Adding new Detectors to Slither

SLITHER

Advanced Topics in Software Engineering(CSE 6324)Fall 2022
Team 4
Iteration 3 – 08/11/2023

- Atul Upadhye
- Kundana Vaka
- Yamini Dhulipalla
- Srikar Sai Yarlagadda
- Vigneshwar Selvaraj

GitHub link: https://github.com/AtulUpadhye17/CSE-6324-Team4

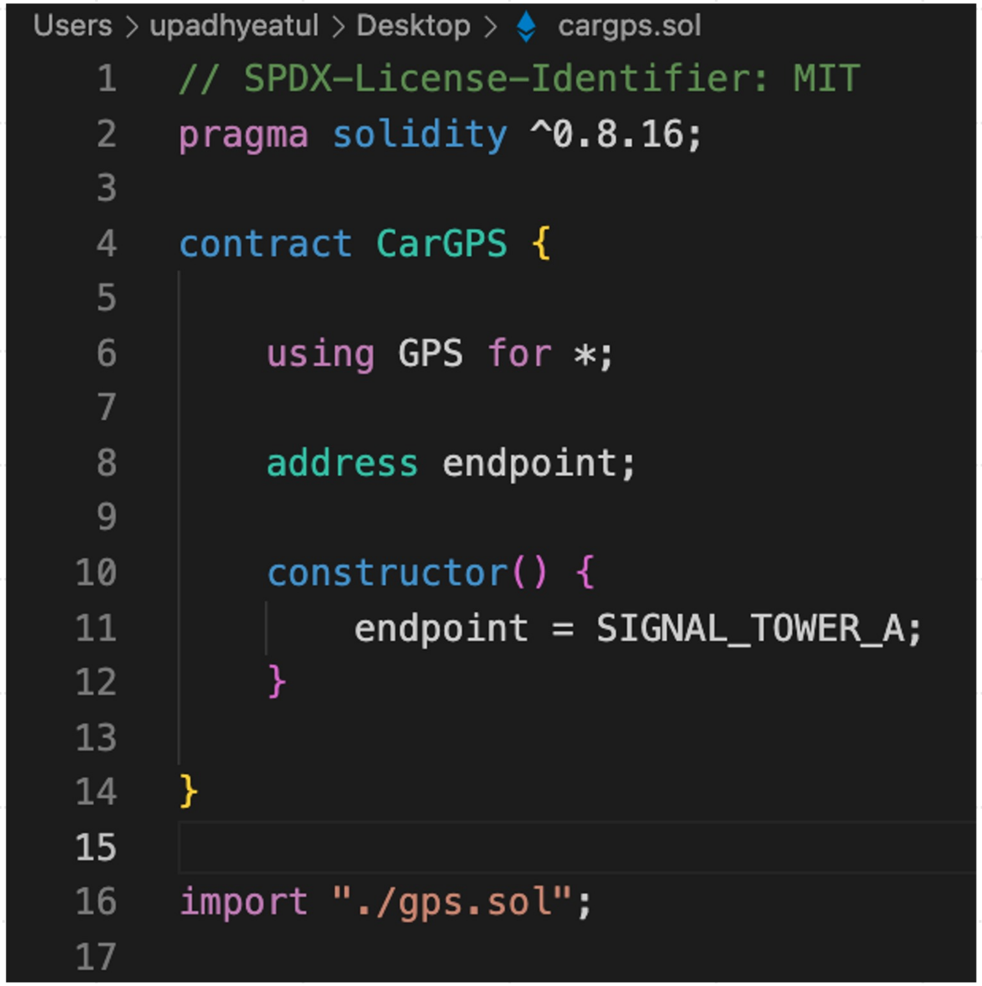
Plan:

Iteration 3	Iteration 4
• Compare Slither[2] with other static analysis tools: Solhint[3] and Ethlint[4]. • Identified additional detectors missing or should be part of Slither : 1. Imports on Top 2. Pragma statement on top. • Write code and test detector for each of the identified detectors. • Integrate the implemented detectors in the Slither, and built and test the tool.	• Ordering Detector : Combining all the ordering related detectors into one.(Function, Contract, Library etc.). • Test the implemented detectors and check for any bugs and fix them. • Create the documentation of the project.

Imports on Top

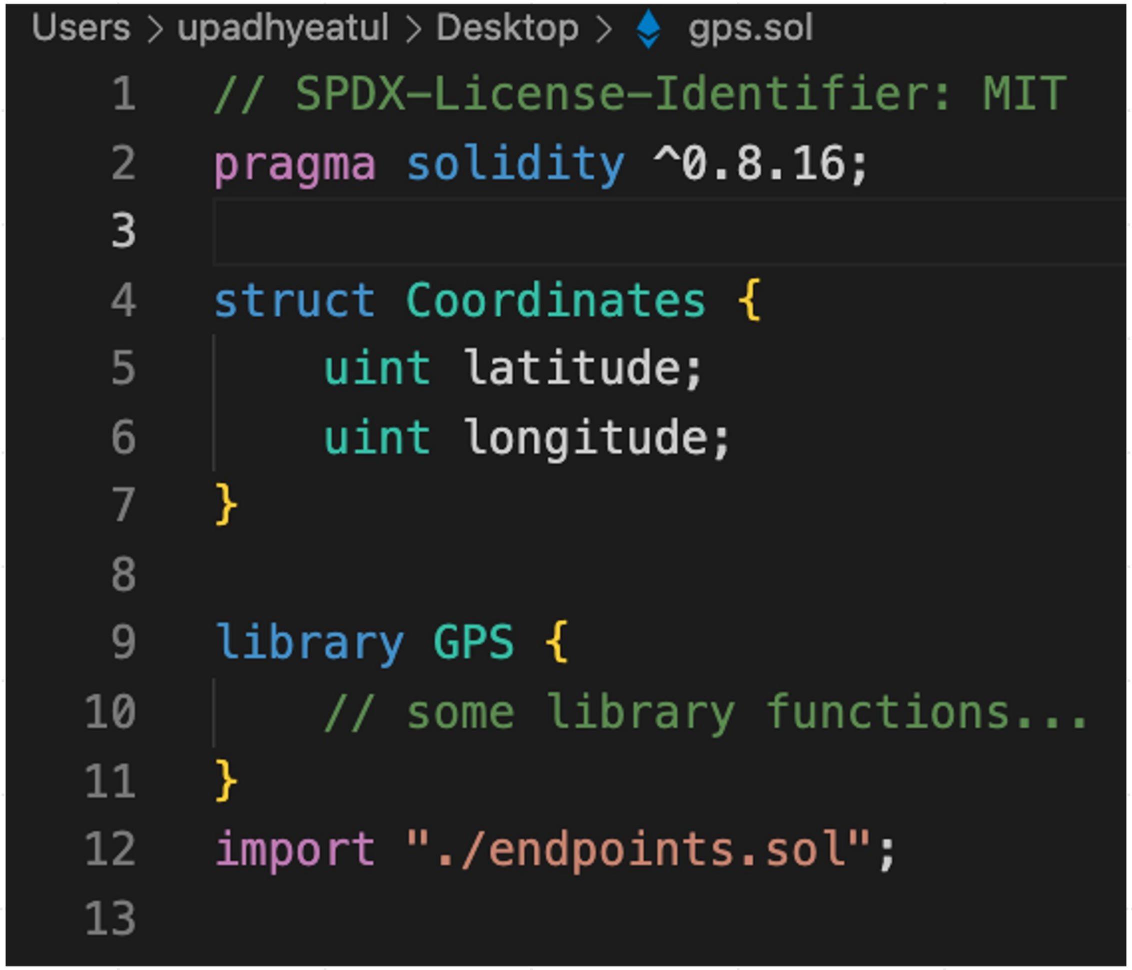
- Description:
  - In general, the Solidity import statement imports code from one module into another. To import all the code from a module, we specify the import keyword followed by the module name
  - Import statements appear at the top of the solidity file, beneath any comments and docstrings that may exist, and before module globals and constants. This is because importing modules or packages at the top of a file makes the structure of the code clearer.[3][6]
  - This detector also checks through nested input solidity files to find whether the import statement follows the style guide convention.

Contract Samples



Users > upadhyeatul > Desktop > cargps.sol

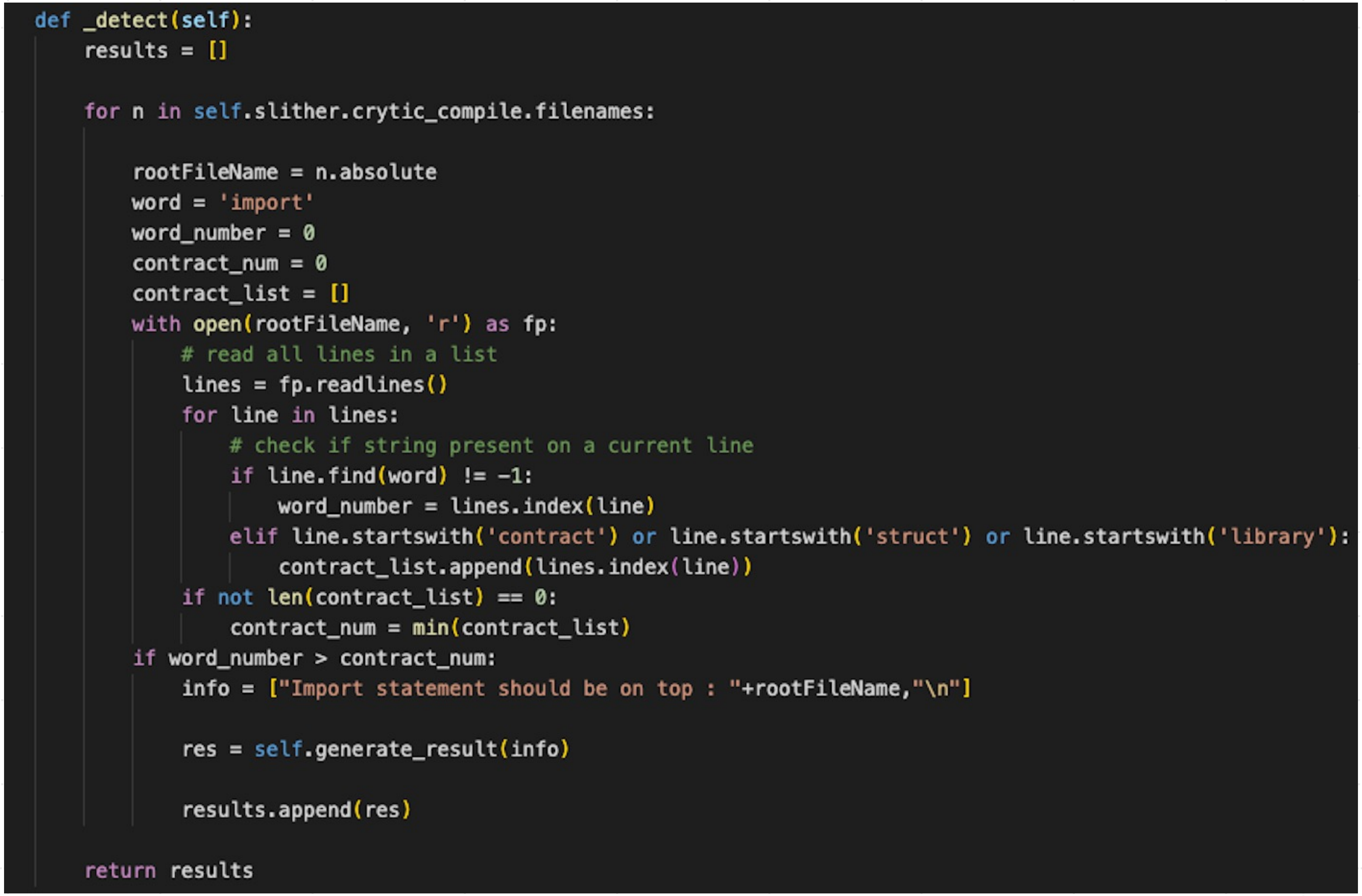
```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.16;

contract CarGPS {

    using GPS for *;

    address endpoint;

    constructor() {
        endpoint = SIGNAL_TOWER_A;
    }

}

import "./gps.sol";

```



Users > upadhyeatul > Desktop > gps.sol

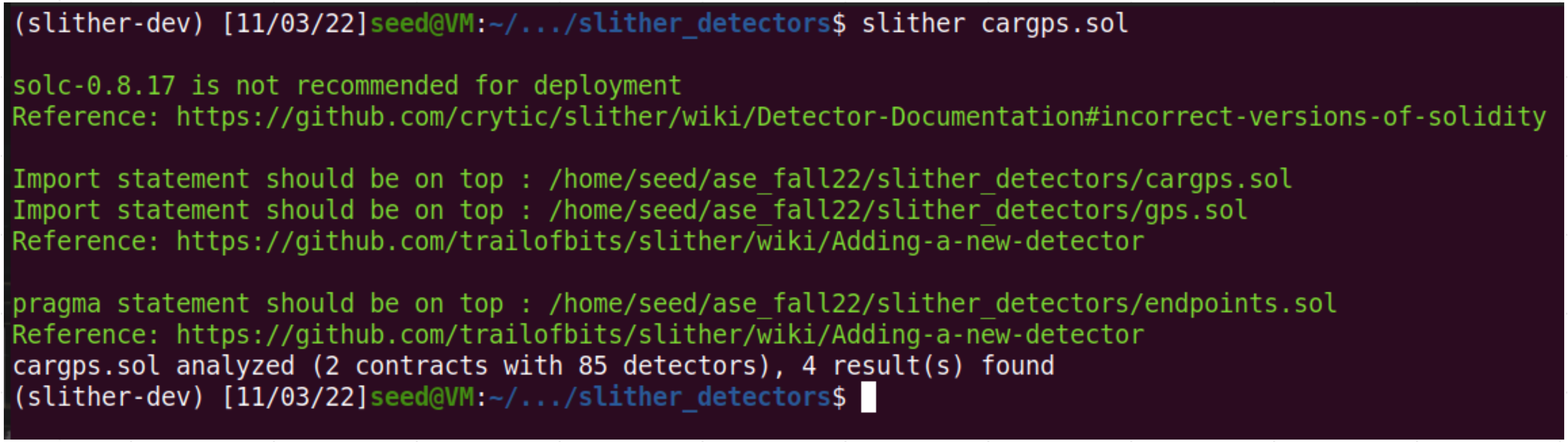
```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.16;

struct Coordinates {
    uint latitude;
    uint longitude;
}

library GPS {
    // some library functions...
}
import "./endpoints.sol";

```

Detector Code



```python
def _detect(self):
    results = []

    for n in self.slither.crytic_compile.filenames:

        rootFileName = n.absolute
        word = 'import'
        word_number = 0
        contract_num = 0
        contract_list = []
        with open(rootFileName, 'r') as fp:
            # read all lines in a list
            lines = fp.readlines()
            for line in lines:
                # check if string present on a current line
                if line.find(word) != -1:
                    word_number = lines.index(line)
                elif line.startswith('contract') or line.startswith('struct') or line.startswith('library'):
                    contract_list.append(lines.index(line))
            if not len(contract_list) == 0:
                contract_num = min(contract_list)
        if word_number > contract_num:
            info = ["Import statement should be on top : "+rootFileName,"\n"]

            res = self.generate_result(info)

            results.append(res)

    return results
```

Output



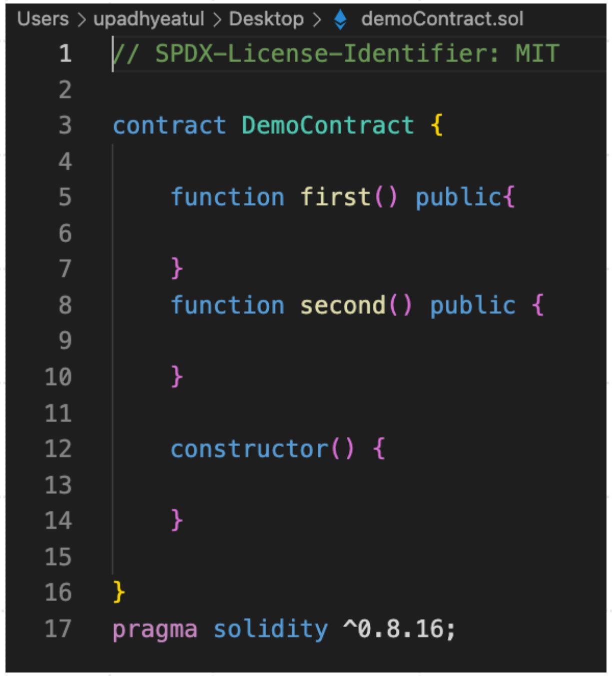
```
(slither-dev) [11/03/22]seed@VM:~/.../slither_detectors$ slither cargps.sol

solc-0.8.17 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity

Import statement should be on top : /home/seed/ase_fall22/slither_detectors/cargps.sol
Import statement should be on top : /home/seed/ase_fall22/slither_detectors/gps.sol
Reference: https://github.com/trailofbits/slither/wiki/Adding-a-new-detector

pragma statement should be on top : /home/seed/ase_fall22/slither_detectors/endpoints.sol
Reference: https://github.com/trailofbits/slither/wiki/Adding-a-new-detector
cargps.sol analyzed (2 contracts with 85 detectors), 4 result(s) found
(slither-dev) [11/03/22]seed@VM:~/.../slither_detectors$ 
```

Pragma statement on Top

- **Description:**
  - Pragma statements appear at the top of the solidity file, beneath any comments and docstrings that may exist, and before module globals and constants.[4]
  - This detector also checks through nested input solidity files to find whether the pragma statement follows the style guide convention.[4]

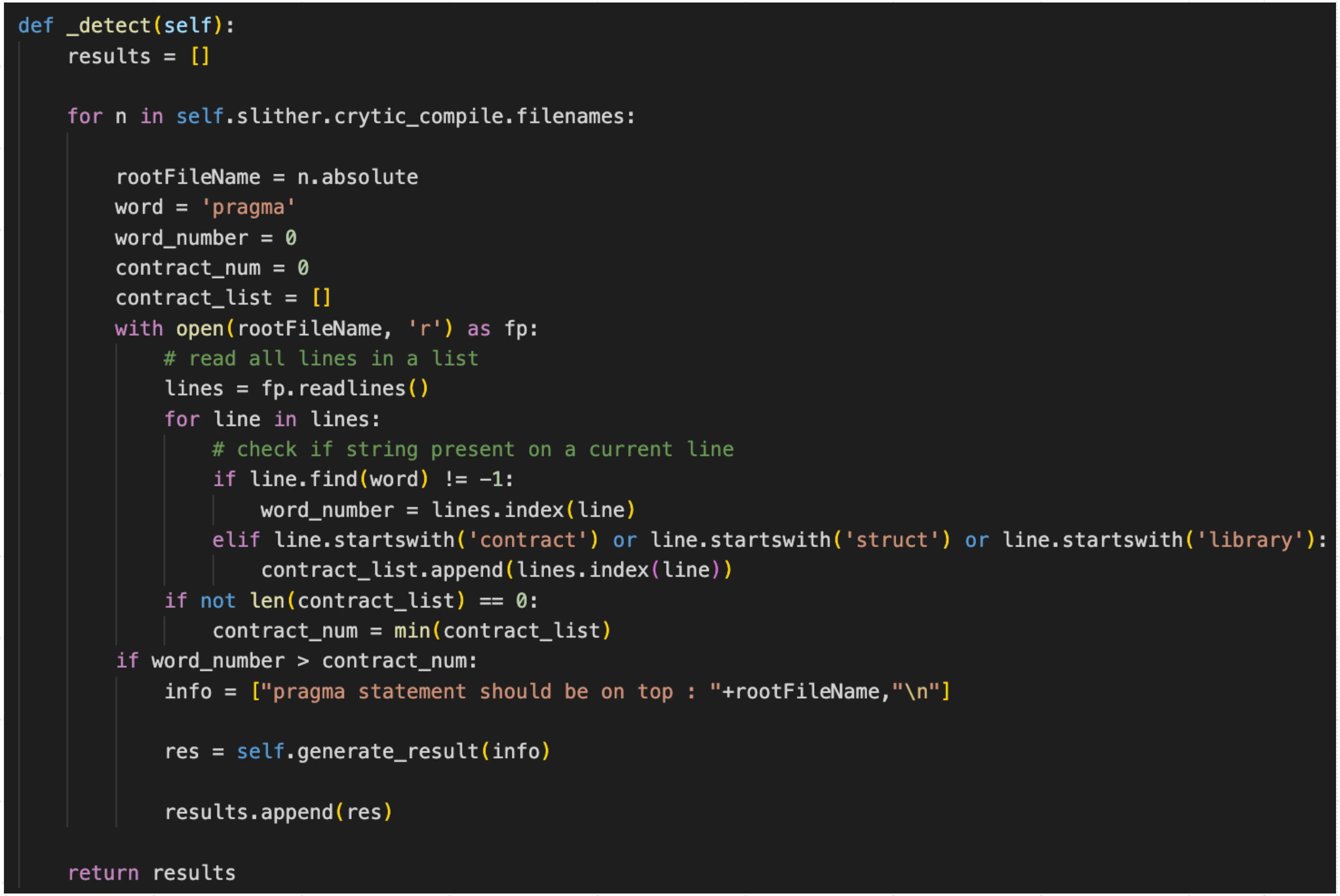
Detector Code

Sample Contract :



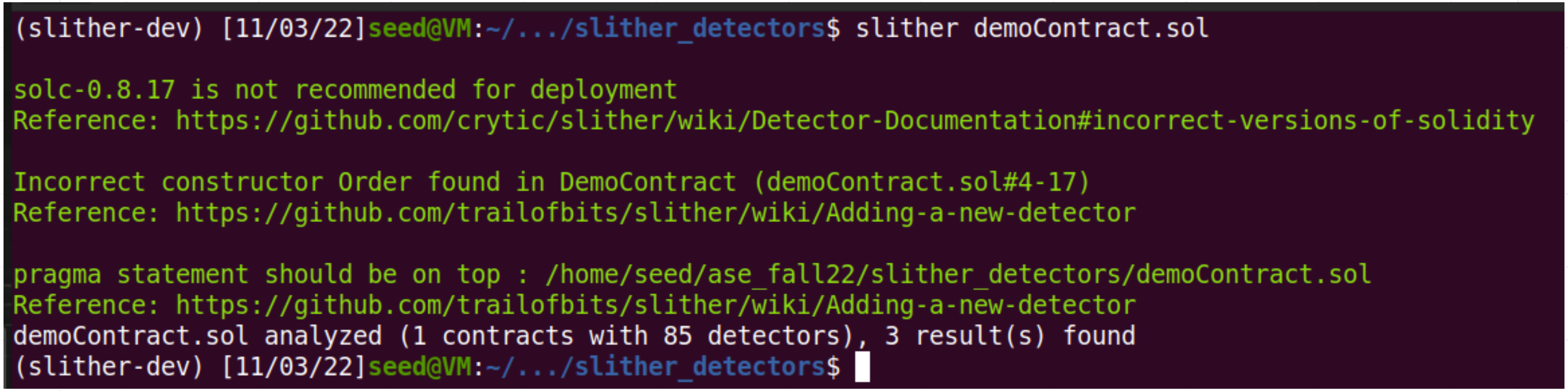
Users > upadhyeatul > Desktop > demoContract.sol

```
// SPDX-License-Identifier: MIT

contract DemoContract {

    function first() public{

    }
    function second() public {

    }

    constructor() {

    }

}
pragma solidity ^0.8.16;
```



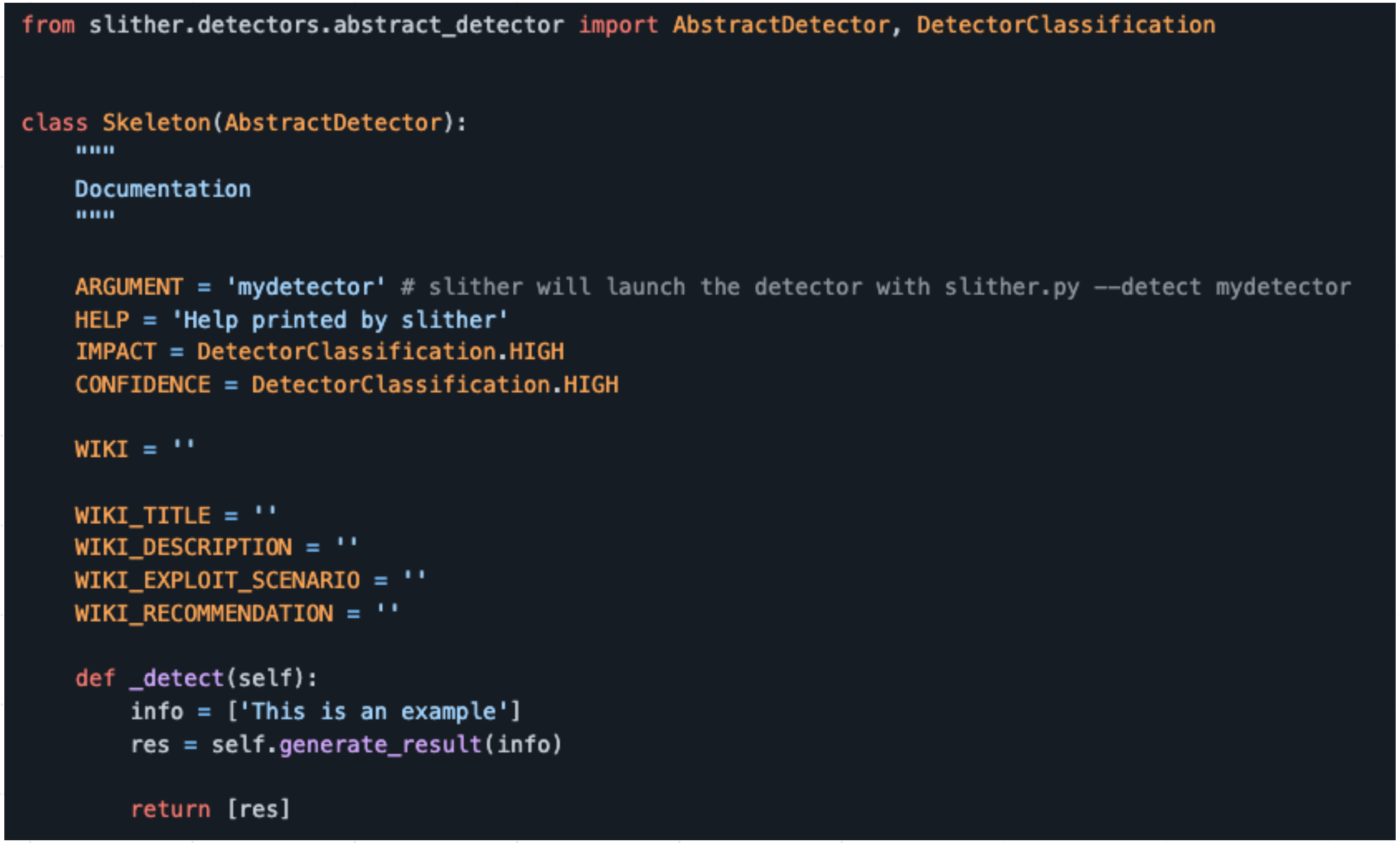
```
def _detect(self):
    results = []

    for n in self.slither.crytic_compile.filenames:

        rootFileName = n.absolute
        word = 'pragma'
        word_number = 0
        contract_num = 0
        contract_list = []
        with open(rootFileName, 'r') as fp:
            # read all lines in a list
            lines = fp.readlines()
            for line in lines:
                # check if string present on a current line
                if line.find(word) != -1:
                    word_number = lines.index(line)
                elif line.startswith('contract') or line.startswith('struct') or line.startswith('library'):
                    contract_list.append(lines.index(line))
            if not len(contract_list) == 0:
                contract_num = min(contract_list)
        if word_number > contract_num:
            info = ["pragma statement should be on top : "+rootFileName,"\n"]

            res = self.generate_result(info)

            results.append(res)

    return results
```

Output



```
(slither-dev) [11/03/22]seed@VM:~/.../slither_detectors$ slither demoContract.sol

solc-0.8.17 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity

Incorrect constructor Order found in DemoContract (demoContract.sol#4-17)
Reference: https://github.com/trailofbits/slither/wiki/Adding-a-new-detector

pragma statement should be on top : /home/seed/ase_fall22/slither_detectors/demoContract.sol
Reference: https://github.com/trailofbits/slither/wiki/Adding-a-new-detector
demoContract.sol analyzed (1 contracts with 85 detectors), 3 result(s) found
(slither-dev) [11/03/22]seed@VM:~/.../slither_detectors$
```

Skeleton Code



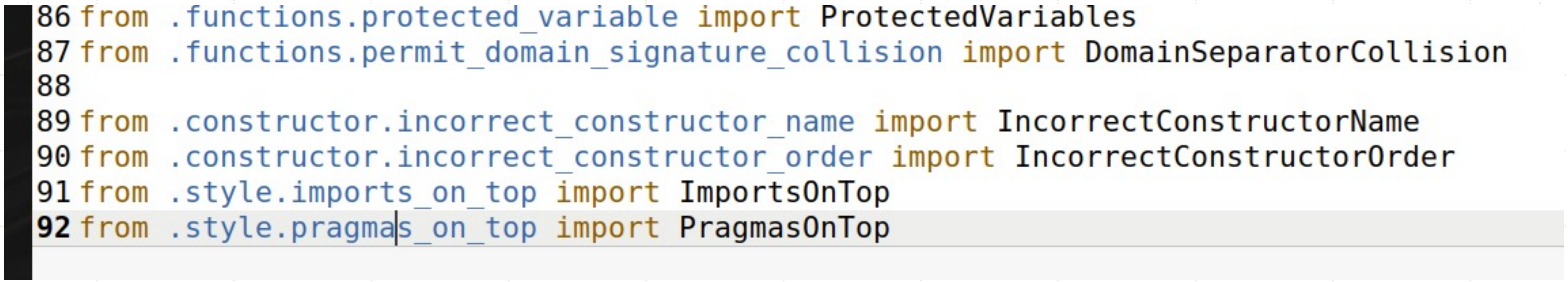
```python
from slither.detectors.abstract_detector import AbstractDetector, DetectorClassification


class Skeleton(AbstractDetector):
    """
    Documentation
    """

    ARGUMENT = 'mydetector' # slither will launch the detector with slither.py --detect mydetector
    HELP = 'Help printed by slither'
    IMPACT = DetectorClassification.HIGH
    CONFIDENCE = DetectorClassification.HIGH

    WIKI = ''

    WIKI_TITLE = ''
    WIKI_DESCRIPTION = ''
    WIKI_EXPLOIT_SCENARIO = ''
    WIKI_RECOMMENDATION = ''

    def _detect(self):
        info = ['This is an example']
        res = self.generate_result(info)

        return [res]
```

Code Parameters

- **ARGUMENT** lets you run the detector from the command line
- **HELP** is the information printed from the command line
- **IMPACT** indicates the impact of the issue. Allowed values are:
  - **DetectorClassification.OPTIMIZATION**: printed in green
  - **DetectorClassification.INFORMATIONAL**: printed in green
  - **DetectorClassification.LOW**: printed in green
  - **DetectorClassification.MEDIUM**: printed in yellow
  - **DetectorClassification.HIGH**: printed in red
- **CONFIDENCE** indicates your confidence in the analysis. Allowed values are:
  - **DetectorClassification.LOW**
  - **DetectorClassification.MEDIUM**
  - **DetectorClassification.HIGH**
- **WIKI** constants are used to generate automatically the documentation.[5]

Integration

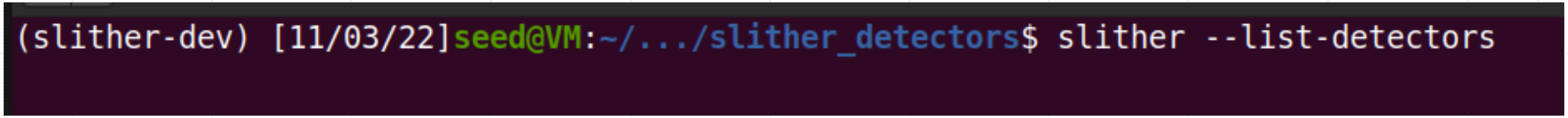
- You can integrate your detector into Slither by:
  - Adding it in slither/detectors/all_detectors.py.(We followed this approach)
  - or, by creating a plugin package.[5]



```python
from .functions.protected_variable import ProtectedVariables
from .functions.permit_domain_signature_collision import DomainSeparatorCollision

from .constructor.incorrect_constructor_name import IncorrectConstructorName
from .constructor.incorrect_constructor_order import IncorrectConstructorOrder
from .style.imports_on_top import ImportsOnTop
from .style.pragmas_on_top import PragmasOnTop
```

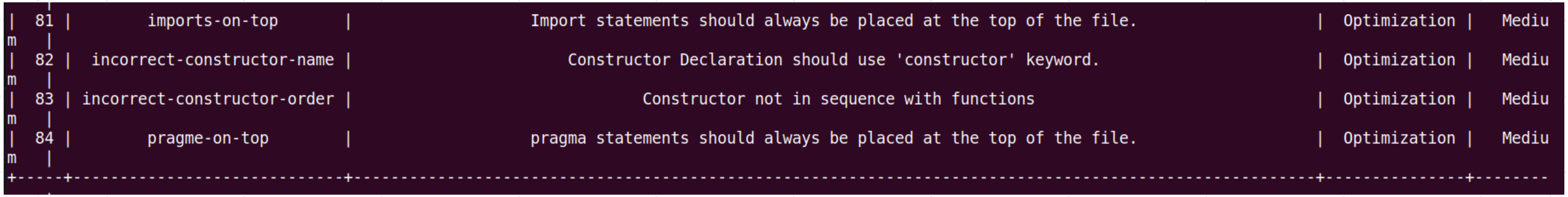
Integration(Continued)

- To check if detectors are properly integrated run below command :



```
(slither-dev) [11/03/22]seed@VM:~/.../slither_detectors$ slither --list-detectors
```

- Integrated Detectors :



```
|  81 |       imports-on-top       |                  Import statements should always be placed at the top of the file.                  | Optimization |   Mediu
m    |
|  82 |  incorrect-constructor-name |                     Constructor Declaration should use 'constructor' keyword.                       | Optimization |   Mediu
m    |
|  83 | incorrect-constructor-order |                            Constructor not in sequence with functions                               | Optimization |   Mediu
m    |
|  84 |        pragme-on-top        |                  pragma statements should always be placed at the top of the file.                  | Optimization |   Mediu
m    |
+-----+----------------------------+-----------------------------------------------------------------------------------------------------+--------------+--------
```

Risks

- The virtual Machine got crashed and we had to setup everything from scratch.
- We have faced issue in setting up the slither developer environment in our local machine. It took us more time in fixing those compiler and runtime errors. We fixed those issues by installing the necessary certificates which isn't mentioned in the developer instructions.
- Understanding the slither code was not apparent enough. We had to run different test cases with many sample sol files to catch on with the code.
- Integrating our detectors with the slither tool was complex since we had to modify our code with slither API features. Thus, making sure the newly added detector flow is in accord with the existing detectors.

Customers and Users

- Any security audit firm, smart contract developer, security expert, or academic researcher can use this tool with our new detectors added to Slither to make the smart contract auditing process more efficient.

Feedback from Shovan:

Shovan suggested us to review the Solidity Style Guidelines and try to implement the detectors for them, as well as work on integrating all the implemented detectors. For this iteration, we have focused on implementing and integrating the detectors.

References

- [1] Overview. (n.d.). Retrieved October 15, 2022, from https://blog.trailofbits.com/2019/05/27/slither-the-leading-static-analyzer-for-smart-contracts/
- [2] GitHub - crytic/slither: Developer installation. (n.d.). GitHub. Retrieved October 15, 2022, from https://github.com/crytic/slither/wiki/Developer-installation
- [3] Rule Index of Solhint. (n.d.). Solhint. Retrieved October 17, 2022, from https://protofire.github.io/solhint/docs/rules.html
- [4] User Guide — Solium 1.0.0 documentation. (n.d.). https://ethlint.readthedocs.io/en/latest/user-guide.html
- [5] Adding a new detector · crytic/slither Wiki. (n.d.-b). GitHub. https://github.com/crytic/slither/wiki/Adding-a-new-detector
- [6] imports-on-top Solhint. (n.d.). Solhint. https://protofire.github.io/solhint/docs/rules/order/imports-on-top.html
- [7] Style Guide — Solidity 0.8.17 documentation. (n.d.). https://docs.soliditylang.org/en/v0.8.17/style-guide.html

Thank You