables" (hint: "variables: Dict[<type>, <type>] = ...")  [var-annotated]
Found 1 error in 1 file (checked 1 source file)

C:\Users\student\Downloads\202001183>
```

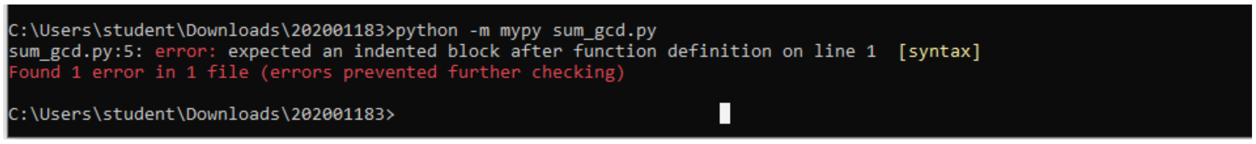
Error:

In file assembler.py there is one error of “Need type annotation for variables “

Solution:

Use comments to annotate variable type

C) Second File:

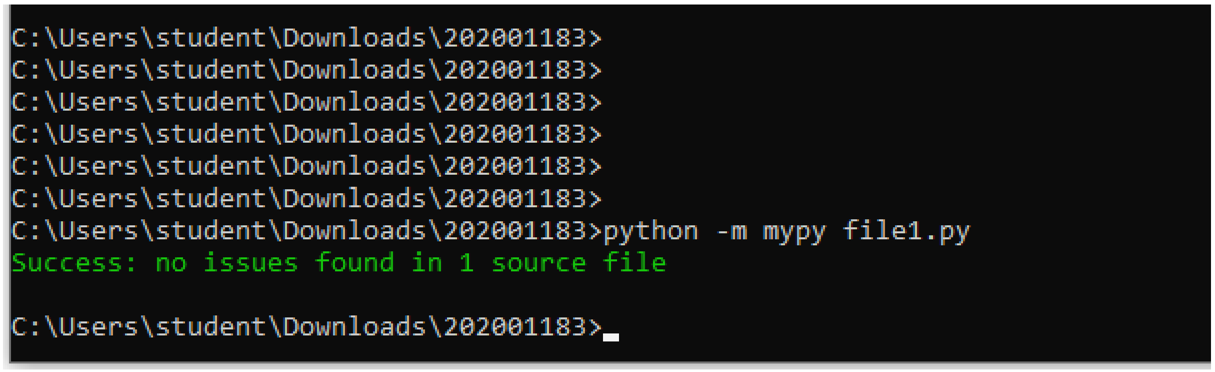


```
C:\Users\student\Downloads\202001183>python -m mypy sum_gcd.py
sum_gcd.py:5: error: expected an indented block after function definition on line 1  [syntax]
Found 1 error in 1 file (errors prevented further checking)

C:\Users\student\Downloads\202001183>
```

Error:
expected an indented block after function definition on line 1
This error is raised when you forget to add an indent in your code

Solution:
To solve this error make sure your code contains the proper number of indents.

D) Third File:



```
C:\Users\student\Downloads\202001183>
C:\Users\student\Downloads\202001183>
C:\Users\student\Downloads\202001183>
C:\Users\student\Downloads\202001183>
C:\Users\student\Downloads\202001183>
C:\Users\student\Downloads\202001183>
C:\Users\student\Downloads\202001183>python -m mypy file1.py
Success: no issues found in 1 source file

C:\Users\student\Downloads\202001183>
```

No error in this file: