Workshop

PROGRAM:02

CODEGRAPHS

Jan 16, 2025
Toronto, Canada

Suchakra Sharma
Chief Scientist, Privado Inc.

privado

whoami

I spend a lot of time analyzing computers and code. PhD in Computer Engineering from Polytechnique Montréal. I write poems and click film photos. Lately, AI & art

Talks and trainings at RSA, USENIX Enigma, LISA, NorthSec, SCALE, Blackhat Arsenal, PWL etc.

https://suchakra.wordpress.com/about

Preparations

- Setup Dependencies
  - `Java 21 (Preferably OpenJDK)`
    - `https://www.oracle.com/in/java/technologies/downloads/#java21`
  - `Graphviz`
    - `https://graphviz.org/download`
- Download and Install Joern
  - `wget https://github.com/joernio/joern/releases/latest/download/joern-install.sh`
  - `chmod +x ./joern-install.sh`
  - `sudo ./joern-install.sh`
- Ensure you can access AI code generation tools

Workshop Goals

Answer the following questions about computers,

- What is even *code*?
- How does a machine structure, build, understand and run code?

Primary Goal

- Build a program that analyzes an AI generated program

Stretch Goal

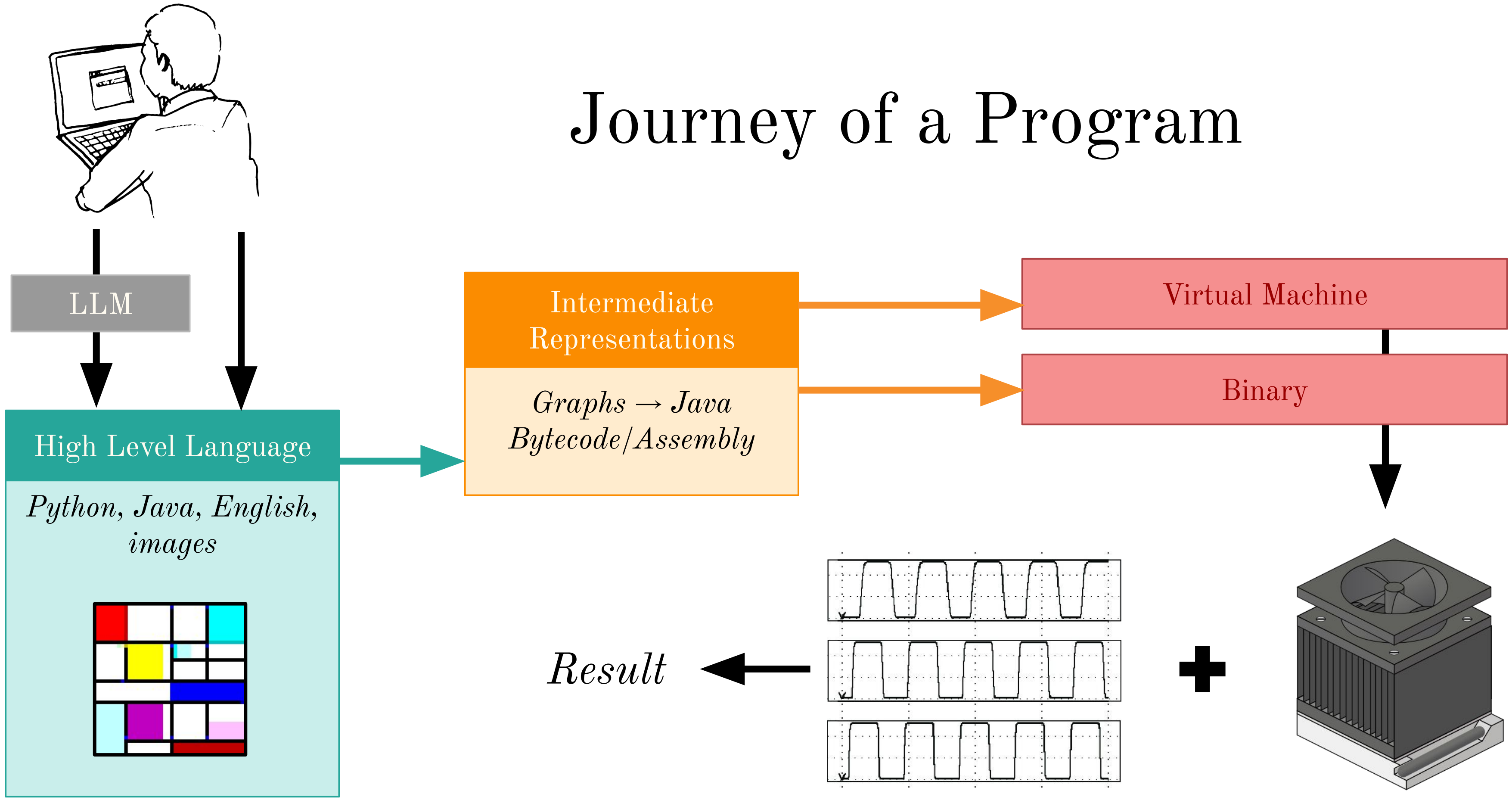
- Build a program that analyzes a program that generates a program

Programming Langauges

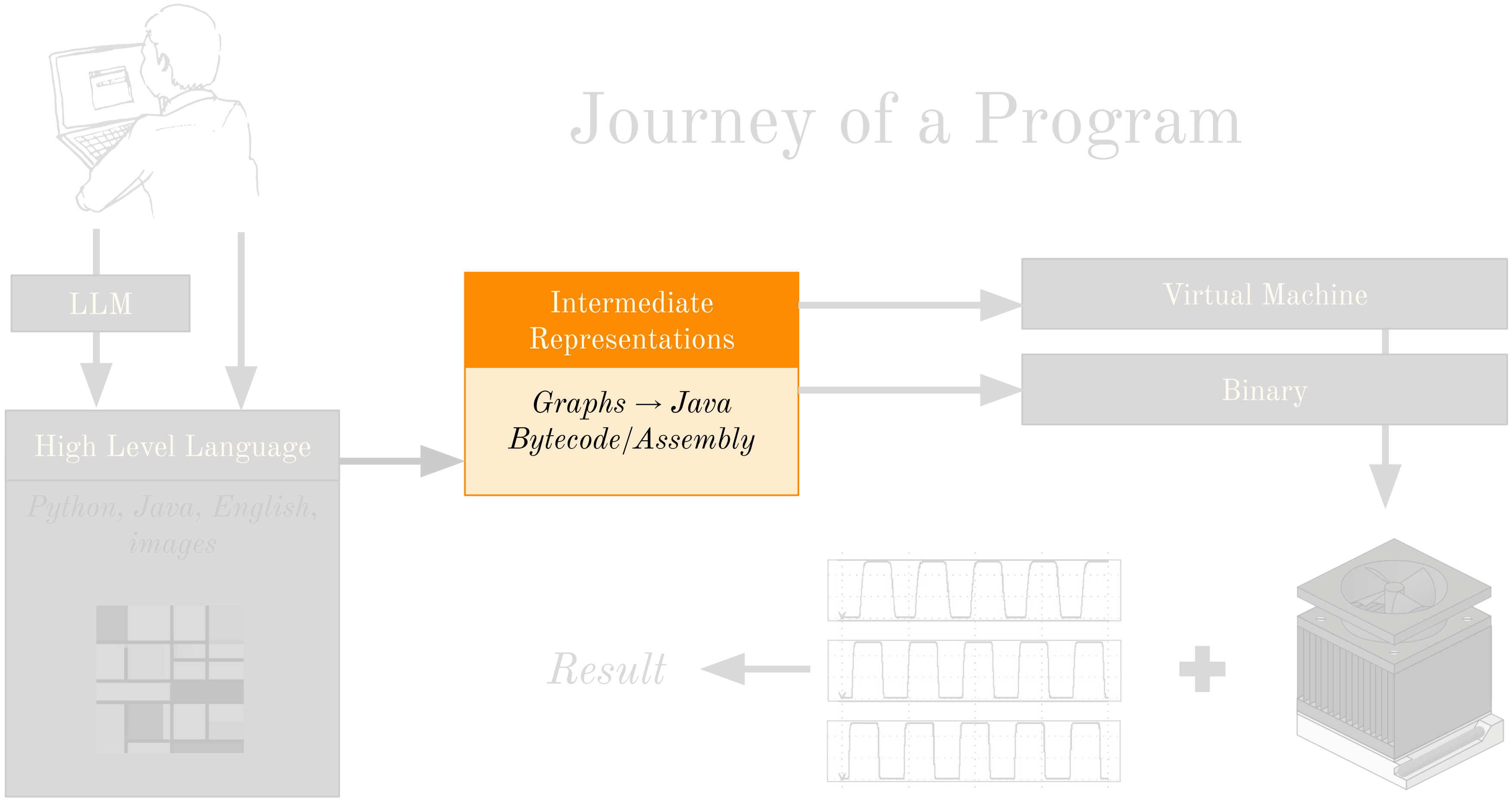
A Gentle Introduction

A bird's eye view of programs

Journey of a Program

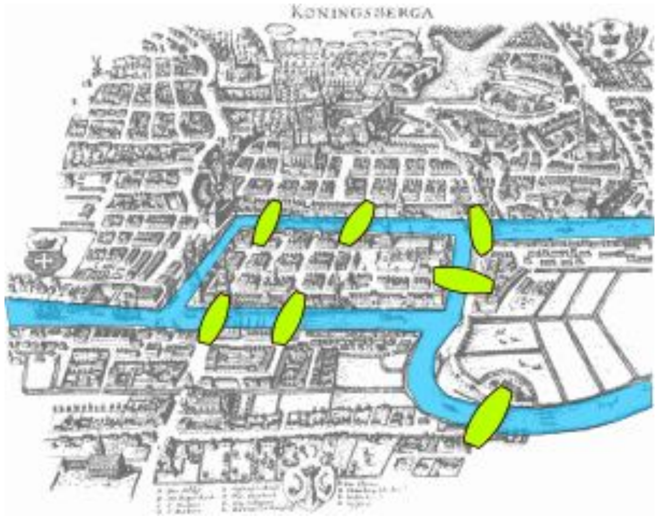

LLM
High Level Language
Python, Java, English, images
Intermediate Representations
Graphs → Java Bytecode/Assembly
Virtual Machine
Binary
Result

Journey of a Program


LLM
High Level Language
Python, Java, English, images
Intermediate Representations
Graphs → Java Bytecode/Assembly
Virtual Machine
Binary
Result

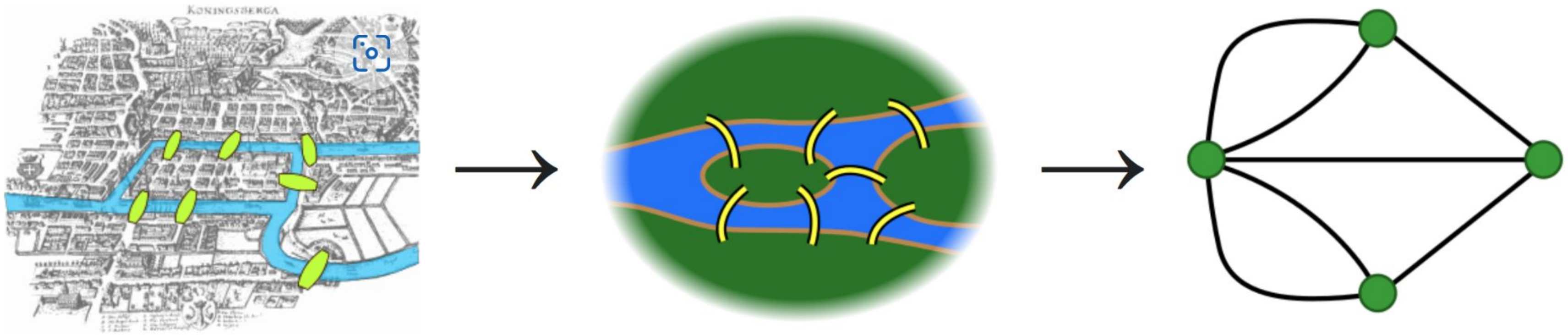
One day in Königsberg in 1736

a dude named Euler had a thought



One day in Königsberg in 1736

a dude named Euler had a thought



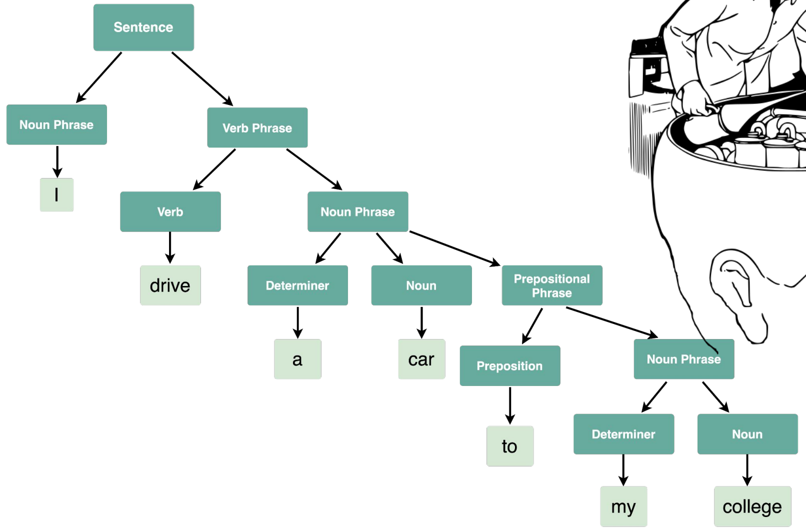
Graphs help us formalize and visualize language

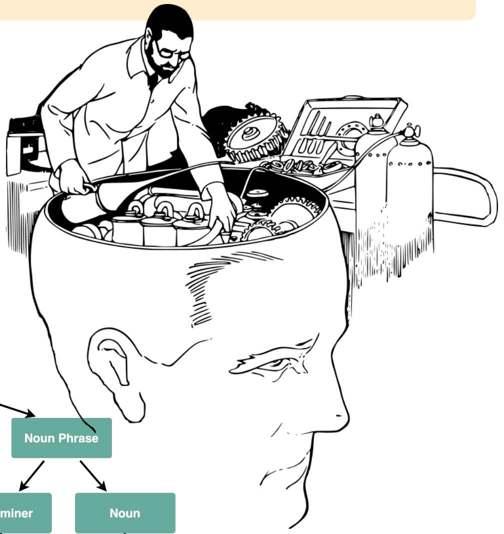
We think in graphs when we speak or code

Programs → Language → Graphs

While programs help solve problems of the world, they themselves are a math problem. Our usually imprecise instructions ***needs determinism to run on a machine***

I drive a car to my college[1]





[1] *geeksforgeeks.com*

What is even *code?*

The language of computers

What is even *code*?

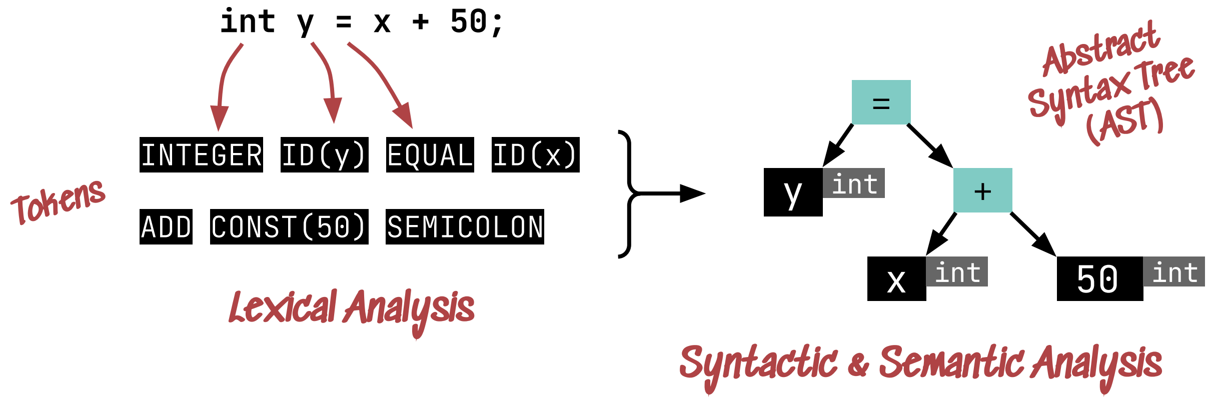
```
int y = x + 50;
```

Tokens

```
INTEGER ID(y) EQUAL ID(x)

ADD CONST(50) SEMICOLON
```

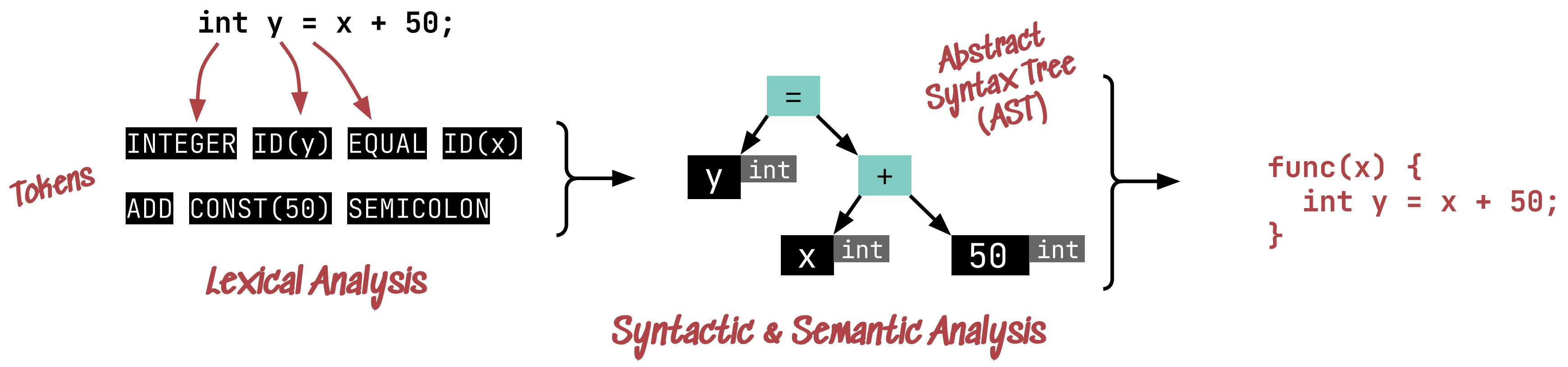
Lexical Analysis

What is even *code?*



```
int y = x + 50;
```

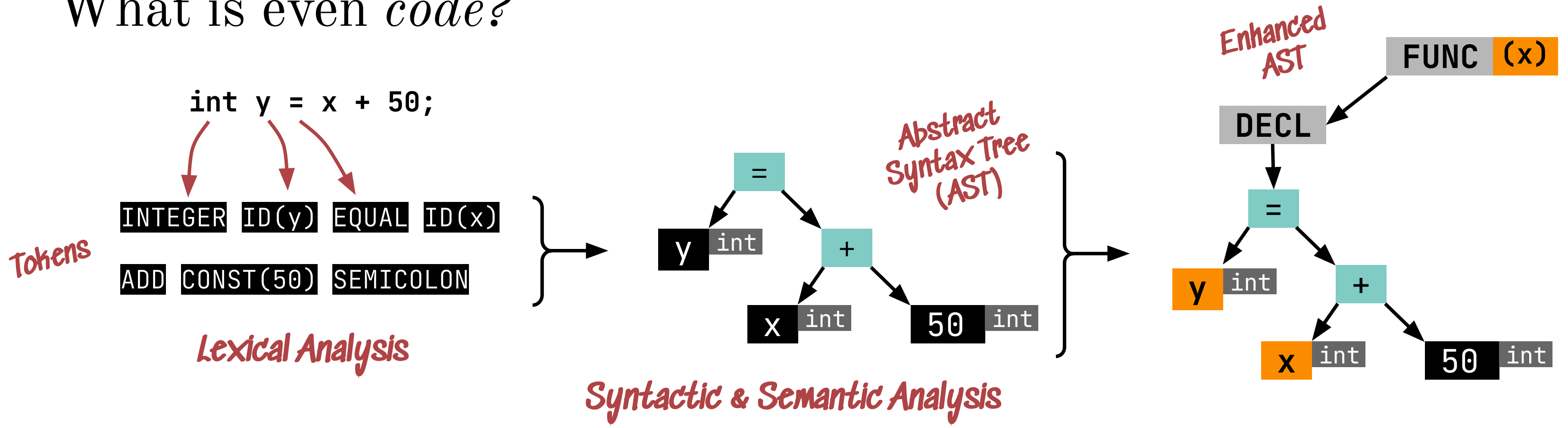
What is even *code?*



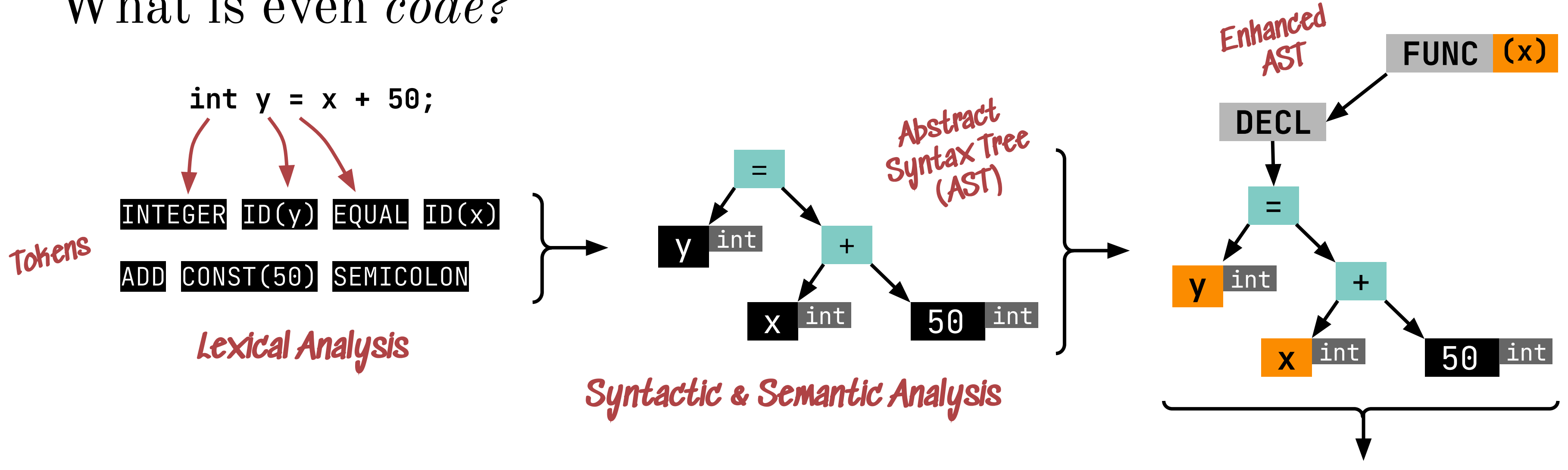
```
int y = x + 50;
```

```
func(x) {
  int y = x + 50;
}
```

What is even *code?*

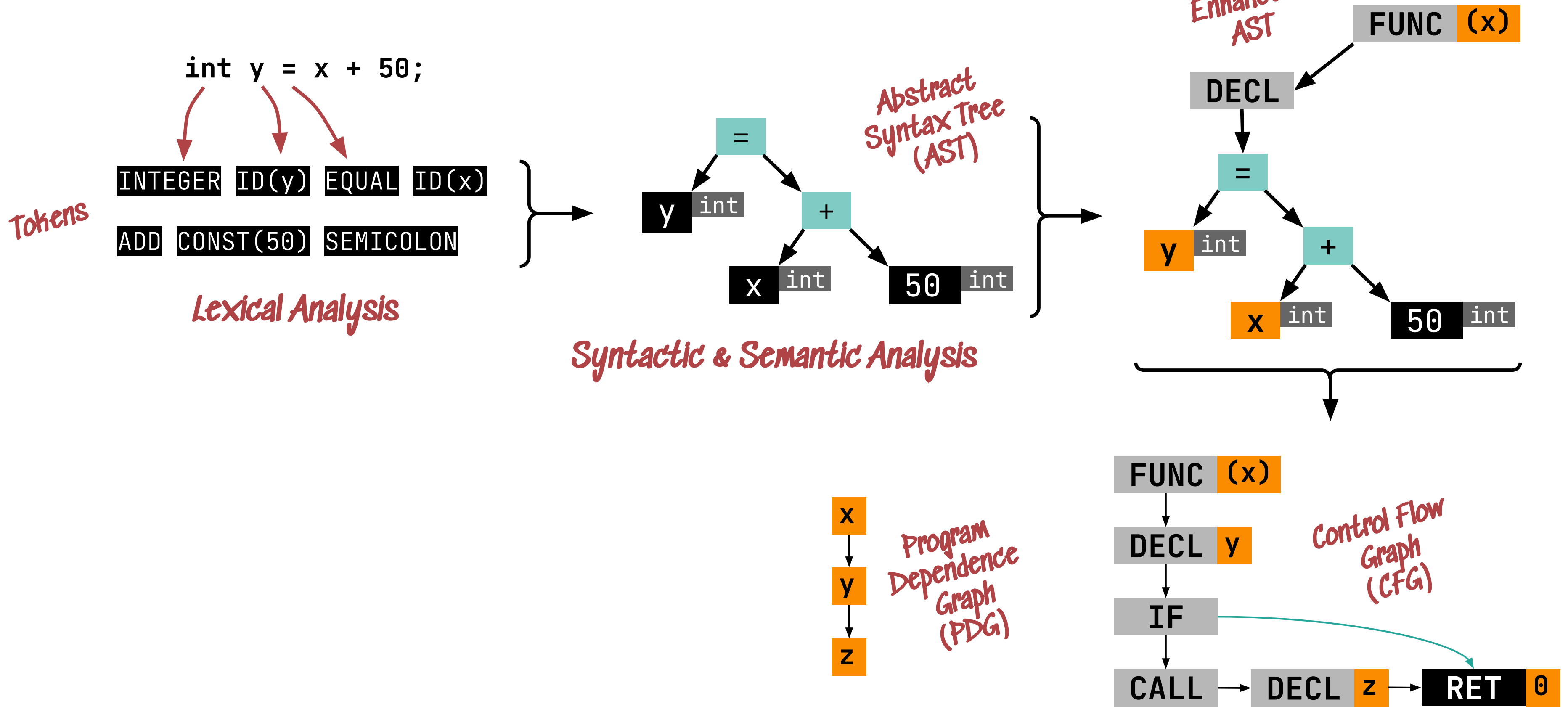


What is even *code?*



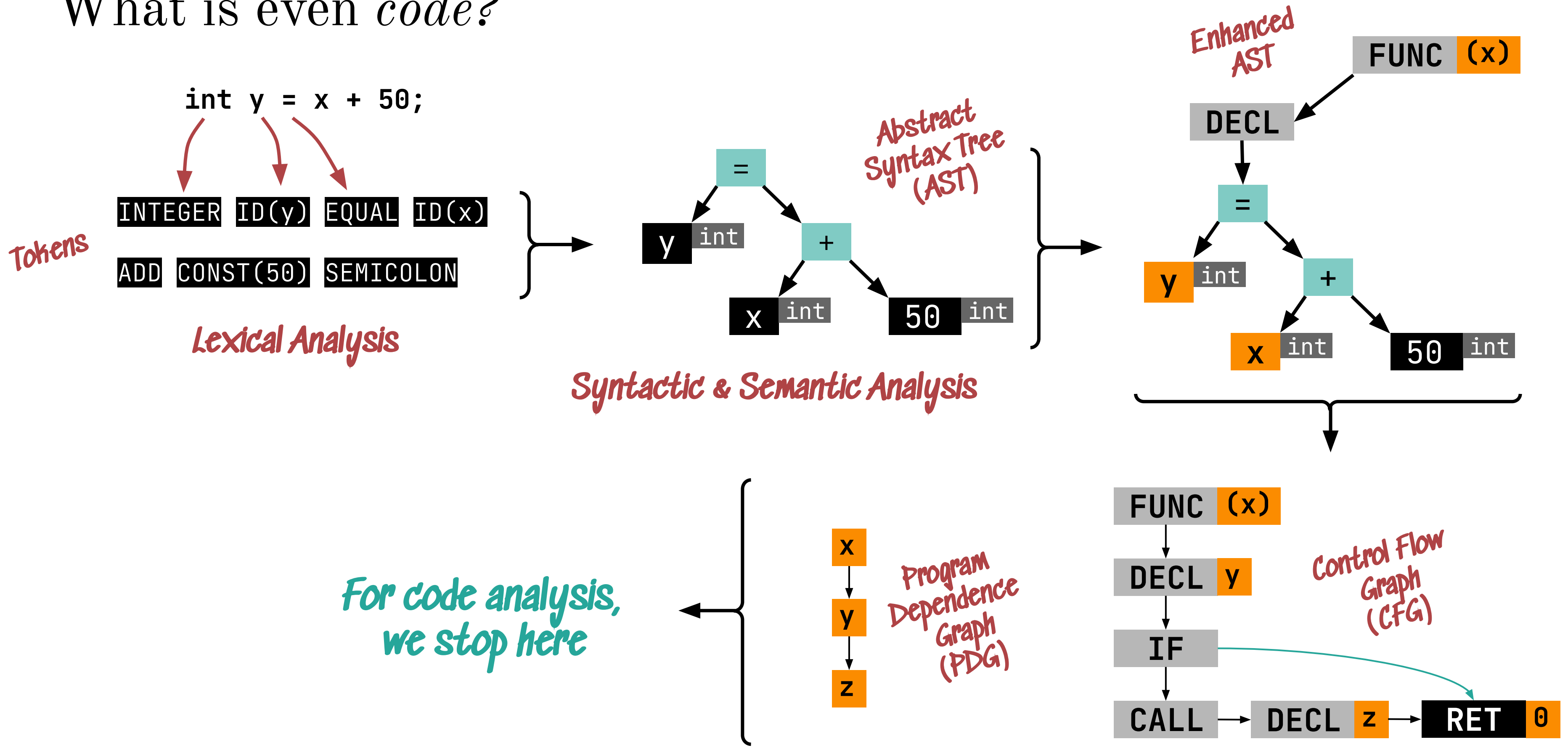
```
func(x) {
  int y = x + 50;
  if (y > 10) {
    wololo()
    z = y
  } else {
    return 0
  }
}
```

What is even *code*?



```
int y = x + 50;
```

What is even *code*?

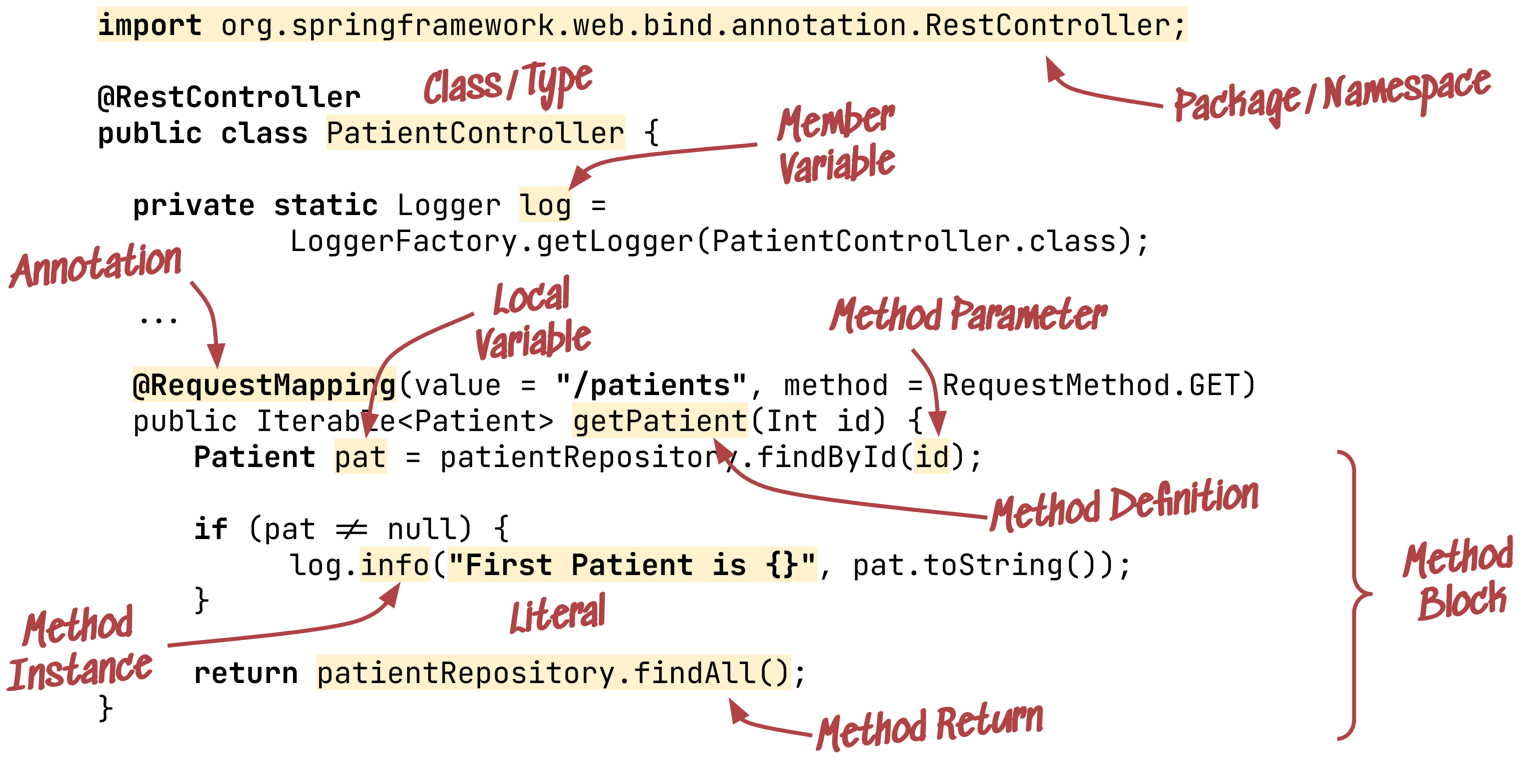


For code analysis, we stop here

Building Blocks of *Programs*

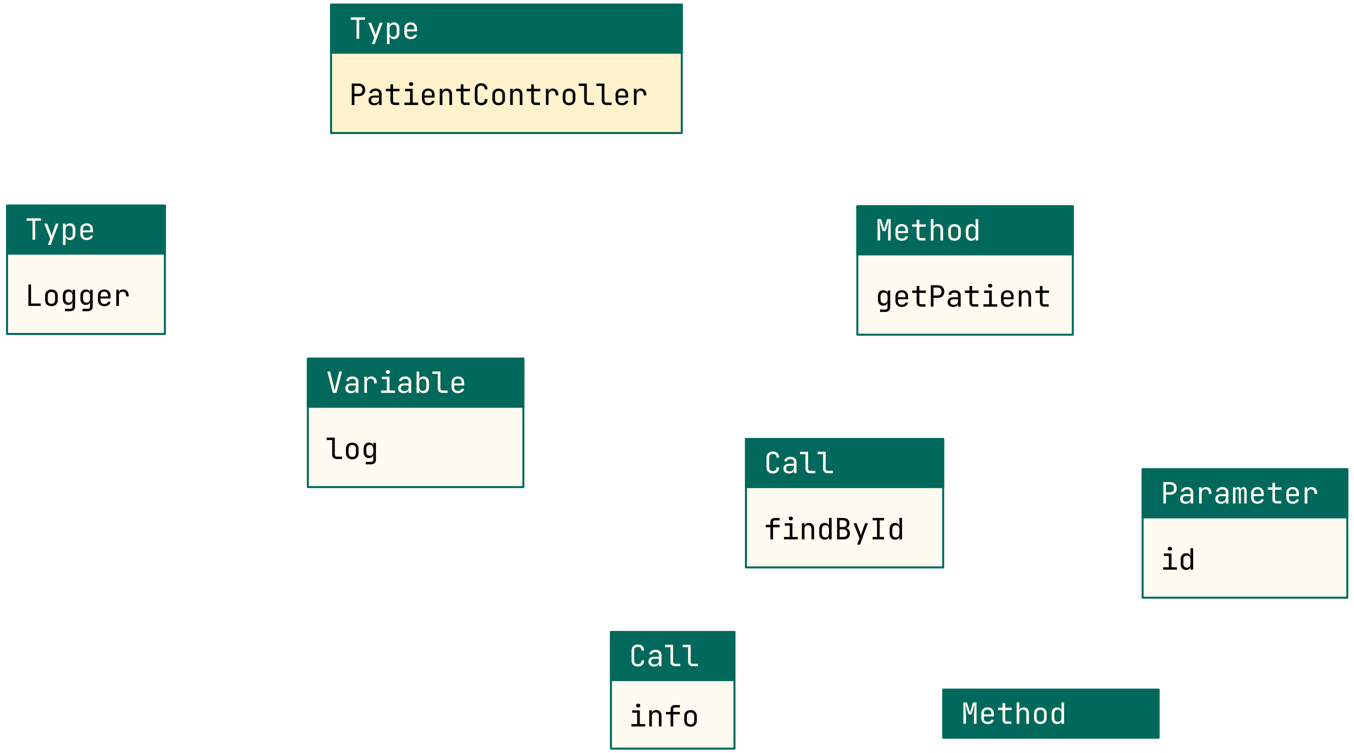
```
import org.springframework.web.bind.annotation.RestController;

@RestController
public class PatientController {

  private static Logger log =
           LoggerFactory.getLogger(PatientController.class);

  ...

  @RequestMapping(value = "/patients", method = RequestMethod.GET)
  public Iterable<Patient> getPatient(Int id) {
     Patient pat = patientRepository.findById(id);

     if (pat ≠ null) {
          log.info("First Patient is {}", pat.toString());
     }

     return patientRepository.findAll();
}
```

Building Blocks of *Programs*

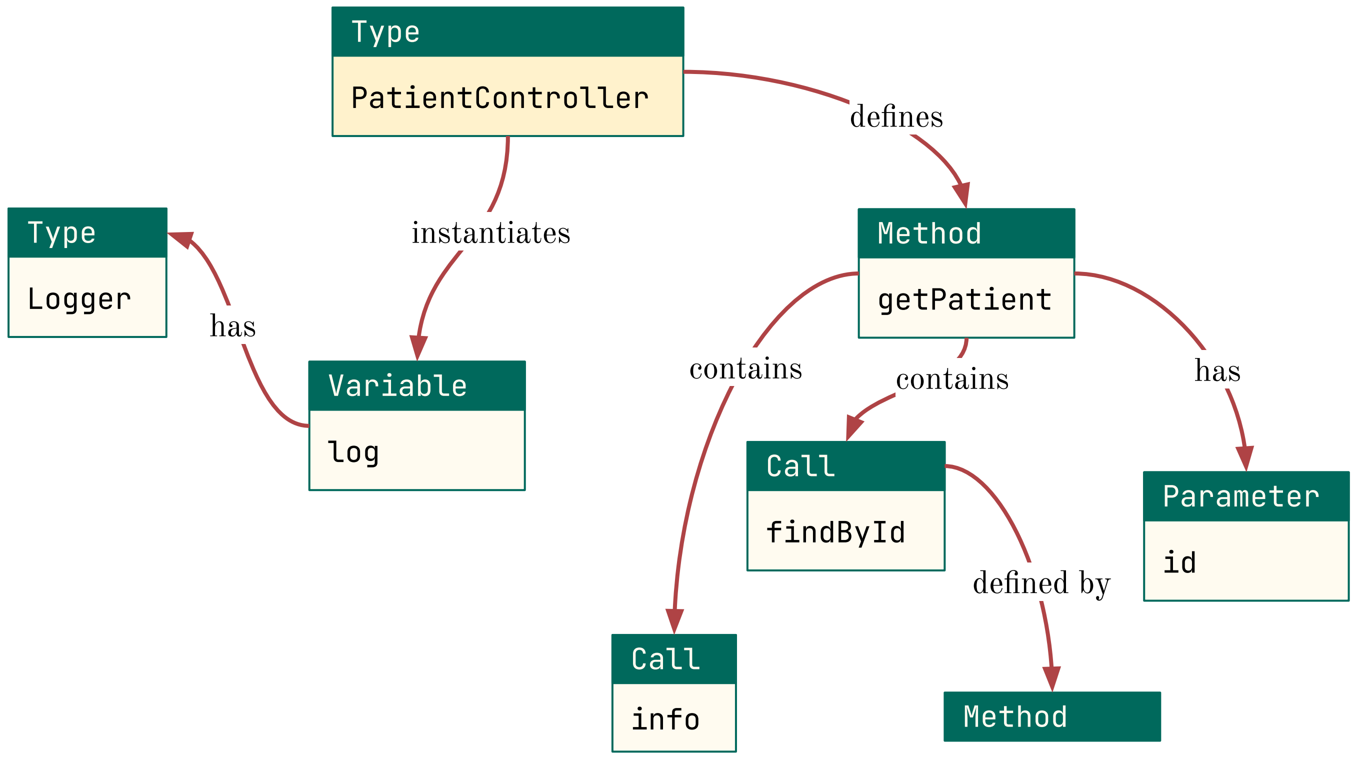


```
import org.springframework.web.bind.annotation.RestController;

@RestController
public class PatientController {

  private static Logger log =
            LoggerFactory.getLogger(PatientController.class);

  ...

  @RequestMapping(value = "/patients", method = RequestMethod.GET)
  public Iterable<Patient> getPatient(Int id) {
      Patient pat = patientRepository.findById(id);

      if (pat ≠ null) {
          log.info("First Patient is {}", pat.toString());
      }

      return patientRepository.findAll();
}
```

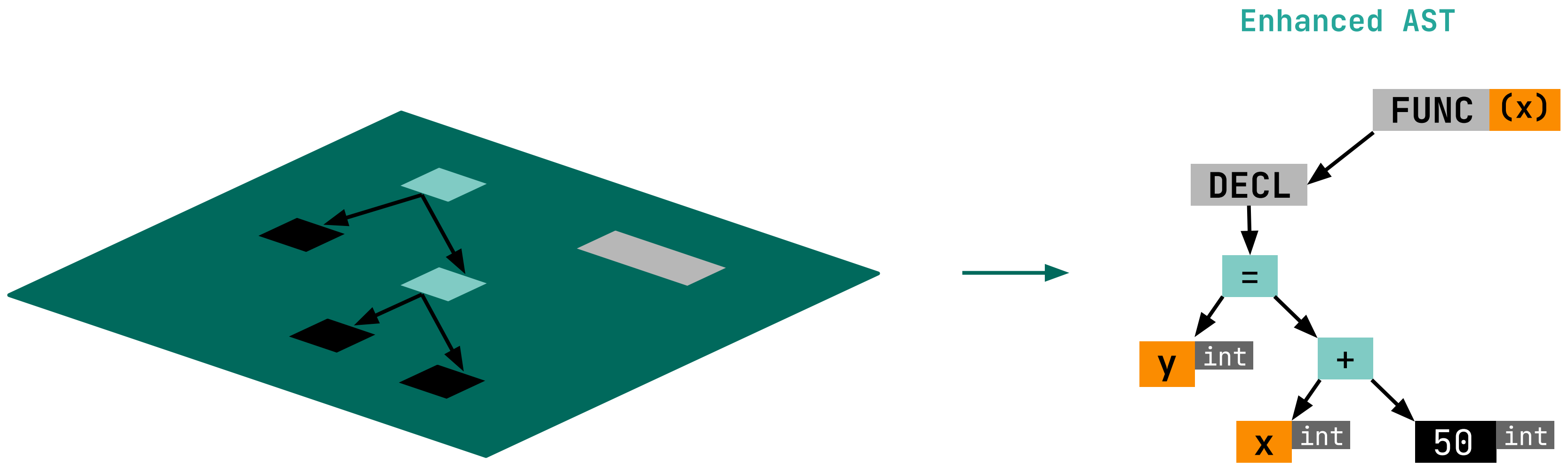
Higher Level Abstractions in *Programs*



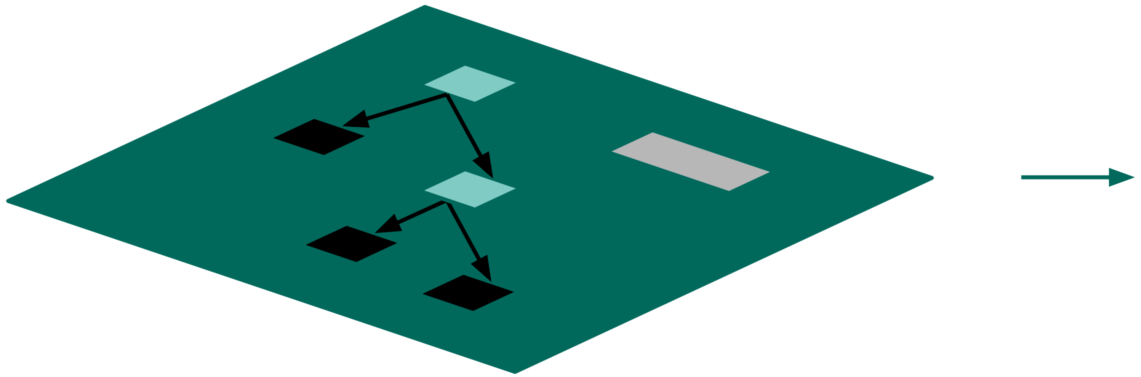
Higher Level Abstractions in *Programs*



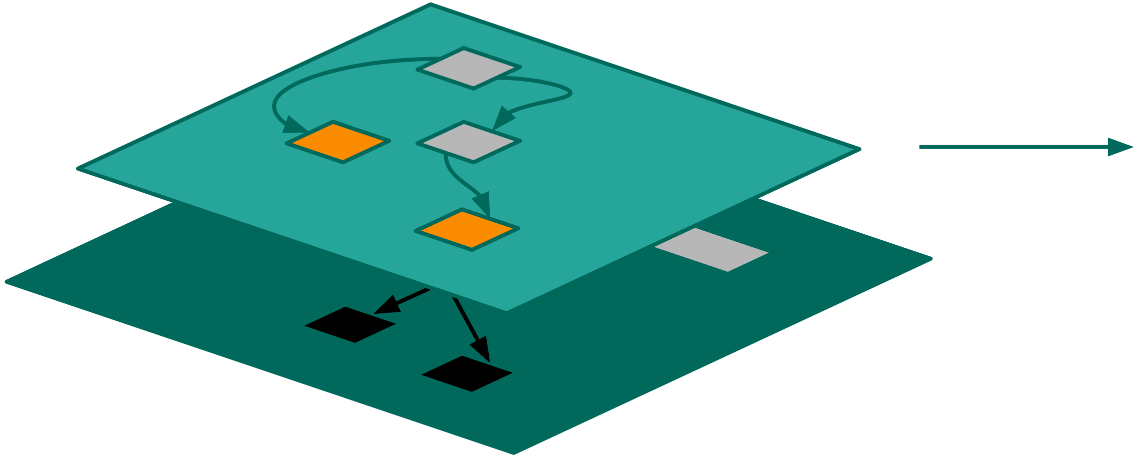
Building a Graph from Code

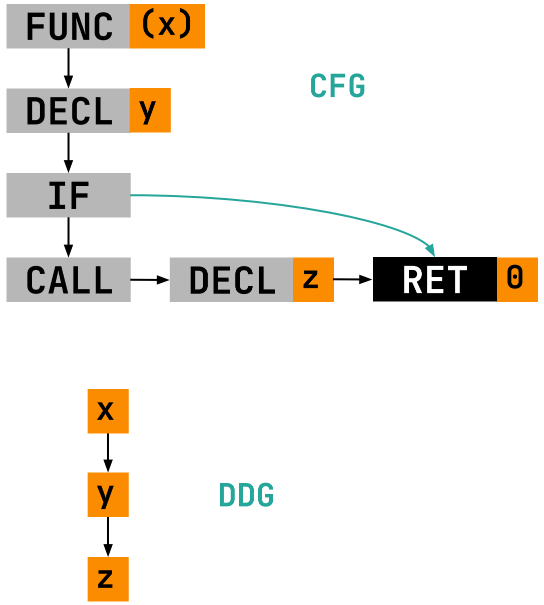


Building a Graph from Code

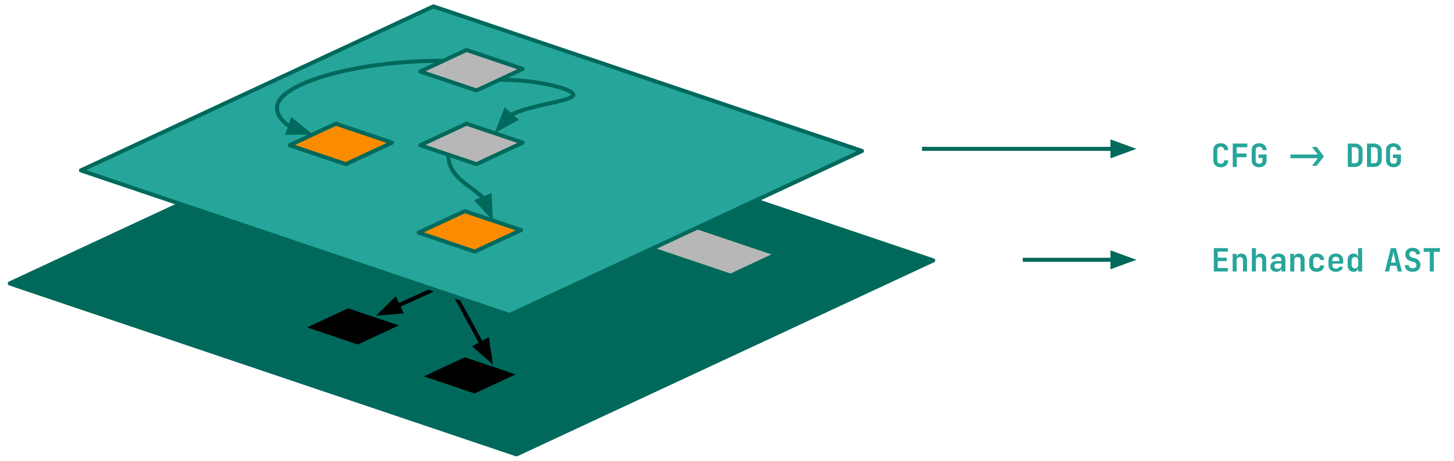


Building a Graph from Code

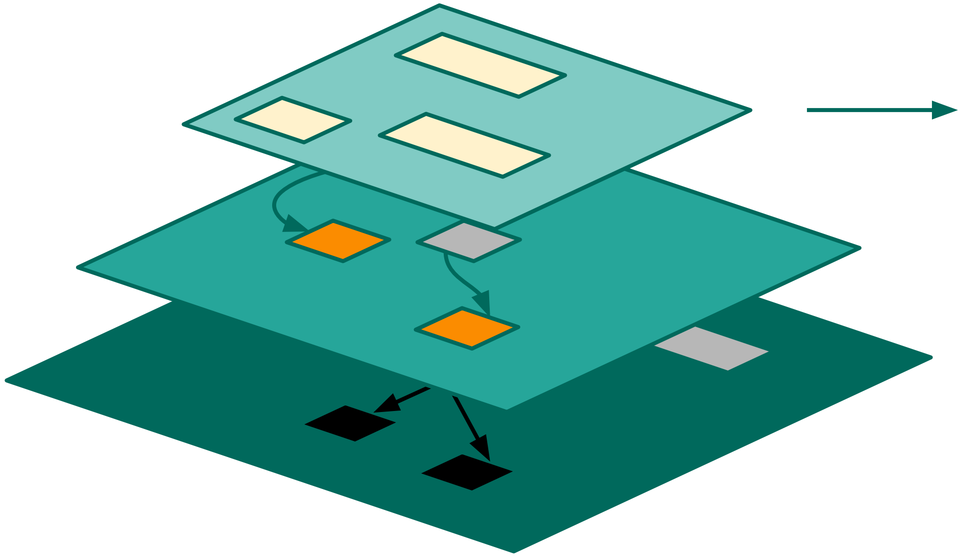


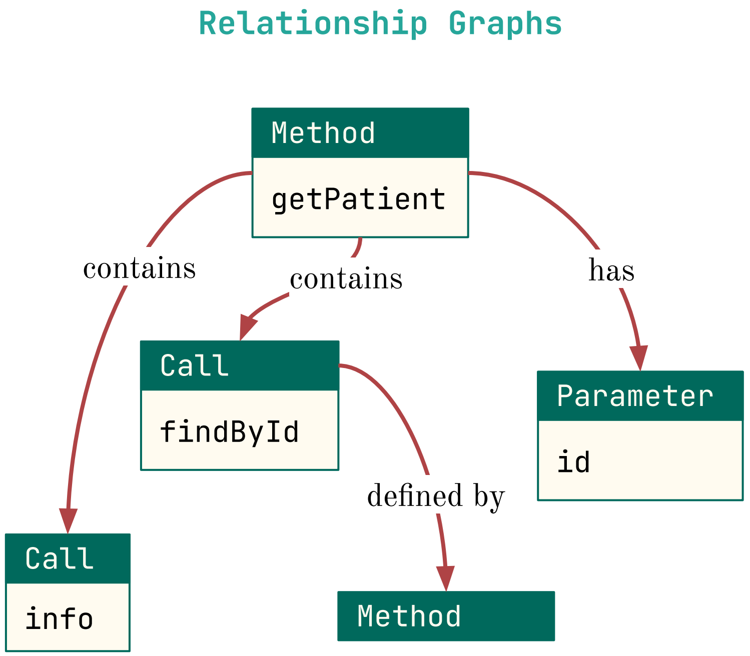


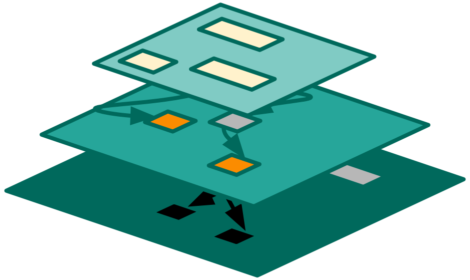
Building a Graph from Code



Building a Graph from Code







Code Property Graph (CPG)[1]

A queryable graph that embeds code knowledge

[1] *https://en.wikipedia.org/wiki/Code_property_graph*

Hands-on Workshop

Module 1	*Code Navigation and Insights*
Module 2	*Finding Dataflows*
Module 3	*Building an Analysis Tool*

Module 1

Code Navigation and Insights

JOERN

(yo-urn)

Module 1

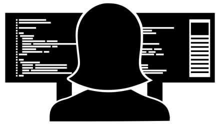
1. What is Joern?

Framework for *understanding* code and building code analysis tools

Query

List all functions in code which have more than one *if-else* statements

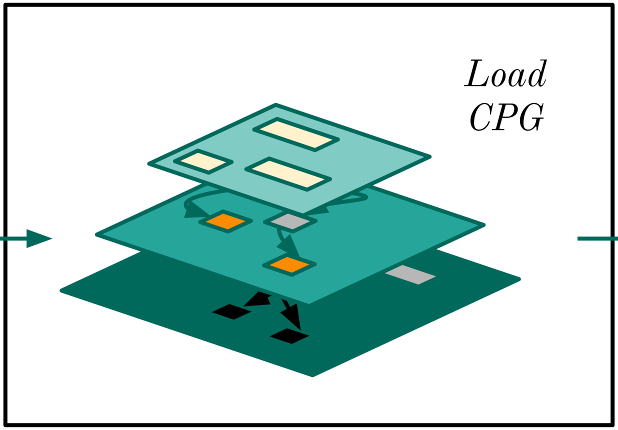
Scala

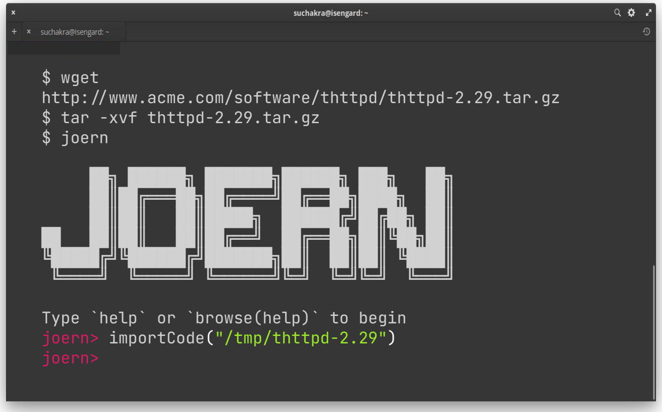


Source Code

LLM

Generate CPG





```
$ wget
http://www.acme.com/software/thttpd/thttpd-2.29.tar.gz
$ tar -xvf thttpd-2.29.tar.gz
$ joern
JOERN

Type `help` or `browse(help)` to begin
joern> importCode("/tmp/thttpd-2.29")
joern>
```

Insights

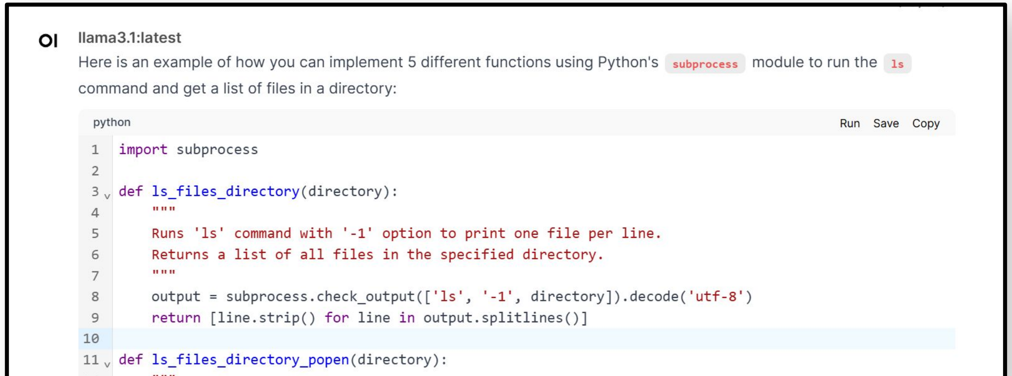
Module 1

1.1 Code Generation

Generate code to list files in a directory

PROMPT: Write a python file that implements 5 functions that all take a directory path (“directory“) as input and return a list of all the files in the specified directory. All the functions with code variations should use “ls” command to list files in a directory.

- Generated using llama-3.1
- Code also present in `2025-PROGRAM/module-1-files`

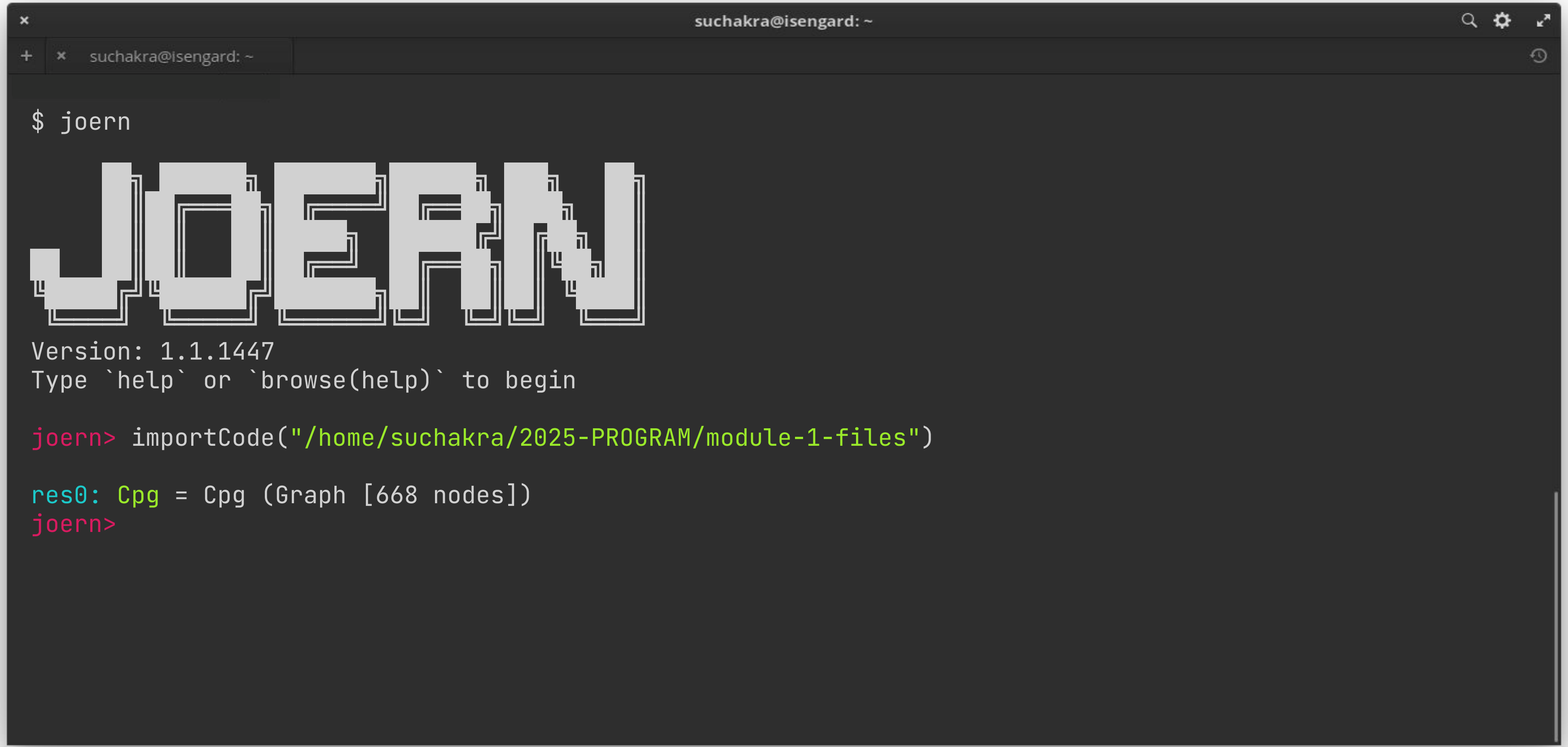

llama3.1:latest

Here is an example of how you can implement 5 different functions using Python's `subprocess` module to run the `ls` command and get a list of files in a directory:

```python
import subprocess

def ls_files_directory(directory):
    """
    Runs 'ls' command with '-1' option to print one file per line.
    Returns a list of all files in the specified directory.
    """
    output = subprocess.check_output(['ls', '-1', directory]).decode('utf-8')
    return [line.strip() for line in output.splitlines()]

def ls_files_directory_popen(directory):
```

Module 1

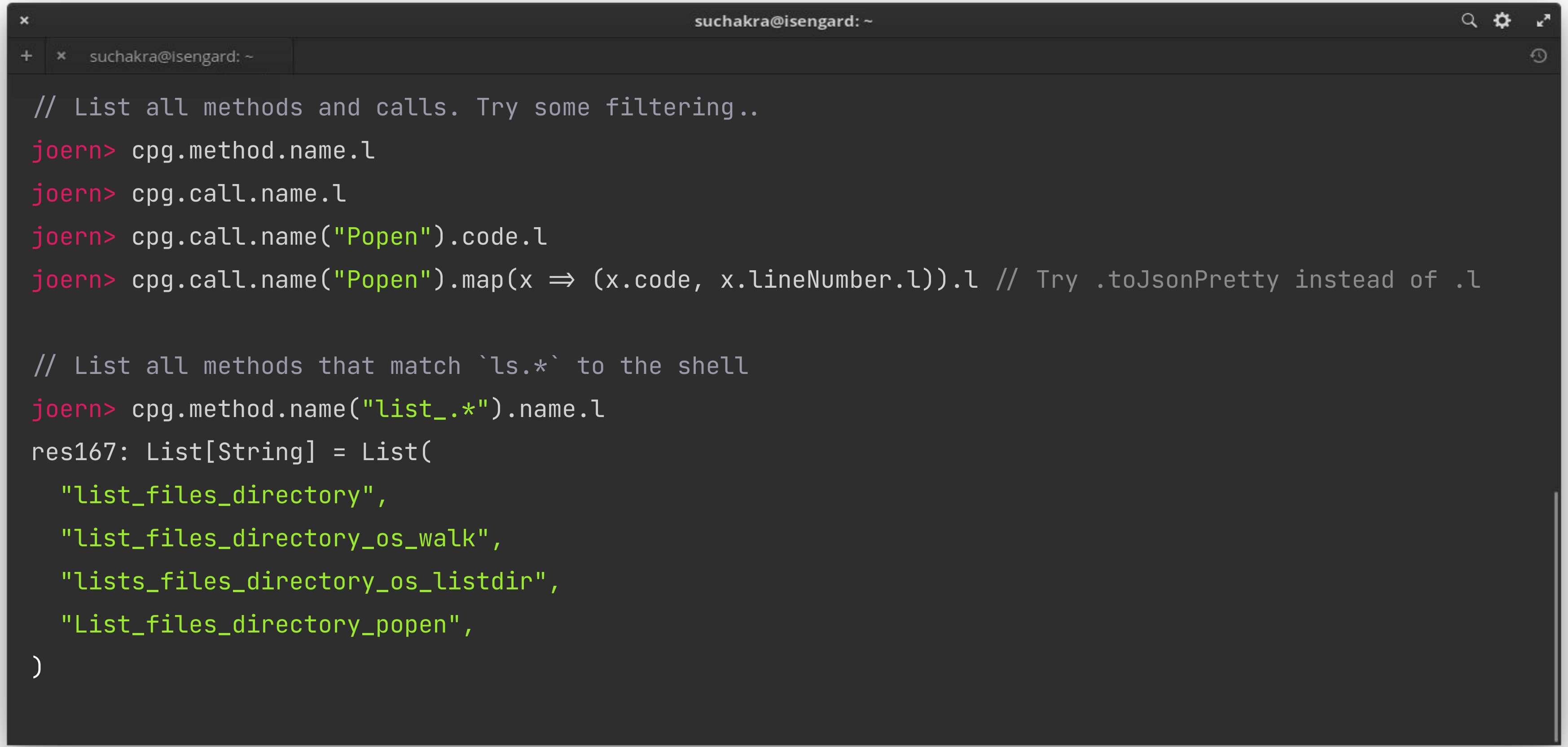
1.2 Load and Generate CPG



```
$ joern

Version: 1.1.1447
Type `help` or `browse(help)` to begin

joern> importCode("/home/suchakra/2025-PROGRAM/module-1-files")

res0: Cpg = Cpg (Graph [668 nodes])
joern>
```

Module 1

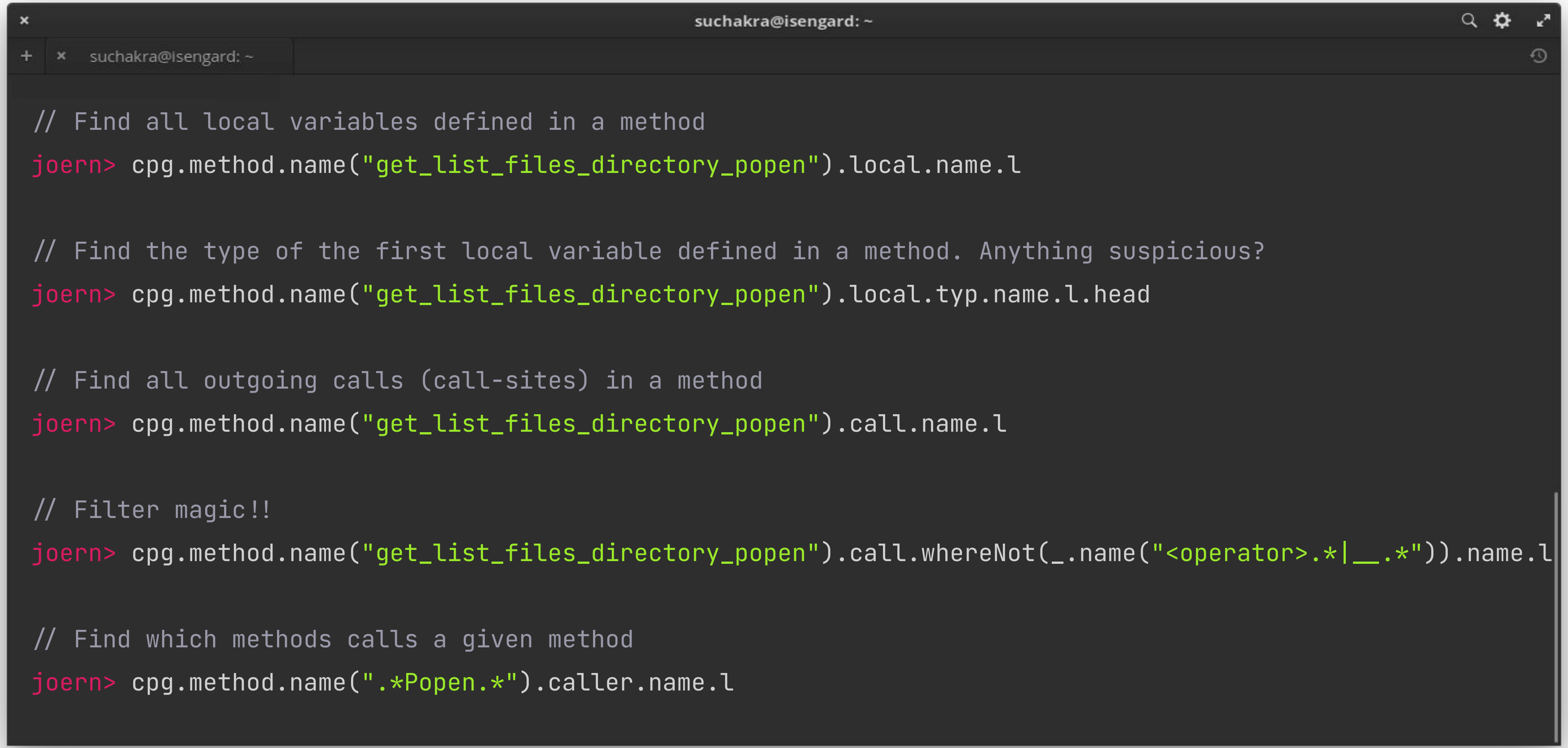
1.3 Basic Navigation - Methods & Calls



```
// List all methods and calls. Try some filtering..
joern> cpg.method.name.l
joern> cpg.call.name.l
joern> cpg.call.name("Popen").code.l
joern> cpg.call.name("Popen").map(x ⇒ (x.code, x.lineNumber.l)).l // Try .toJsonPretty instead of .l

// List all methods that match `ls.*` to the shell
joern> cpg.method.name("list_.*").name.l
res167: List[String] = List(
  "list_files_directory",
  "list_files_directory_os_walk",
  "lists_files_directory_os_listdir",
  "List_files_directory_popen",
)
```

Module 1

1.3 Basic Navigation - Methods & Calls



suchakra@isengard: ~

```
// Find all local variables defined in a method
joern> cpg.method.name("get_list_files_directory_popen").local.name.l

// Find the type of the first local variable defined in a method. Anything suspicious?
joern> cpg.method.name("get_list_files_directory_popen").local.typ.name.l.head

// Find all outgoing calls (call-sites) in a method
joern> cpg.method.name("get_list_files_directory_popen").call.name.l

// Filter magic!!
joern> cpg.method.name("get_list_files_directory_popen").call.whereNot(_.name("<operator>.*|__.*")).name.l

// Find which methods calls a given method
joern> cpg.method.name(".*Popen.*").caller.name.l
```

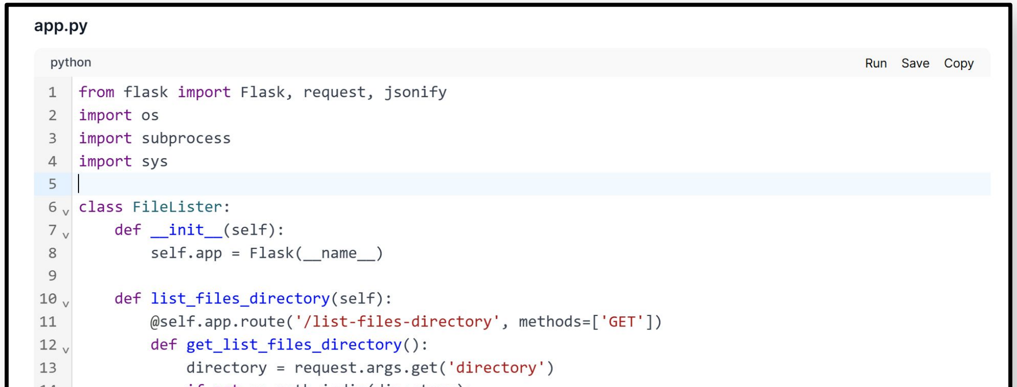
Module 1

1.4 Update Generated Code

Update previous code to create a web app

PROMPT: Create a web application using the above 5 variants of function. The web app should have 5 different routes corresponding to each function and should allow user to specify a directory as a request argument. When a user uses the app, the result should be returned as JSON. Use object oriented programming style. If the user inputs /etc/passwd as the request argument then return "Permission denied" in JSON.

- Code also present in `2025-PROGRAM/module-1-web`



app.py

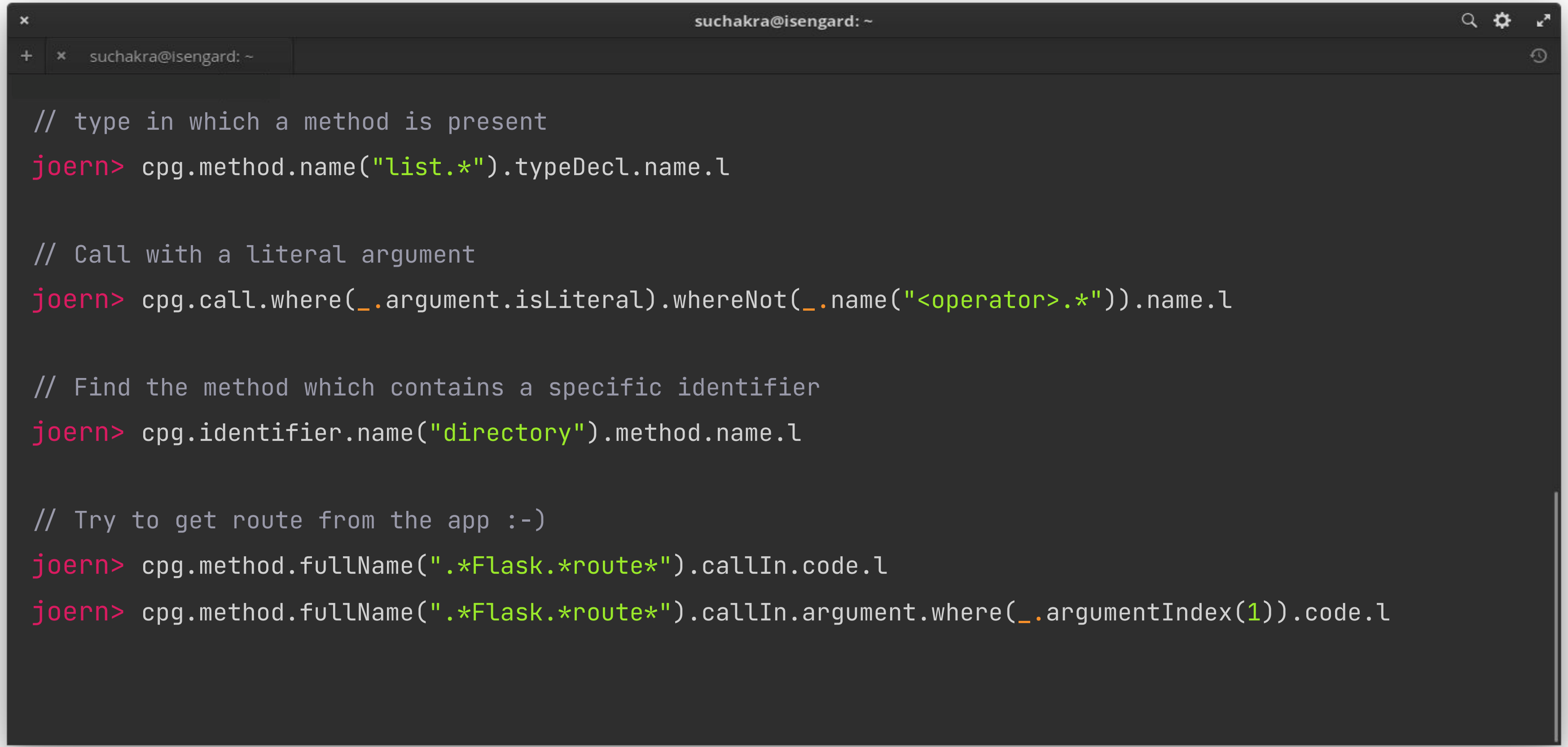
python

Run Save Copy

```python
from flask import Flask, request, jsonify
import os
import subprocess
import sys

class FileLister:
    def __init__(self):
        self.app = Flask(__name__)

    def list_files_directory(self):
        @self.app.route('/list-files-directory', methods=['GET'])
        def get_list_files_directory():
            directory = request.args.get('directory')
```

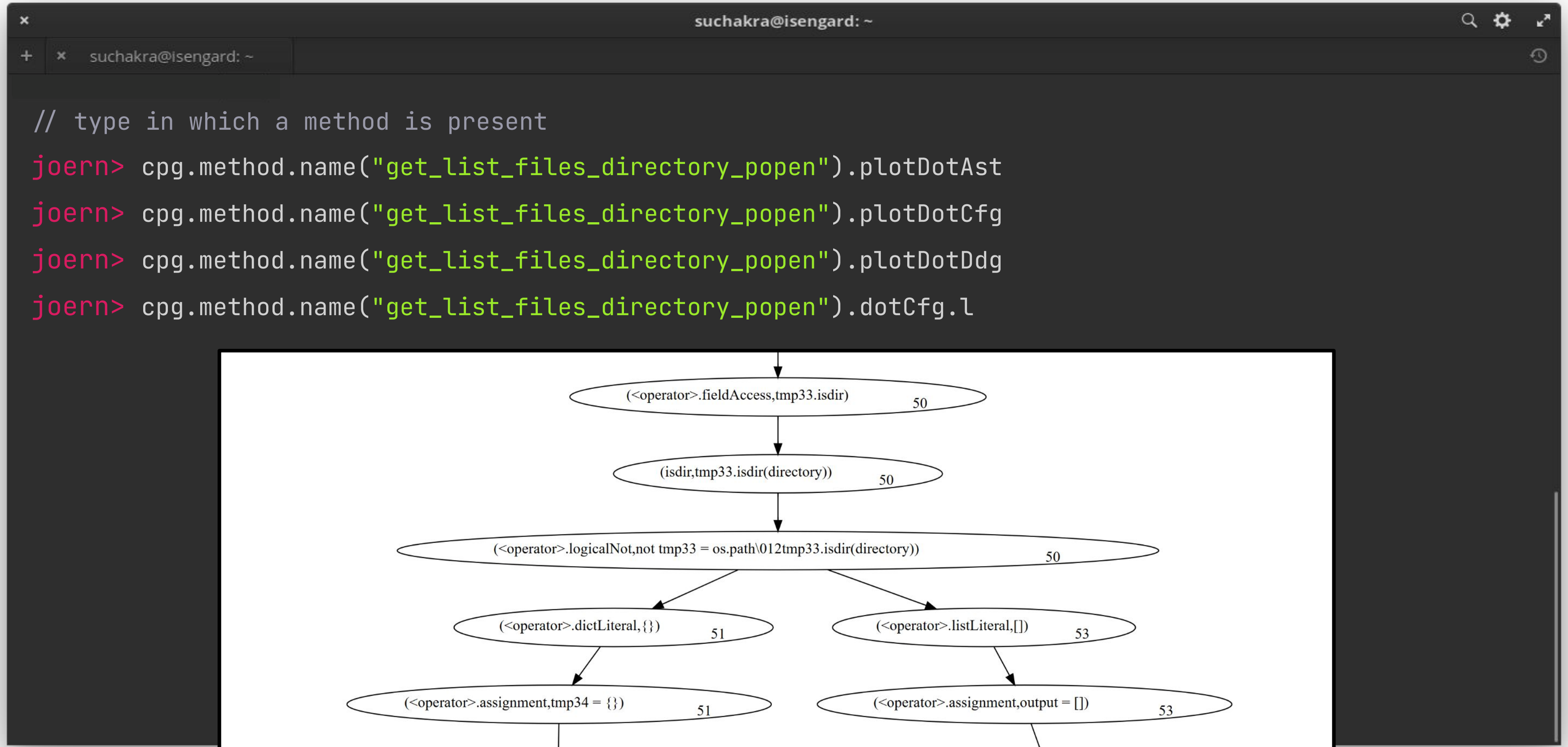
Module 1

1.5 More Insights - Classes, Literals, Identifiers

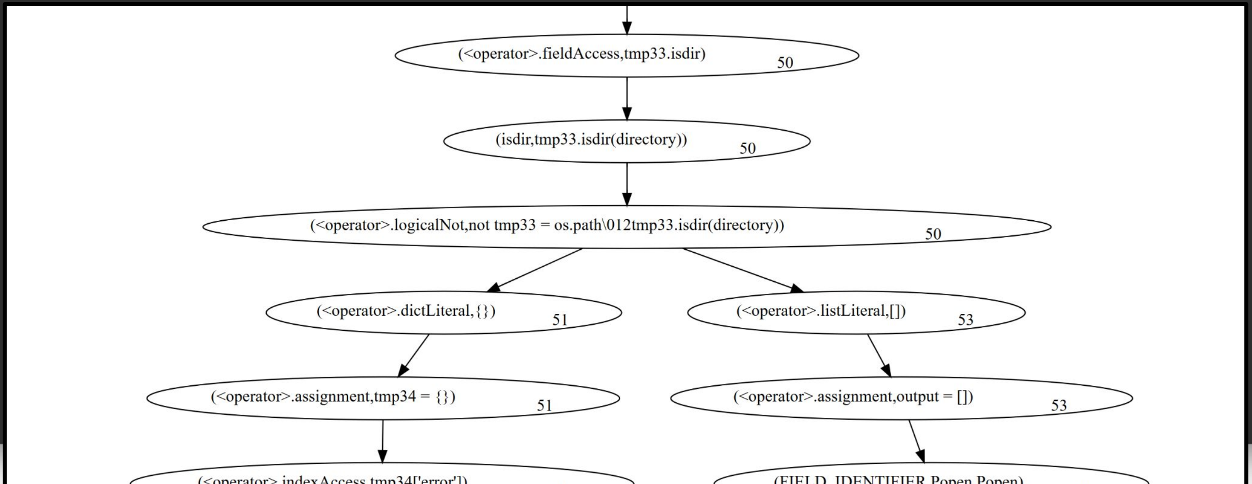


```
// type in which a method is present
joern> cpg.method.name("list.*").typeDecl.name.l

// Call with a literal argument
joern> cpg.call.where(_.argument.isLiteral).whereNot(_.name("<operator>.*")).name.l

// Find the method which contains a specific identifier
joern> cpg.identifier.name("directory").method.name.l

// Try to get route from the app :-)
joern> cpg.method.fullName(".*Flask.*route*").callIn.code.l
joern> cpg.method.fullName(".*Flask.*route*").callIn.argument.where(_.argumentIndex(1)).code.l
```

1.6 Visualizing the Graph



```
// type in which a method is present
joern> cpg.method.name("get_list_files_directory_popen").plotDotAst
joern> cpg.method.name("get_list_files_directory_popen").plotDotCfg
joern> cpg.method.name("get_list_files_directory_popen").plotDotDdg
joern> cpg.method.name("get_list_files_directory_popen").dotCfg.l
```



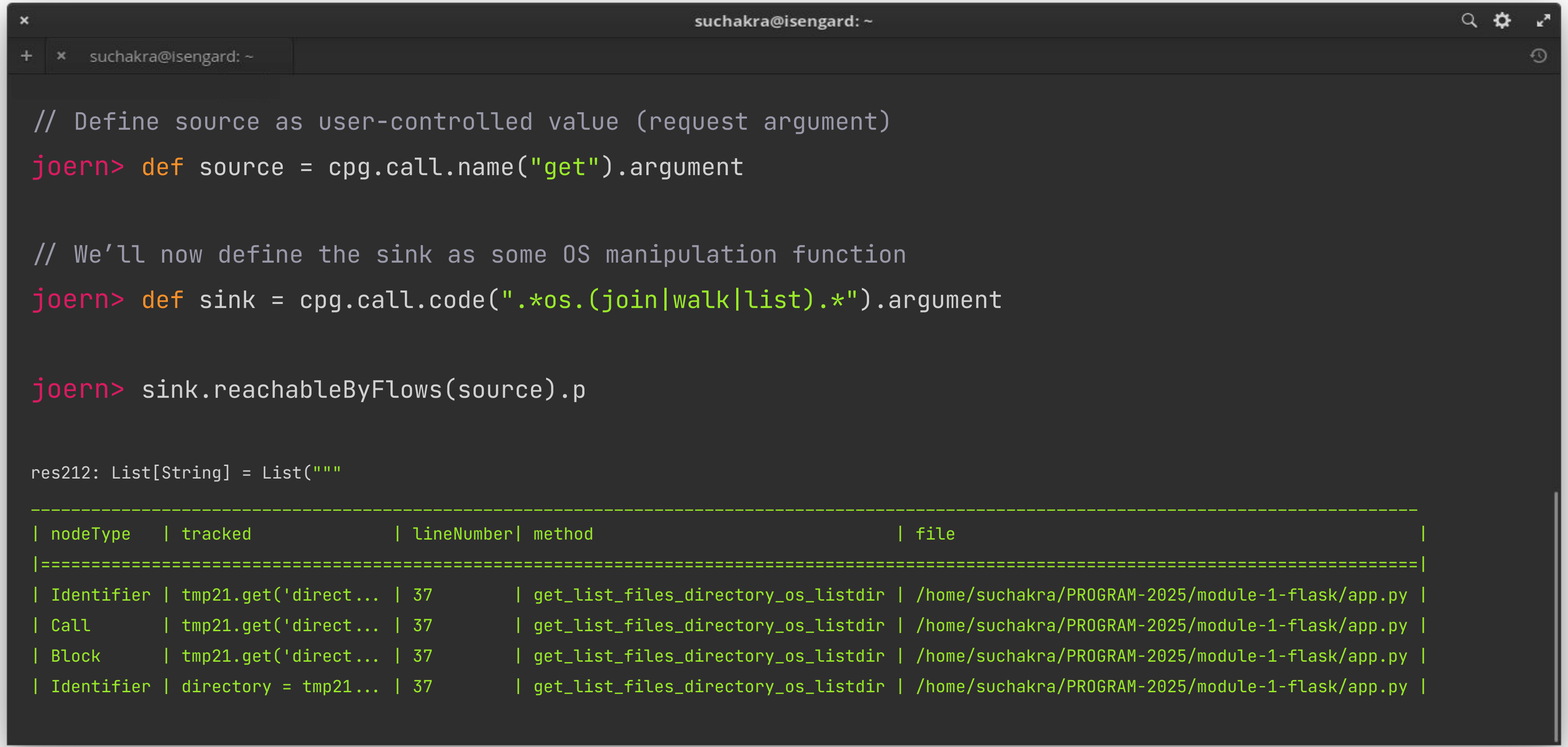
(<operator>.fieldAccess,tmp33.isdir) 50
(isdir,tmp33.isdir(directory)) 50
(<operator>.logicalNot,not tmp33 = os.path\012tmp33.isdir(directory)) 50
(<operator>.dictLiteral,{}) 51
(<operator>.listLiteral,[]) 53
(<operator>.assignment,tmp34 = {}) 51
(<operator>.assignment,output = []) 53
(<operator>.indexAccess,tmp34['error'])
(FIELD_IDENTIFIER,Popen,Popen)

Module 2

Finding Dataflows

Module 2

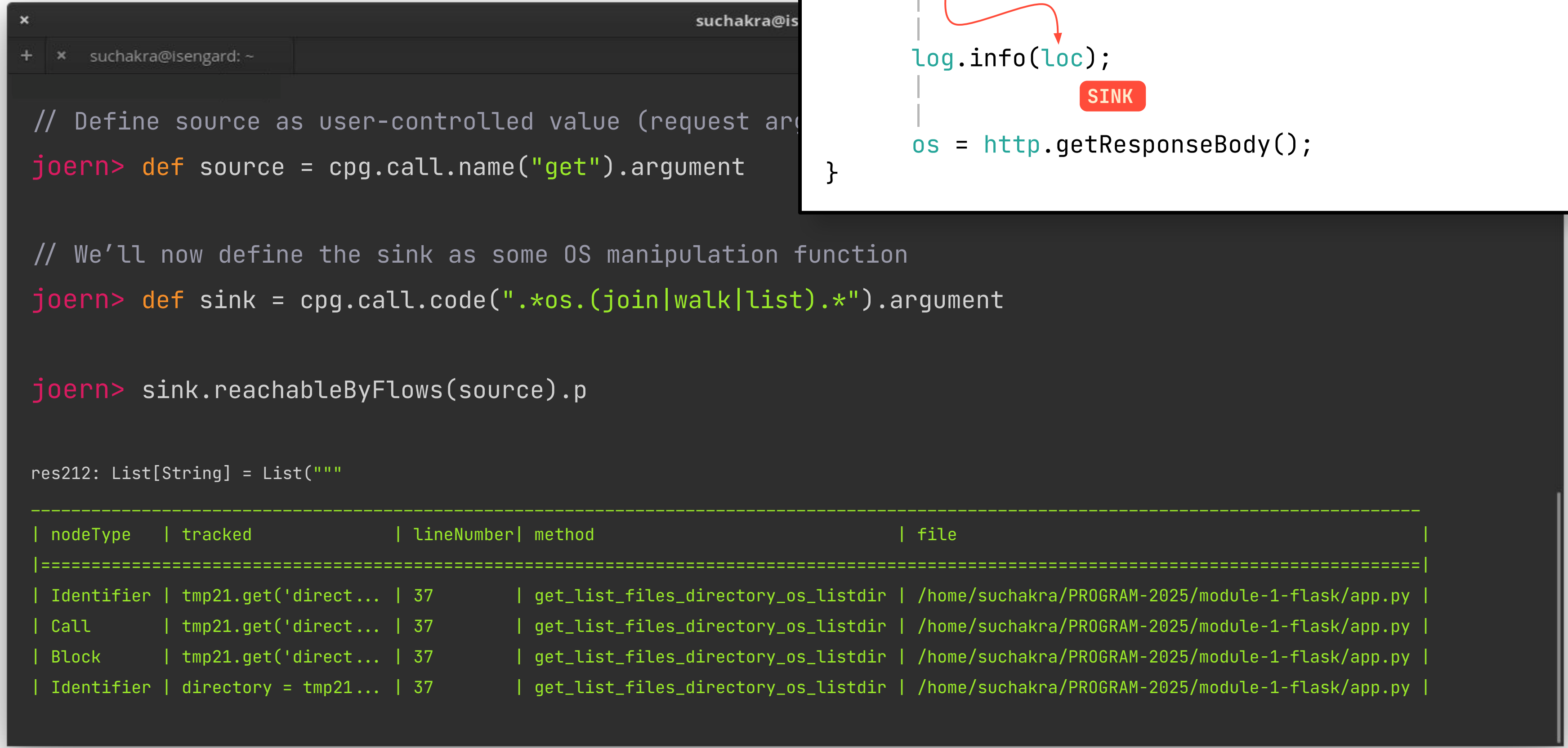
2.1 Dataflow Traversals



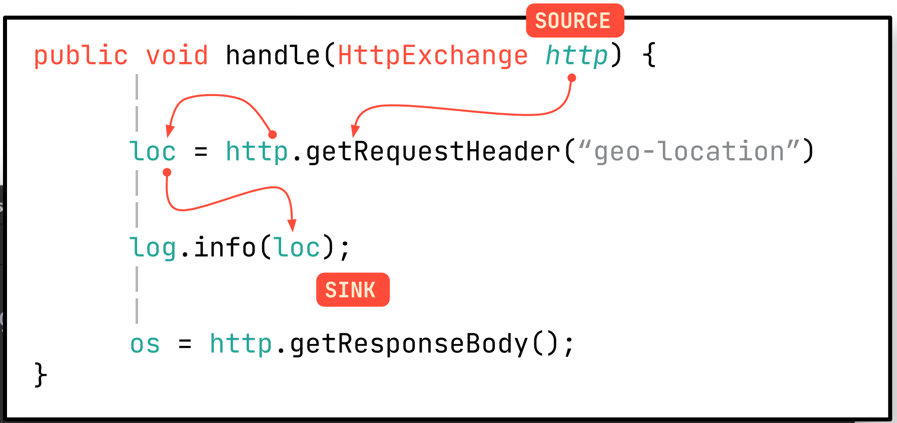
```
// Define source as user-controlled value (request argument)
joern> def source = cpg.call.name("get").argument

// We'll now define the sink as some OS manipulation function
joern> def sink = cpg.call.code(".*os.(join|walk|list).*").argument

joern> sink.reachableByFlows(source).p

res212: List[String] = List("""
------------------------------------------------------------------------------------------------------------------------------
| nodeType   | tracked               | lineNumber| method                               | file                                                   |
|============================================================================================================================|
| Identifier | tmp21.get('direct... | 37        | get_list_files_directory_os_listdir | /home/suchakra/PROGRAM-2025/module-1-flask/app.py |
| Call       | tmp21.get('direct... | 37        | get_list_files_directory_os_listdir | /home/suchakra/PROGRAM-2025/module-1-flask/app.py |
| Block      | tmp21.get('direct... | 37        | get_list_files_directory_os_listdir | /home/suchakra/PROGRAM-2025/module-1-flask/app.py |
| Identifier | directory = tmp21... | 37        | get_list_files_directory_os_listdir | /home/suchakra/PROGRAM-2025/module-1-flask/app.py |
```

Module 2

2.1 Dataflow Traversals



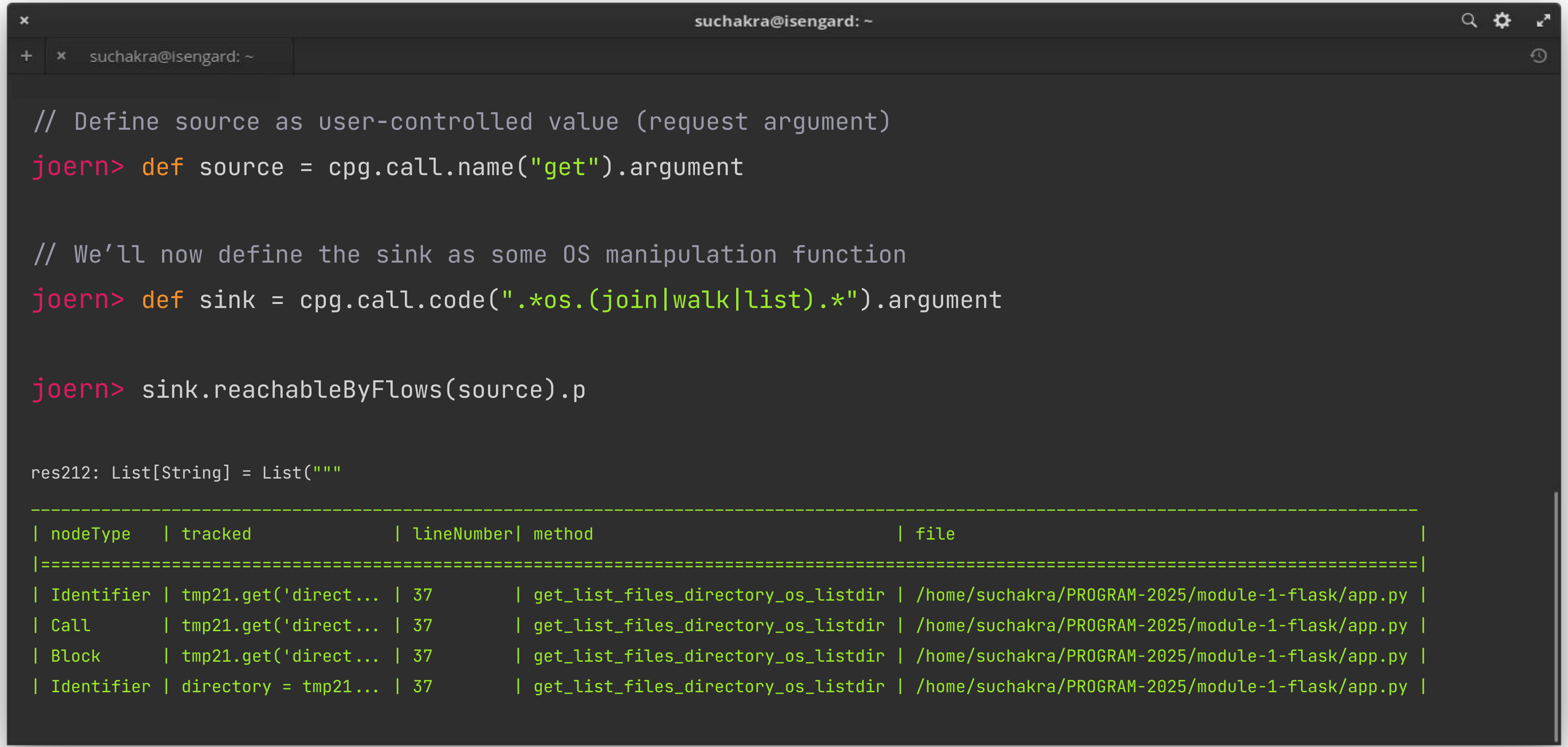
```
suchakra@is
suchakra@isengard: ~

// Define source as user-controlled value (request ar
joern> def source = cpg.call.name("get").argument

// We'll now define the sink as some OS manipulation function
joern> def sink = cpg.call.code(".*os.(join|walk|list).*").argument

joern> sink.reachableByFlows(source).p

res212: List[String] = List("""
_____________________________________________________________________________________________________________________________
| nodeType   | tracked              | lineNumber| method                            | file                                           |
|==========================================================================================================================|
| Identifier | tmp21.get('direct... | 37        | get_list_files_directory_os_listdir | /home/suchakra/PROGRAM-2025/module-1-flask/app.py |
| Call       | tmp21.get('direct... | 37        | get_list_files_directory_os_listdir | /home/suchakra/PROGRAM-2025/module-1-flask/app.py |
| Block      | tmp21.get('direct... | 37        | get_list_files_directory_os_listdir | /home/suchakra/PROGRAM-2025/module-1-flask/app.py |
| Identifier | directory = tmp21... | 37        | get_list_files_directory_os_listdir | /home/suchakra/PROGRAM-2025/module-1-flask/app.py |
```



```
public void handle(HttpExchange http) {
        loc = http.getRequestHeader("geo-location")
        log.info(loc);
        os = http.getResponseBody();
}
```

Module 2

2.1 Dataflow Traversals



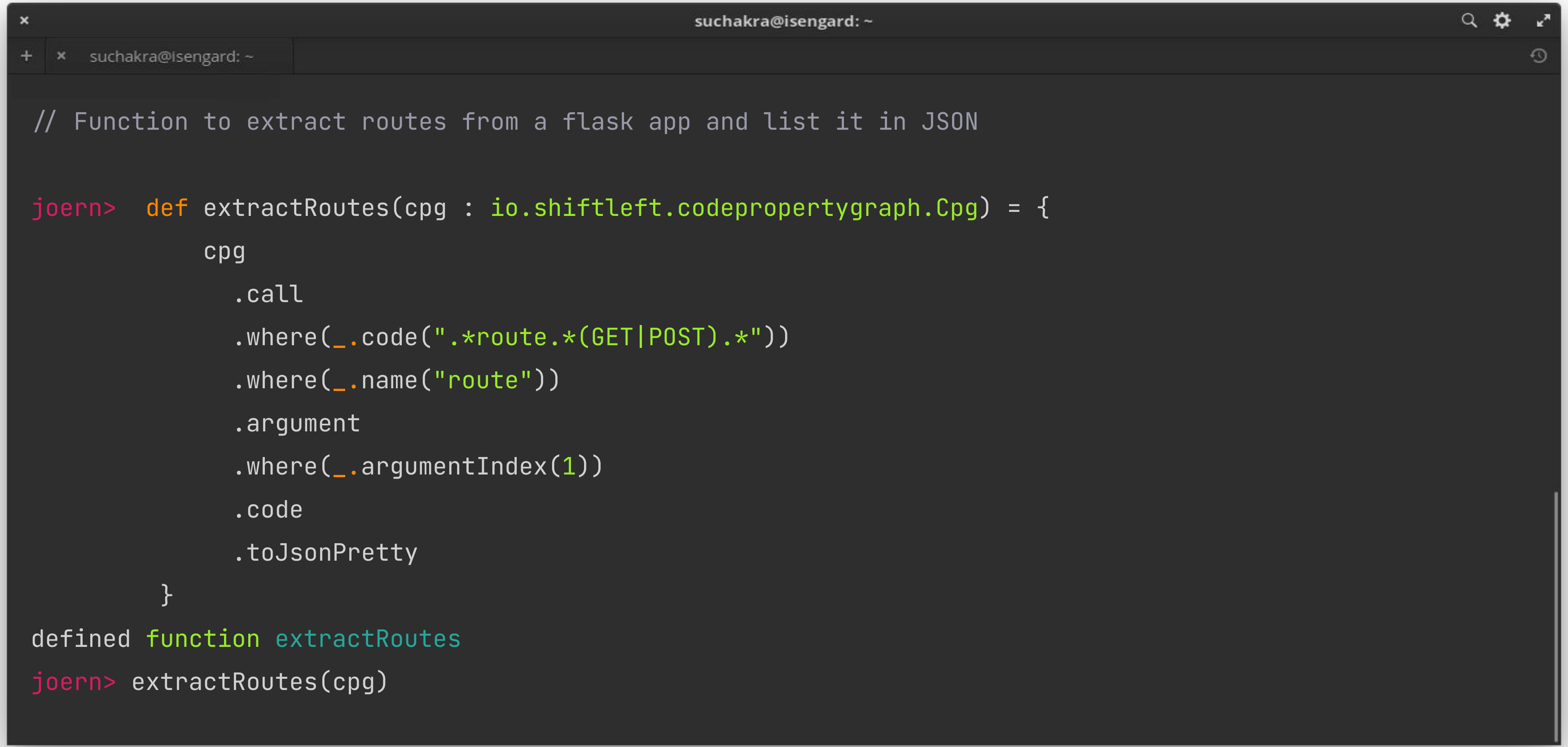
```
// Define source as user-controlled value (request argument)
joern> def source = cpg.call.name("get").argument

// We'll now define the sink as some OS manipulation function
joern> def sink = cpg.call.code(".*os.(join|walk|list).*").argument

joern> sink.reachableByFlows(source).p

res212: List[String] = List("""
-------------------------------------------------------------------------------------------------------------------------------------
| nodeType   | tracked               | lineNumber| method                            | file                                          |
|===================================================================================================================================|
| Identifier | tmp21.get('direct... | 37         | get_list_files_directory_os_listdir | /home/suchakra/PROGRAM-2025/module-1-flask/app.py |
| Call       | tmp21.get('direct... | 37         | get_list_files_directory_os_listdir | /home/suchakra/PROGRAM-2025/module-1-flask/app.py |
| Block      | tmp21.get('direct... | 37         | get_list_files_directory_os_listdir | /home/suchakra/PROGRAM-2025/module-1-flask/app.py |
| Identifier | directory = tmp21... | 37         | get_list_files_directory_os_listdir | /home/suchakra/PROGRAM-2025/module-1-flask/app.py |
```

Module 3

Building an Analysis Tool

Module 3

3.1 Functions in Joern

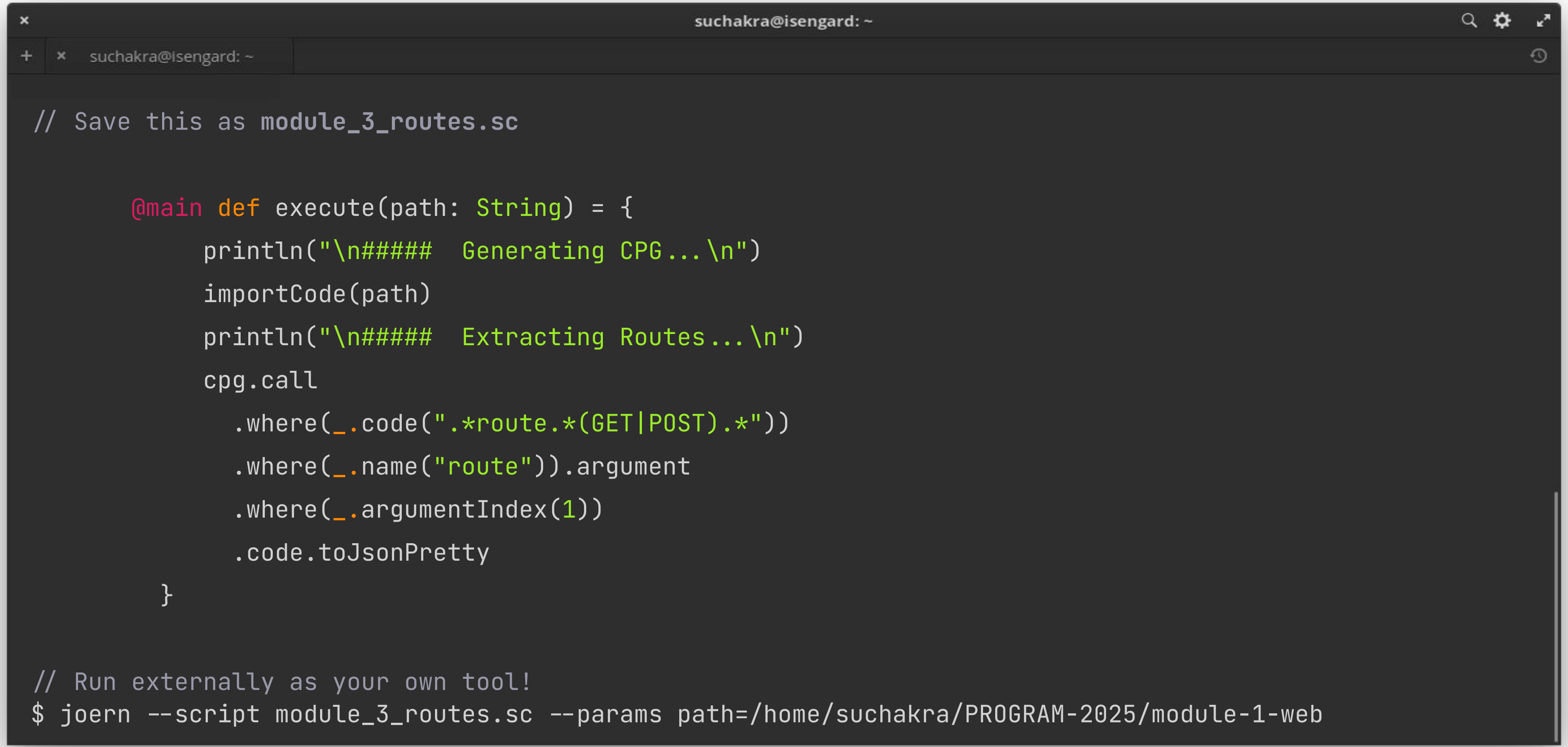


suchakra@isengard: ~

```
// Function to extract routes from a flask app and list it in JSON

joern>  def extractRoutes(cpg : io.shiftleft.codepropertygraph.Cpg) = {
            cpg
              .call
              .where(_.code(".*route.*(GET|POST).*"))
              .where(_.name("route"))
              .argument
              .where(_.argumentIndex(1))
              .code
              .toJsonPretty
         }
defined function extractRoutes
joern> extractRoutes(cpg)
```

Module 3

3.2 Scripting in Joern



```
// Save this as module_3_routes.sc

      @main def execute(path: String) = {
           println("\n#####  Generating CPG...\n")
           importCode(path)
           println("\n#####  Extracting Routes...\n")
           cpg.call
             .where(_.code(".*route.*(GET|POST).*"))
             .where(_.name("route")).argument
             .where(_.argumentIndex(1))
             .code.toJsonPretty
         }

// Run externally as your own tool!
$ joern --script module_3_routes.sc --params path=/home/suchakra/PROGRAM-2025/module-1-web
```

Additional Learning

- Joern: https://joern.io
- Docs: https://docs.joern.io
- **Joern Community:** https://discord.com/invite/vv4MH284Hc
- Tour of Scala: https://docs.scala-lang.org/tour/tour-of-scala.html
- Interesting queries: https://queries.joern.io/

Fin ~ Q&A

✉ *suchakra@privado.ai*

✉ *mail@suchakra.in*

@suchakra@mastodon.social