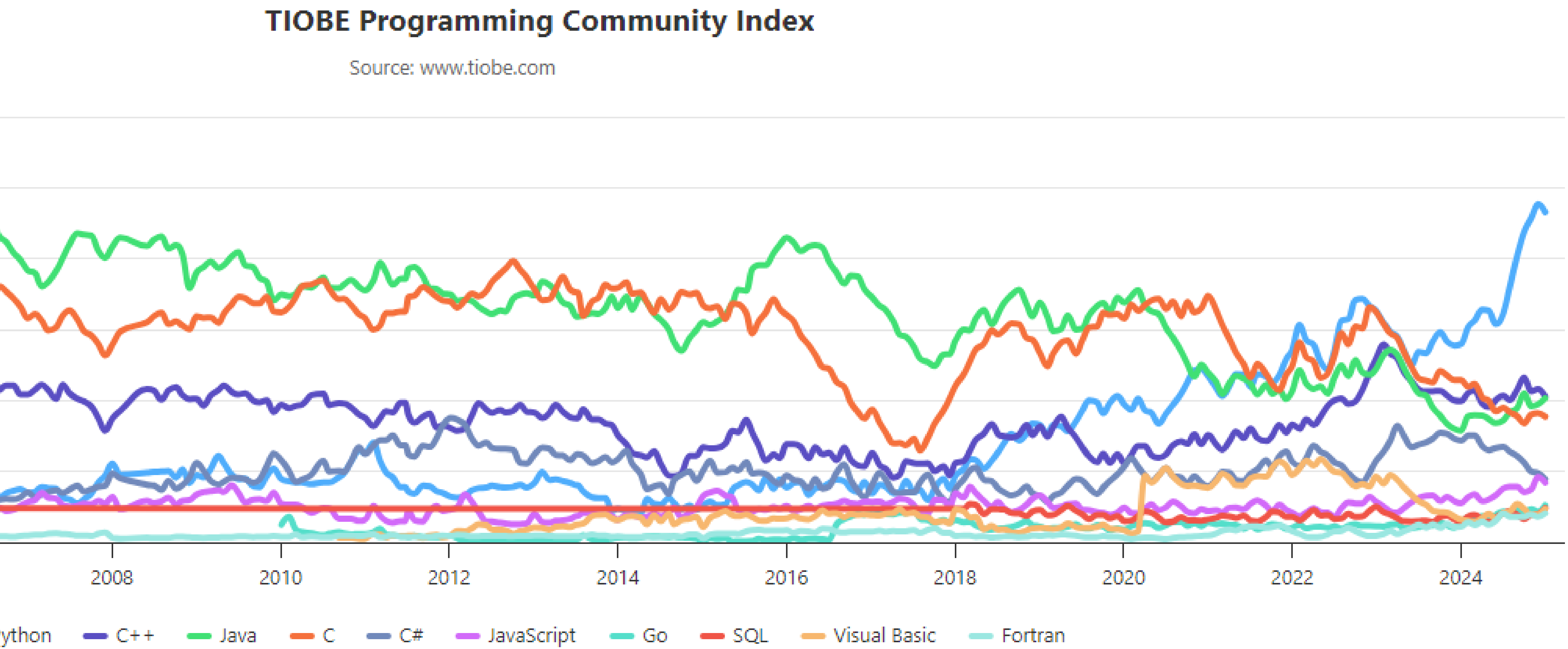
Programming for Social Scientists

Introduction

Johan A. Dornschneider-Elkink

```
from watson.framework import events
from watson.http.messages import Response, Request
from watson.common.imports import get_qualified_name
from watson.common.contextmanagers import suppress


ACCEPTABLE_RETURN_TYPES = (str, int, float, bool)


class Base(ContainerAware, metaclass=abc.ABCMeta):
    """The base class for all controllers.

    Attributes:
        __action__ (string): The last action that was called
    """
    def execute(self, **kwargs):
        method = self.get_execute_method(**kwargs)
        self.__action__ = method
        return method(**kwargs) or {}

    @abc.abstractmethod
    def get_execute_method(self, **kwargs):
```



TIOBE Programming Community Index

Source: www.tiobe.com

The TIOBE Programming Community index is an indicator of the **popularity** of programming languages. The index is updated once a month. The ratings are based on the **number of skilled engineers** world-wide, courses and third party vendors. Popular search engines such as Google, Bing, Yahoo!, Wikipedia, Amazon, YouTube and Baidu are used to calculate the ratings. It is important to note that the TIOBE index is **not about the *best* programming language** or the language in which *most lines of code* have been written.

Python

Scripting vs. programming

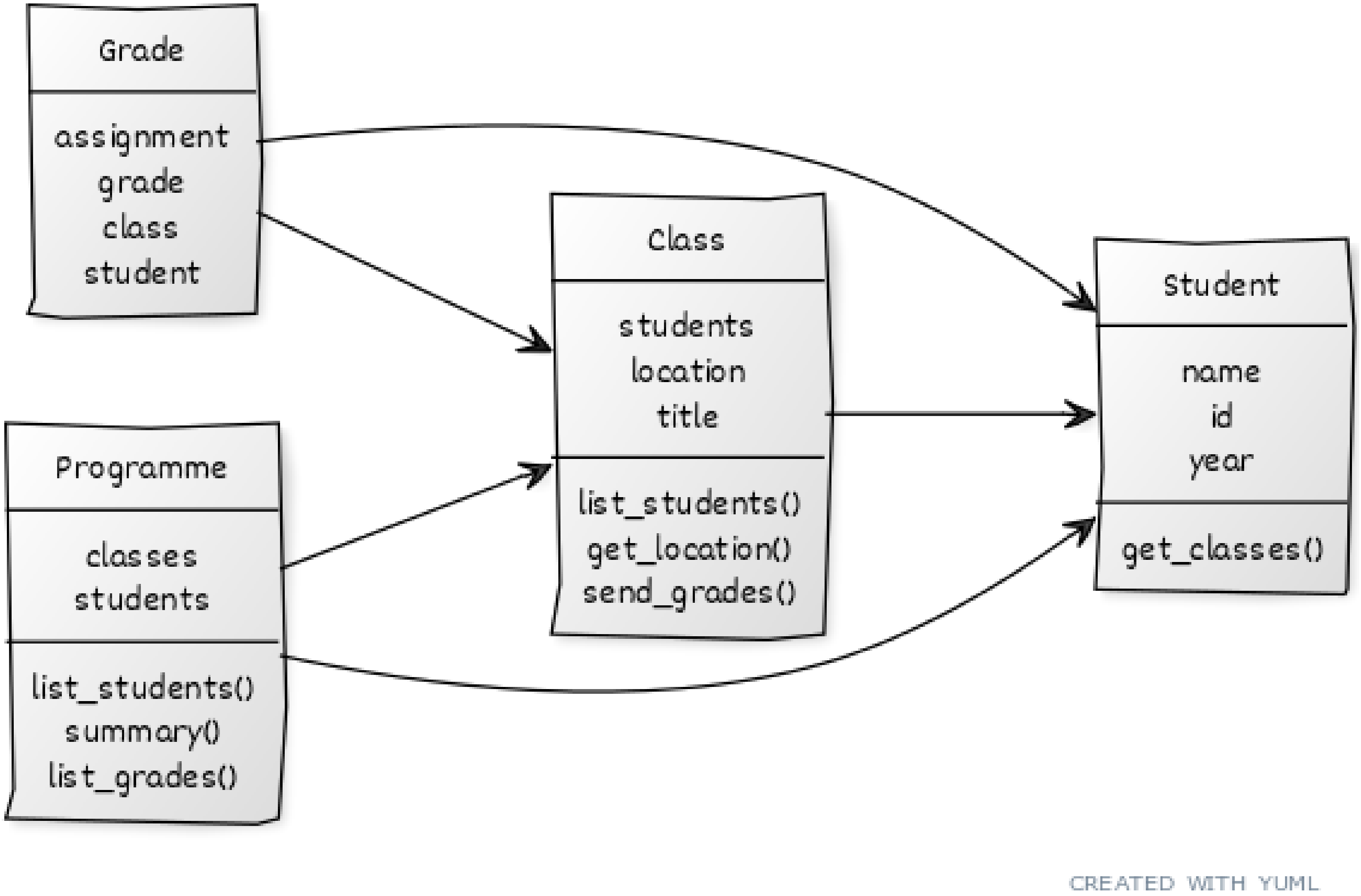
R vs. python

Applications

Object-oriented design

```
irror_mod = modifier_ob.
mirror object to mirror
irror_mod.mirror_object
peration == "MIRROR_X":
irror_mod.use_x = True
irror_mod.use_y = False
irror_mod.use_z = False
_operation == "MIRROR_Y":
irror_mod.use_x = False
irror_mod.use_y = True
irror_mod.use_z = False
_operation == "MIRROR_Z":
rror_mod.use_x = False
rror_mod.use_y = False
rror_mod.use_z = True

selection at the end -add
_ob.select= 1
er_ob.select=1
ntext.scene.objects.active
("Selected" + str(modifier
irror_ob.select = 0
 bpy.context.selected_obj
ata.objects[one.name].sel

int("please select exactly

-- OPERATOR CLASSES ----

types.Operator):
X mirror to the selected
ject.mirror_mirror_x"
rror X"

ontext):
ext.active_object is not
```

Object-oriented design



Tools

Matrix / Element

Github / git

Repl.it / Visual Studio Code

Local installation ?

POL42340 Programming for Social Scientists
installation instructions[*]

Johan A. Dornschneider-Elkink
https://www.joselkink.net

January 17, 2025

1 Python

1. Go to the Python website: `https://www.python.org/`
2. Download the latest version of Python, or at least version 3.10.
3. Run the installer. Make sure to check the box that says "Add Python to PATH".

Deitel, Paul and Harvey Deitel. 2022. *Intro to Python for computer science and data science.* Global edition ed. Harlow, UK: Pearson.

Gamma, Erich. 1995. *Design patterns: Elements of reusable object-oriented software.* Addison-Wesley.

Giridhar, Chetan. 2016. *Learning Python design patterns.* Packt Publishing Ltd.

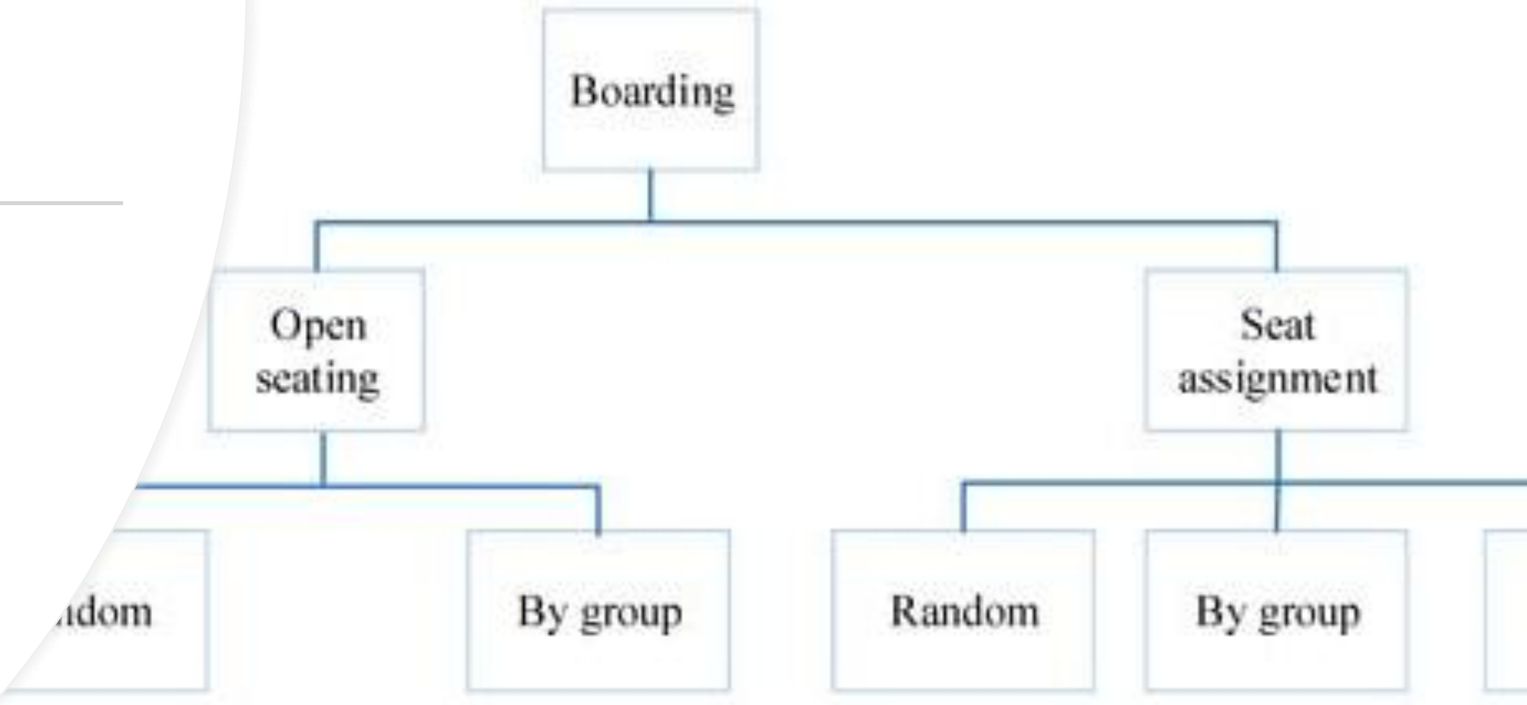
Laver, Michael. 2005. "Policy and the dynamics of political competition." *American Political Science Review* pp. 263–281.

Lubanovic, Bill. 2020. *Introducing Python: Modern computing in simple packages.* 2nd edition ed. O'Reilly.

Muis, Jasper. 2010. "Simulating political stability and change in the Netherlands (1998-2002): An agent-based model of party competition with media effects empirically tested." *Journal of Artificial Societies and Social Simulation* 13(2):4.

MCQ test 1	21 Feb	20%
MCQ test 2	28 Mar	20%
MCQ test 3	25 Apr	20%
Class diagram	28 Mar, 1 pm	10%
Lab report	8 May, 5 pm	30%

Agent-based simulation



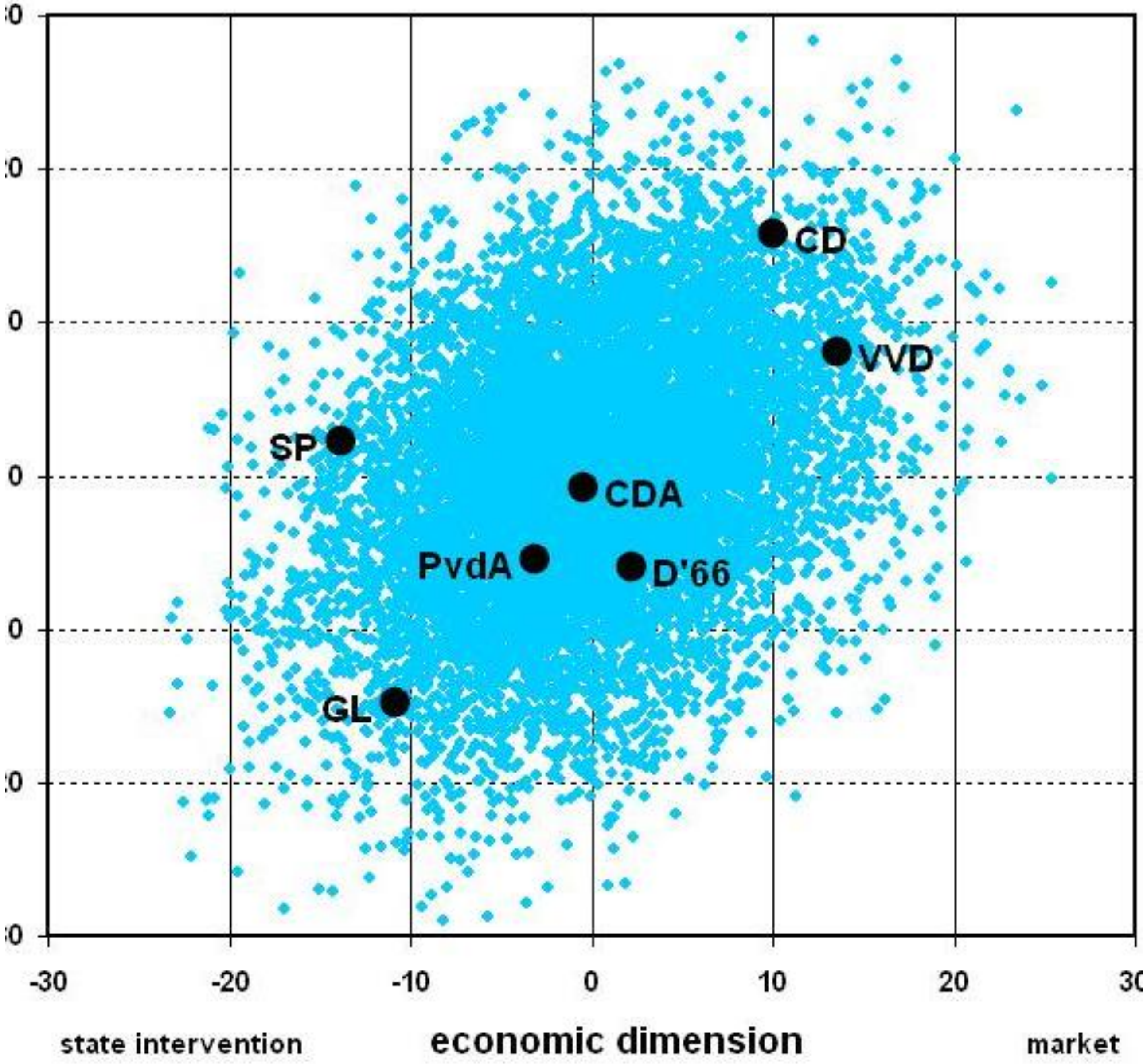
American Political Science Review Vol. 99, No. 2 May 2005

Policy and the Dynamics of Political Competition

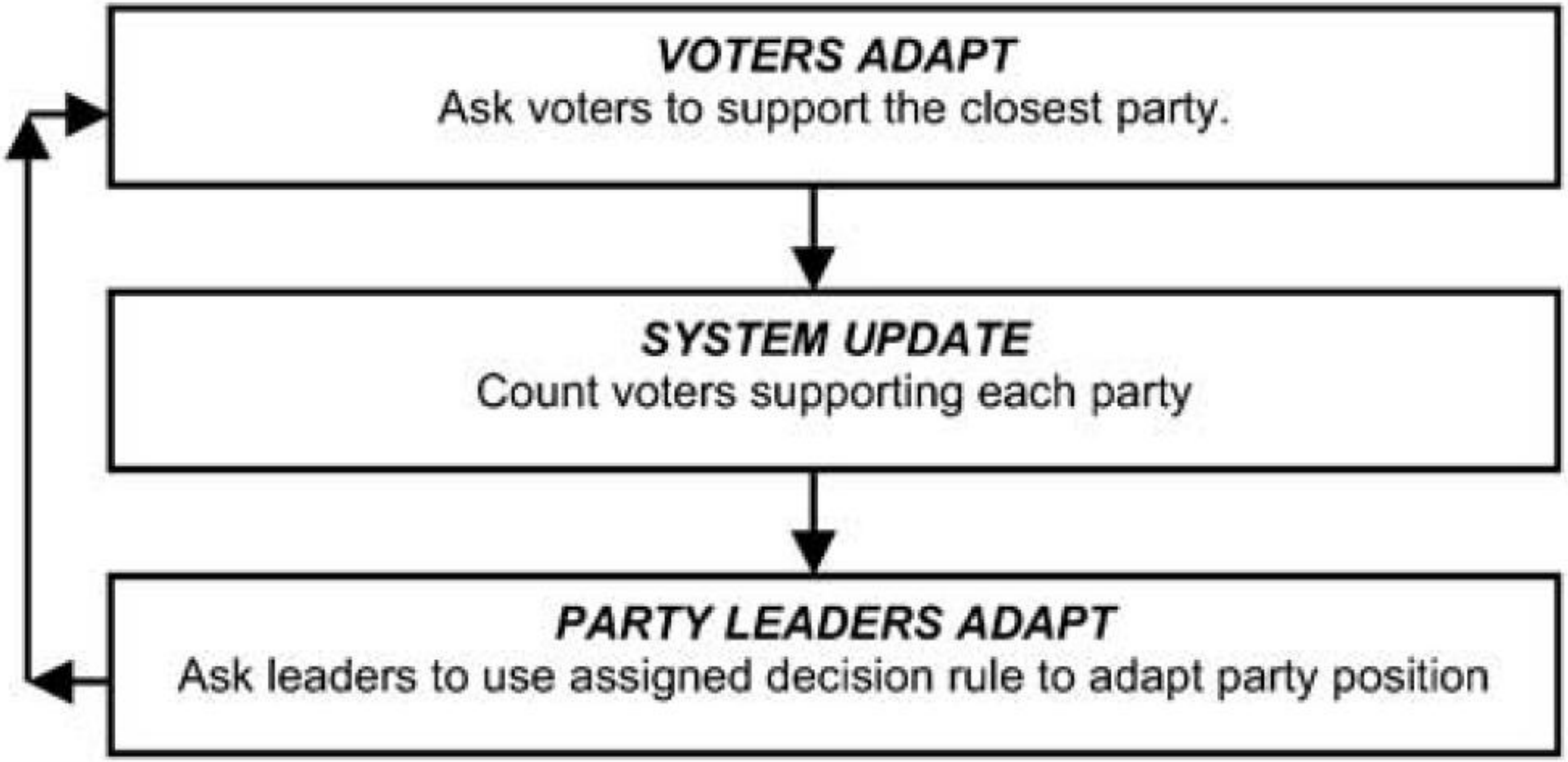
MICHAEL LAVER *New York University*

This paper proposes a model that takes the dynamic agent-based analysis of policy-driven party competition into a multiparty environment. In this, voters continually review party support and switch parties to increase their expectations; parties continually readapt policy positions to the shifting affiliations of voters. Different algorithms for party adaptation are explored, including "Aggregator" (adapt party policy to the ideal policy positions of party supporters), Hunter (repeat policy moves that were rewarded; otherwise make random moves), Predator (move party policy toward the policy position of the largest party), and "Sticker" (never change party policy). Strong trends in the behavior of parties using different methods of adaptation are explored. The model is then applied in a series of experiments to the dynamics of a real party system, described in a published opinion poll time series. This paper reports first steps toward endogenizing key features of the process, including the birth and death of parties, internal party decision rules, and voter ideal points.

Spatial model of voting



Sequence



VOTERS ADAPT

Ask voters to support the closest party.

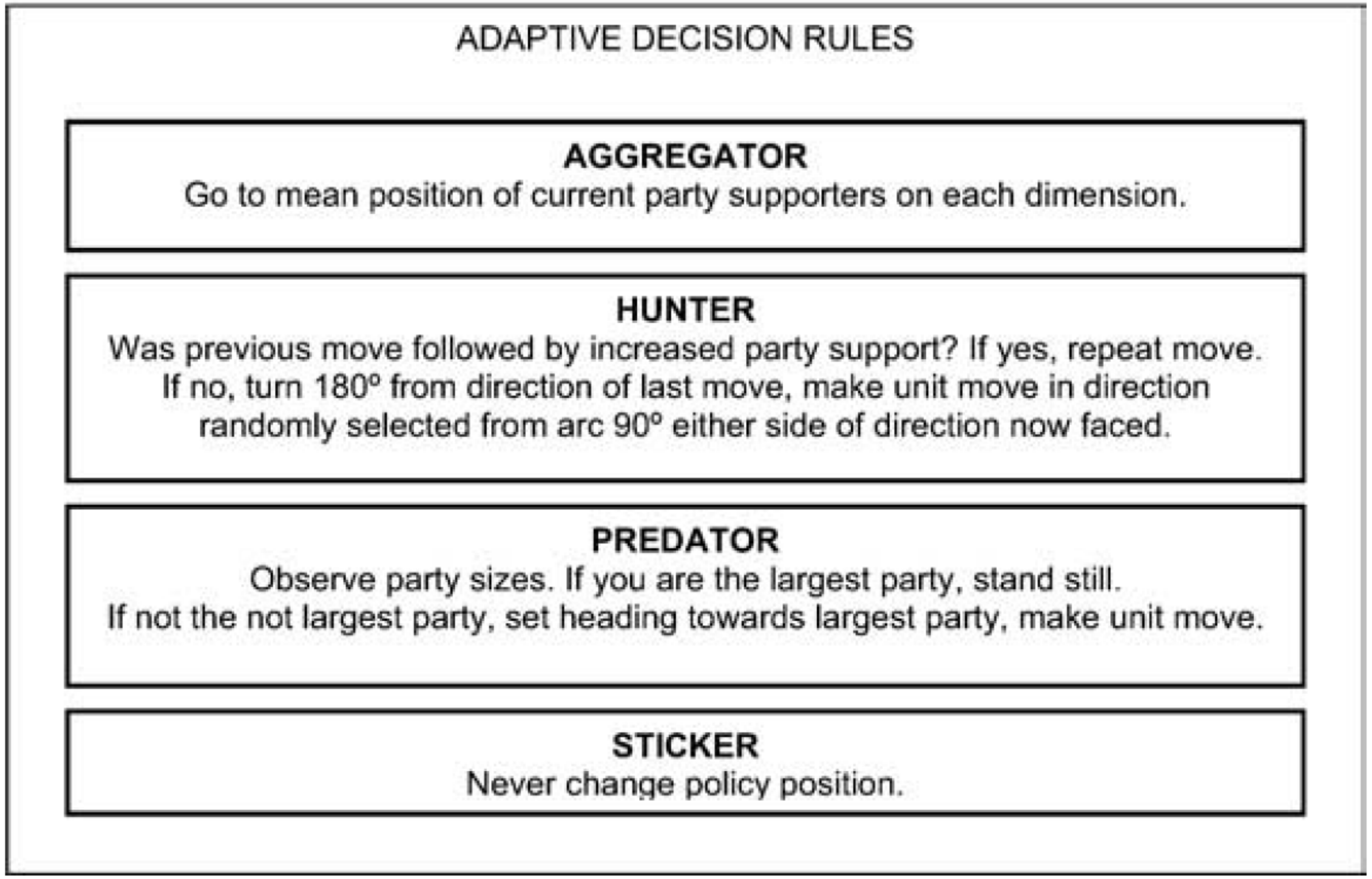
SYSTEM UPDATE

Count voters supporting each party

PARTY LEADERS ADAPT

Ask leaders to use assigned decision rule to adapt party position

Party strategies



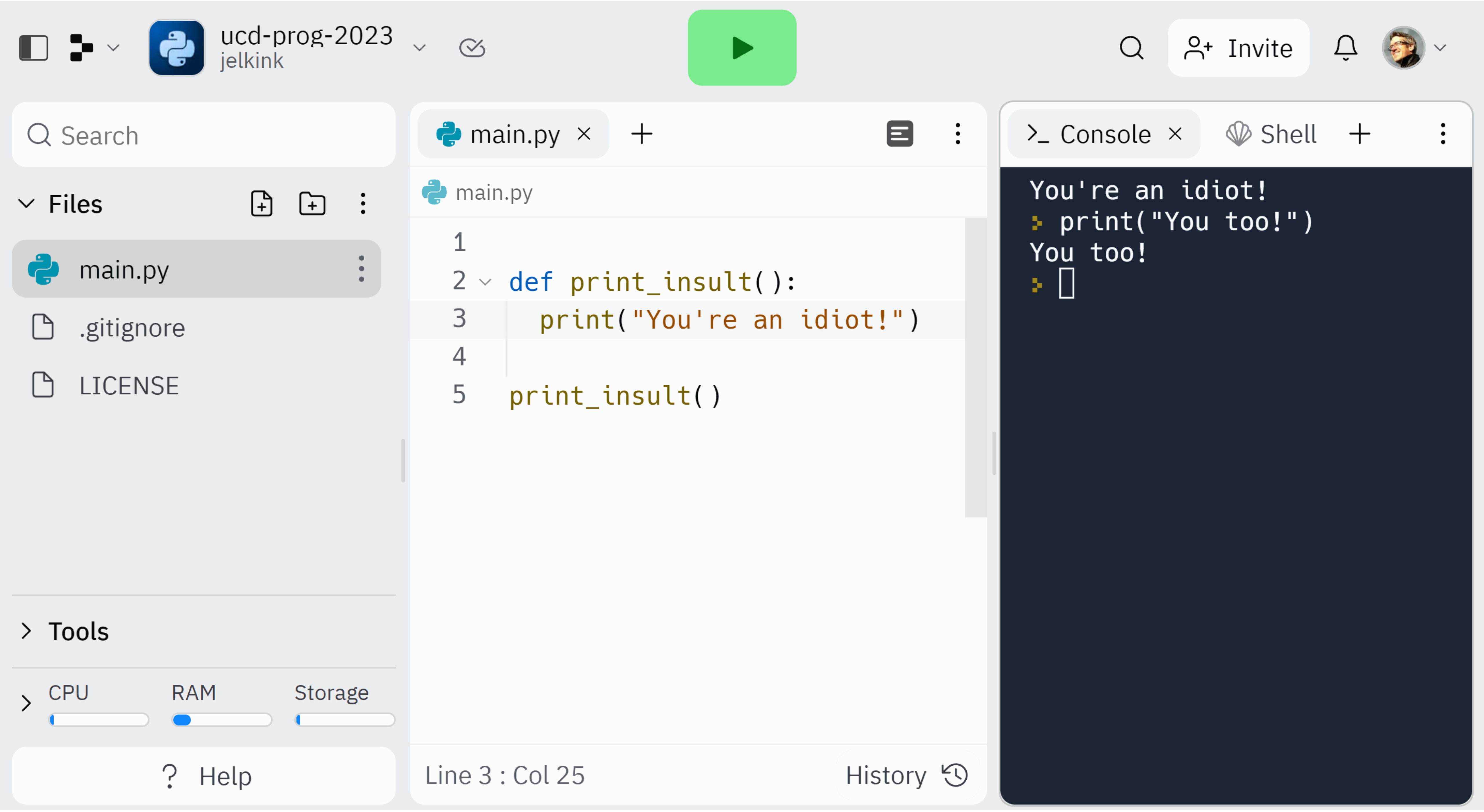
ADAPTIVE DECISION RULES

AGGREGATOR
Go to mean position of current party supporters on each dimension.

HUNTER
Was previous move followed by increased party support? If yes, repeat move.
If no, turn 180° from direction of last move, make unit move in direction
randomly selected from arc 90° either side of direction now faced.

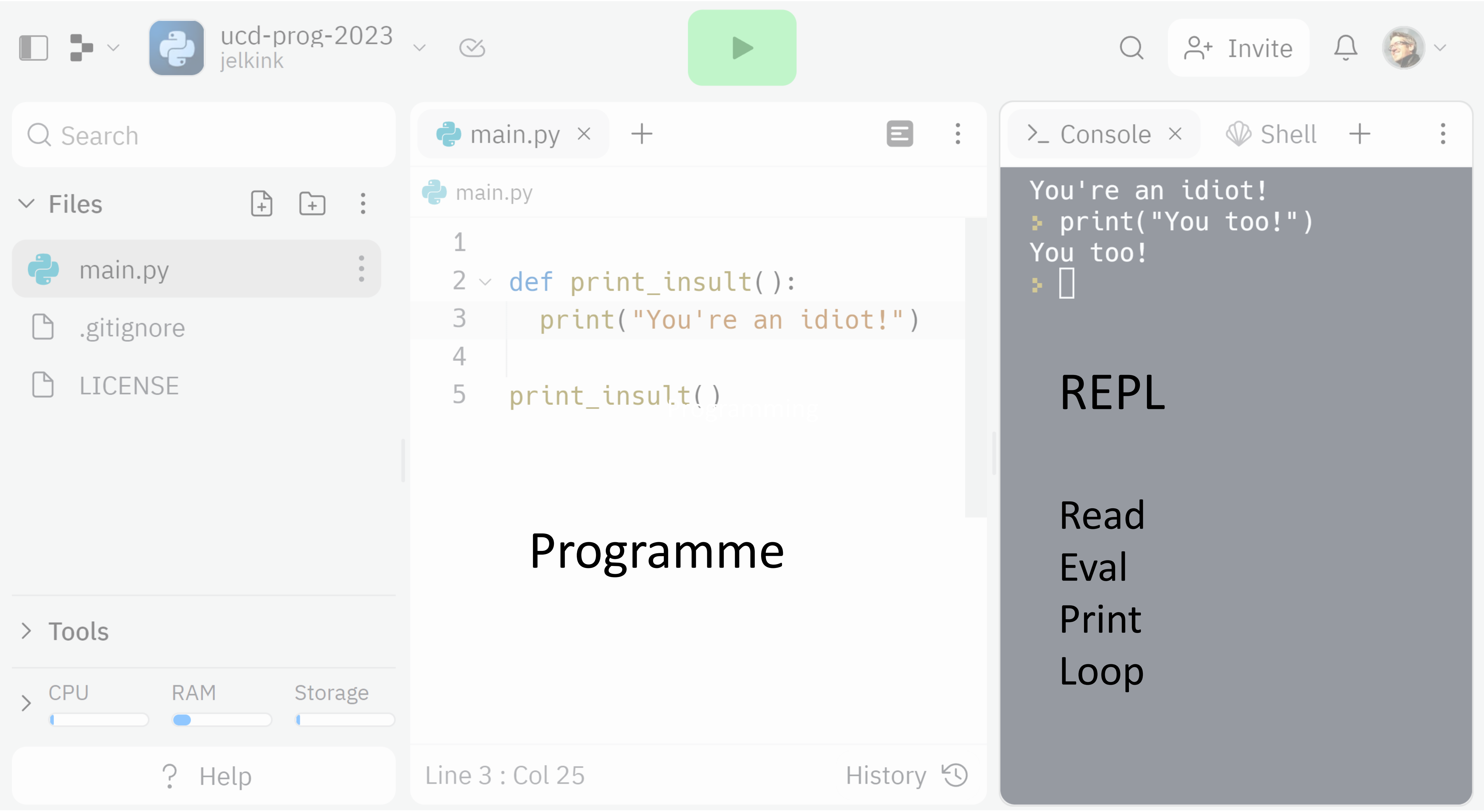
PREDATOR
Observe party sizes. If you are the largest party, stand still.
If not the not largest party, set heading towards largest party, make unit move.

STICKER
Never change policy position.



```python

def print_insult():
  print("You're an idiot!")

print_insult()
```

```
You're an idiot!
> print("You too!")
You too!
> 
```



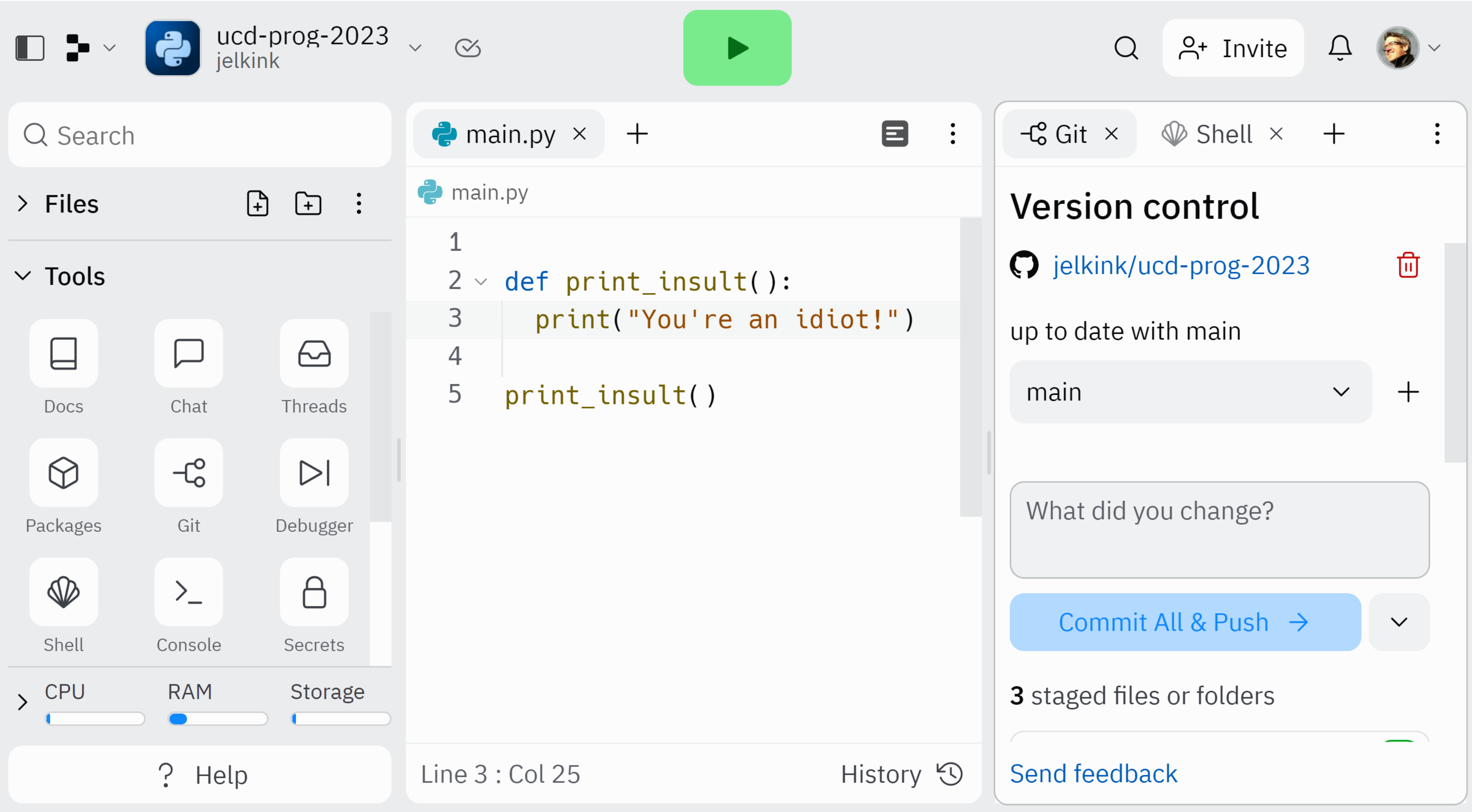
```python
def print_insult():
  print("You're an idiot!")

print_insult()
```

```
You're an idiot!
> print("You too!")
You too!
> 
```

Programme

REPL

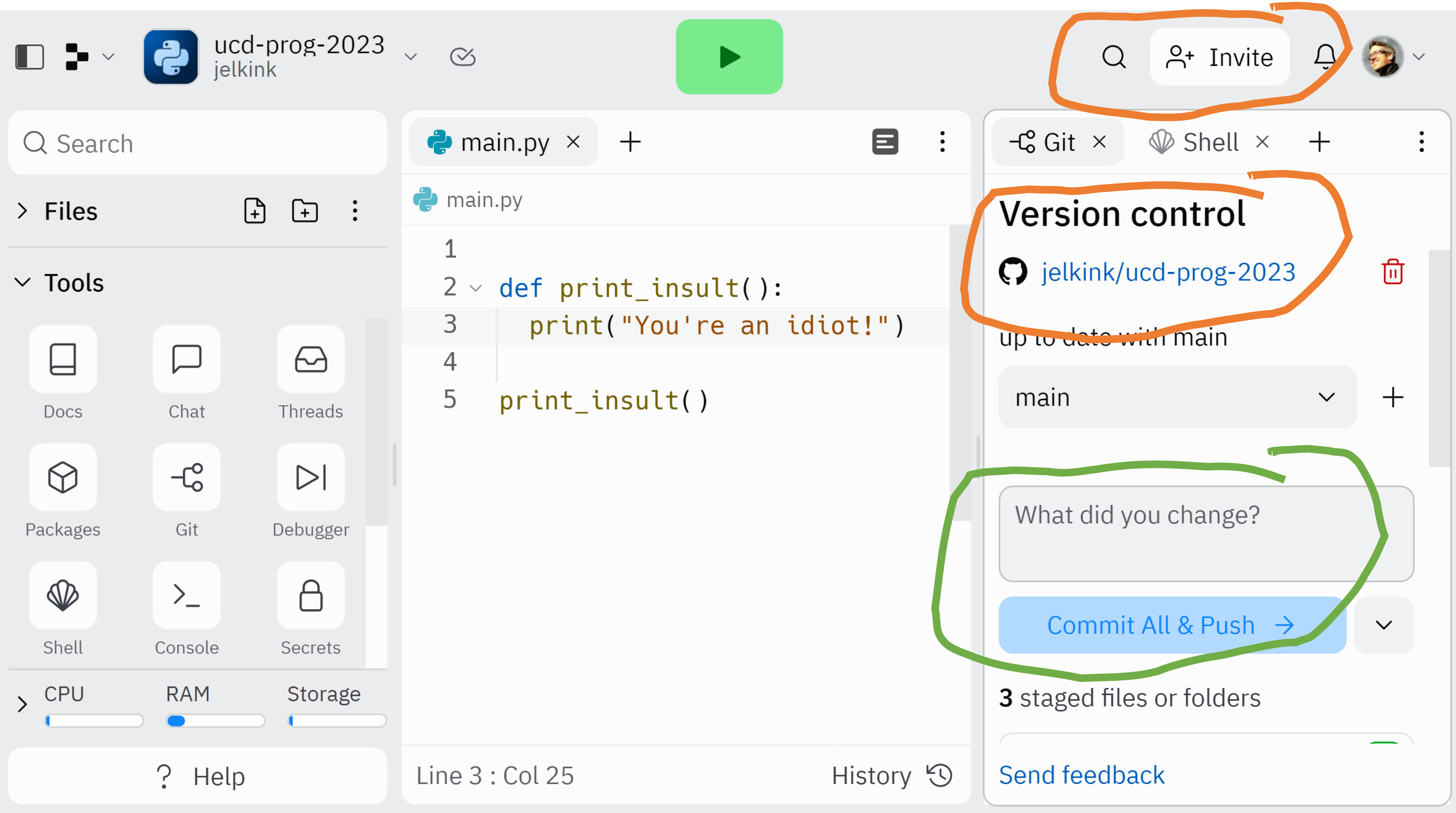
Read
Eval
Print
Loop



```

def print_insult():
  print("You're an idiot!")

print_insult()
```

up to date with main

3 staged files or folders



main.py

```python
def print_insult():
  print("You're an idiot!")

print_insult()
```

Version control

jelkink/ucd-prog-2023

up to date with main

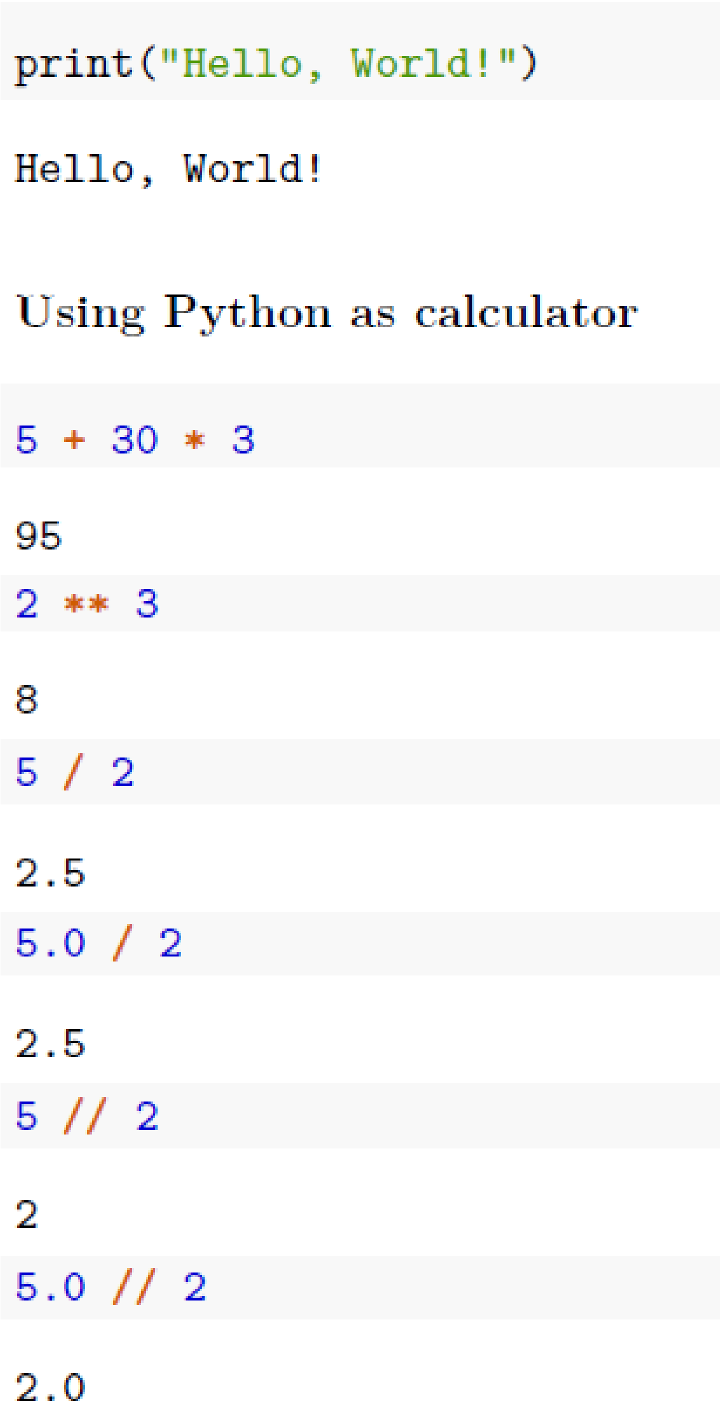
main

What did you change?

Commit All & Push →

3 staged files or folders

Getting used to Python



```
print("Hello, World!")
```

```
Hello, World!
```

Using Python as calculator

```
5 + 30 * 3
```

```
95
```

```
2 ** 3
```

```
8
```

```
5 / 2
```

```
2.5
```

```
5.0 / 2
```

```
2.5
```

```
5 // 2
```

```
2
```

```
5.0 // 2
```

```
2.0
```

Work on Lab 1

Working with your neighbours is a good idea!