Critical Reasoning
Lecture-Seminar 9
Data Science (Part II):
Coding and Python

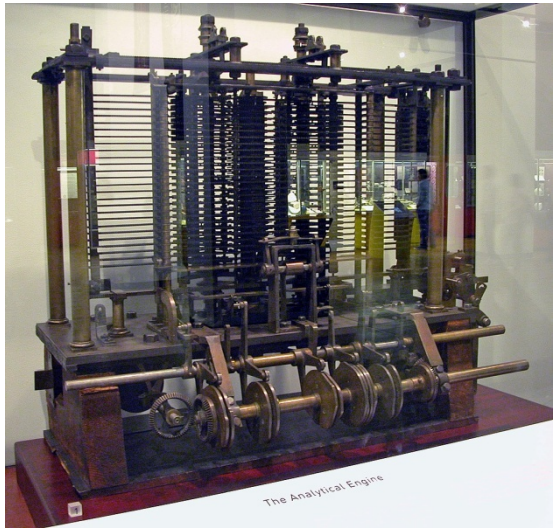
Dr. Ioannis Votsis
NCH, Philosophy Faculty

ioannis.votsis@nchlondon.ac.uk
www.votsis.org

Introduction

What is programming?

- Programming, also known as coding, consists of instructing a computer to perform one or more tasks.
- A program, or algorithm, is a sequence (ordered series) of instructions. When executed, it manipulates data.
- The first-ever program was designed by English mathematician Ada Lovelace (1815-1852).



The variety of programming languages

- To code a computer one first needs a way to communicate with it. That's where programming languages come into play.

- These come in various 'shapes and sizes'. They can be:

  * low-level or high-level
  * more or less like the English language
  * instruction set heavy vs. instruction set light
  * special-purpose vs. general-purpose
  * object-oriented vs. procedural
  * …

- There are currently hundreds, if not thousands, of coding languages out there.

Low-level vs. high-level

- A low-level language is a language whose instructions are identical to/closely resemble the processor's own instructions.

Machine code
0100010100101010101
1110000101010111100

Assembly code
MOV T, 2F
ADD T, E

- By contrast, a high-level language's instructions resemble English and are worlds apart from the processor's own.

Basic
10 LET x = 3
20 PRINT x + 4
30 GOTO 10

COBOL
PROCEDURE DIVISION.
DISPLAY 'Let it be'.
STOP RUN.

Compilers and interpreters

- Since there are many distinct processor architectures, e.g. x86, 6502, ARM, there are many distinct sets of machine code.
- High-level languages provide a more universal basis upon which to instruct any computer.
- Their instructions must be translated into the corresponding machine code. Hence, the need for compilers and interpreters.
- Compilers translate a whole program into machine code. The program can then be executed.
- Interpreters translate and execute a program into machine code, one instruction at a time.

Instruction-heavy vs. instruction-light

- There is a trade-off between the instruction set size of a language and the length of its programs.
- That is, the larger the instruction set of a language, the smaller the size of a program for a given task.
- Low-level languages have a small instruction set while high-level languages have a large one.
- The simplest ever hypothetical machine and coding language is Hao Wang's register machine (1957).
- The machine has only three instructions: Inc, Deb and End. Amazingly, these are sufficient to process any computation!

Instruction-light: The register machine

- "A register machine is an idealized, imaginary (and perfectly possible) computer that consists of nothing but some (finite number of) *registers* and a *processing unit*" (Wang 1957: 7).

- Registers are memory locations where numbers are stored. They have unique addresses, e.g. register 1, register 2, etc.

- The three instructions of the register machine are:

  Inc *n*, *m*: Increase value of register *n* by 1 (up to a certain limit) and go to step *m*.

  Deb *n*, *m*, *p*: If the value of register *n* > 0 then decrease the value of register *n* by 1 and go to step *m*, else go to step *p*.

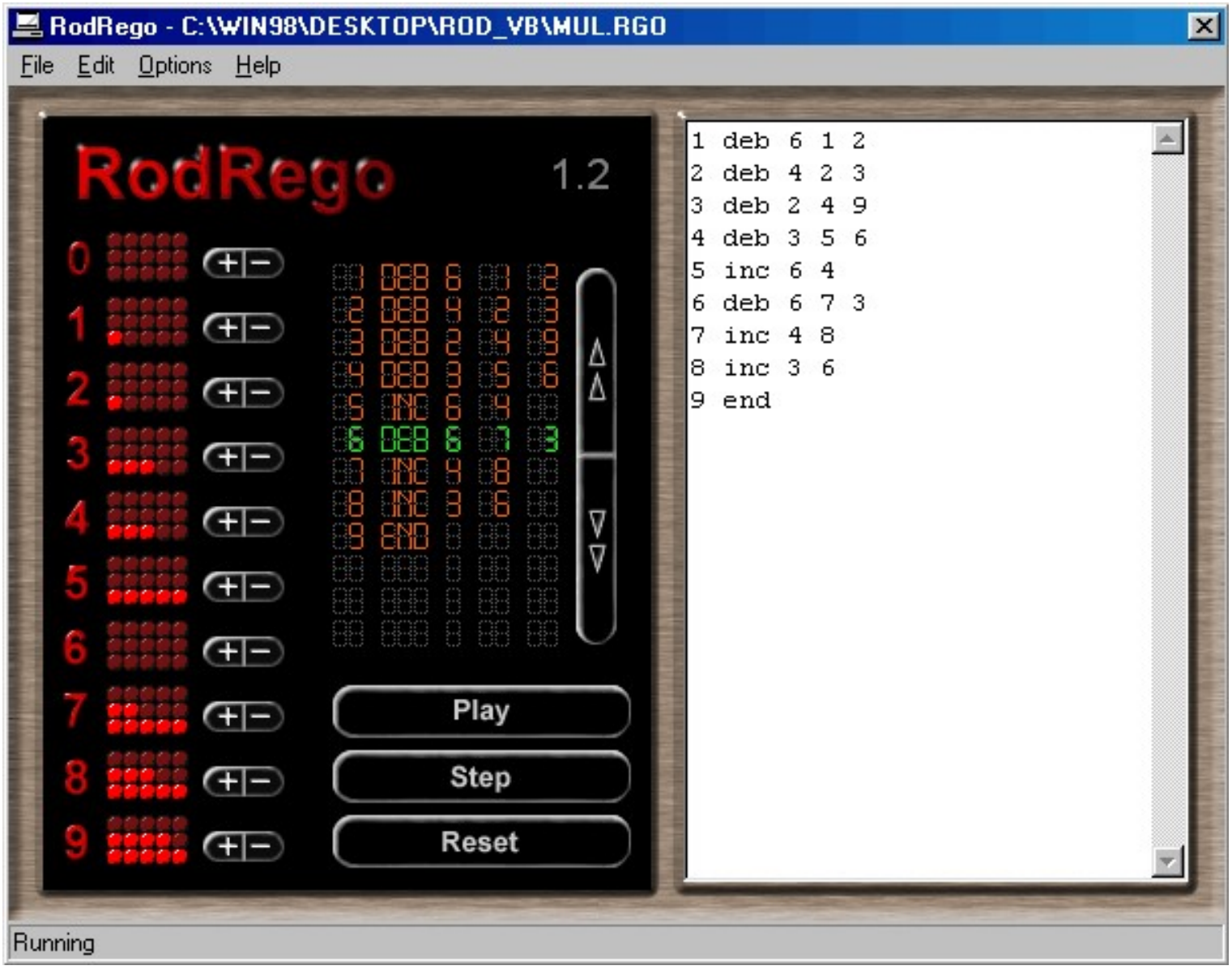
  End: Stop the program.

Hand-simulation anyone?

- In principle, one can simulate these programs by hand:

  “If the way this program works isn’t dead obvious to you yet, get out some cups for registers (pencil a number on each cup, its address) and a pile of pennies (or beans) and “hand simulate” the whole process” (Dennett 2015: p. 12).

Dennett's emulation of a register machine



Special- vs. general-purpose

- A special-purpose language is domain-specific. That is, it can only carry out a restricted set of tasks specific to that domain.

  Examples: HTML, LaTeX, SQL, Mathematica.

- By contrast, a general-purpose language is one that is domain-neutral. It can carry out any computable task.

  Examples: Basic, C#, Objective-C, Python.

- General-purpose languages are Turing-complete. That is, they allow the creation of programs that can emulate any machine.

Control structures

- As the name suggests, these are instructions that control the program's behaviour and flow.

- There are two major types of such instructions:

  Selection structures: Produce decisions and branching and employ logical conditions (a.k.a. Boolean operators) to do so.

  Examples: if-then, if-then-else, elseif (w/and, or, not clauses)

  Repetition structures: Produce loops where the same part of the code runs more than once over distinct values.

  Examples: for, while, do-while, repeat-until, etc.

Selection structures: Example

- The same selection structures are found in most languages.

C#

```
string x = Console.ReadLine();
string y1 = "is a teacher";
string y2 = "is a student";
string z;
if (x == "Ioannis")
{
      z = x + " " + y1;
}
else  z = x + " " + y2;
Console.WriteLine (z);
```

```
Ioannis
Ioannis is a teacher
>
```

Python

```
x = input ()
y1 = 'is a teacher'
y2 = 'is a student'
if x == 'Ioannis':
        z = x + ' ' + y1
else:
        z = x + ' ' + y2
print (z)
```

```
Alex
Alex is a student
>
```

Repetition structures: Example

- The same repetition structures are found in most languages.

C#

```
int z = 0;
for (int x = 0; x < 6; x++)
{
    z = z + x;
    Console.WriteLine ("Z="+z);
}
```

Python

```
z = 0
for x in range(6):
        z = z + x
        print('Z=', z)
```

```
Mono C# compiler version 4.6.2.0
> mcs -out:main.exe main.cs
> mono main.exe
Z=0
Z=1
Z=3
Z=6
Z=10
Z=15
>
```

```
Z= 0
Z= 1
Z= 3
Z= 6
Z= 10
Z= 15
>
```

Python: Some (More) Ideas

The language's vitals

- Python is a general-purpose programming language that comes with both an interpreter and a compiler.

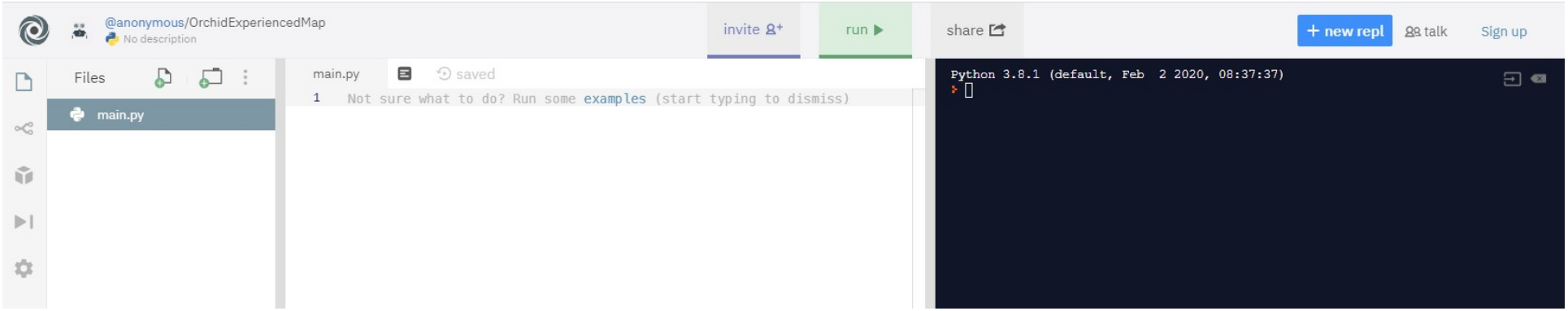
- It is widely used in academia and beyond as it is allows its users to code with easy and parsimony.

  "Python, the fastest-growing major programming language, has risen in the ranks of programming languages in our survey yet again, edging out Java this year and standing as the second most loved language (behind Rust)" (Stack Overflow Developer Survey Results 2019).

- It is also supported by numerous libraries which 'extend its power' through various specialised functions.

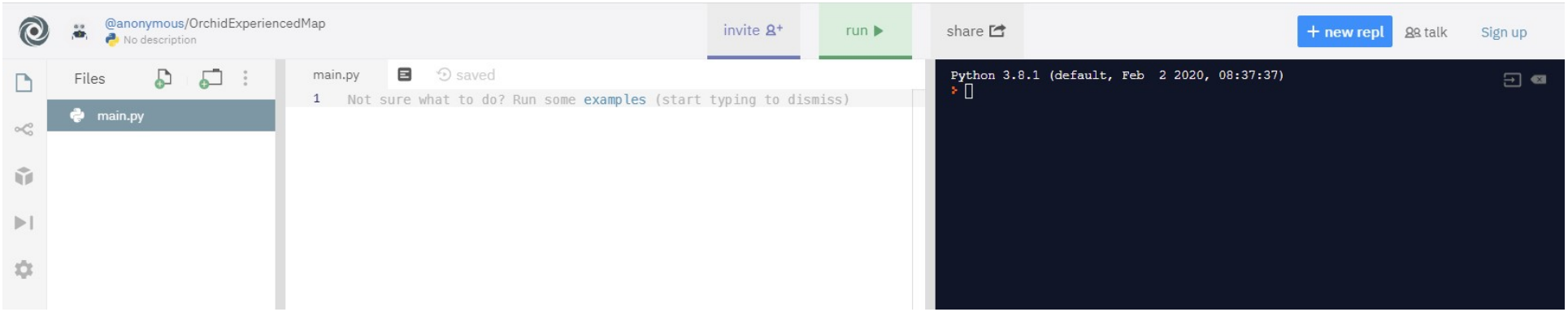
Python online

- For the purposes of this lecture, it doesn't make sense to install Python on your system (laptop, tablet, etc.).
- Instead, you can just go online to: https://repl.it/languages/
- Find Python from the list of languages and click on it. You should be able to see the following screen.



Python online: Diving in

File Manager **Script Window/Mode** **Interactive Window/Mode**



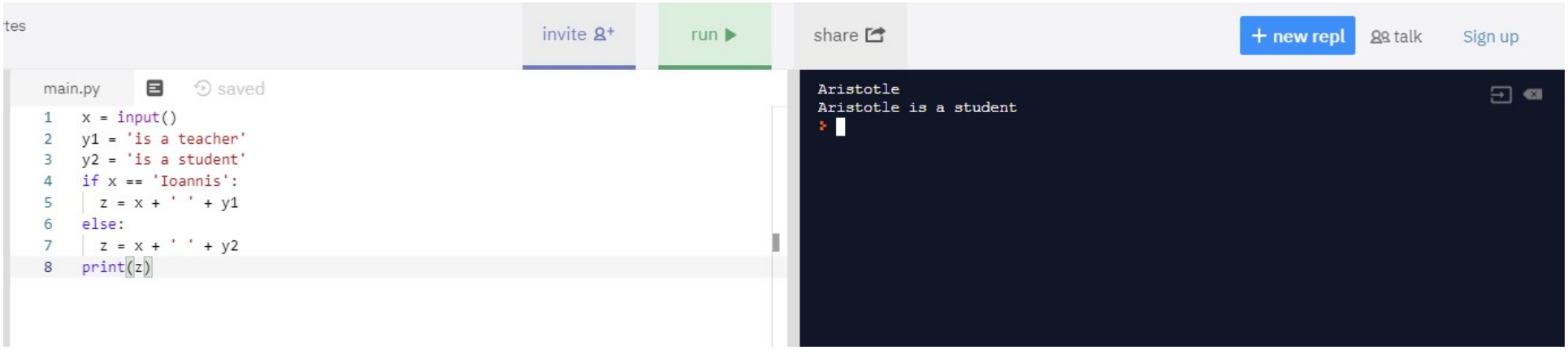
- The interactive mode allows us to execute instructions one-by-one. It's useful in finding out how snippets of code work.

  NB: This mode is also known as the *shell*.

- The interactive mode doesn't save our code. For that we need the script mode. Here all the code is compiled and executed.

Executing code

- Suppose we write some of the aforementioned code in the script window. Press the **run** button to execute it!



- The interactive window will then show the **result** (output and any prompts for input).

Functions in Python

- Some functions are built-into the programming language.

Input:
```
print(‘I didn’t, did I?’)
abs(-5)
type(19)
```

Output:
```
I didn’t, did I?
5
<class ‘int’>
```

- Others are user-defined. These can be as complex as we like and are constructed on the basis of the built-in functions.

Input:
```
def tentothe(x):
    y = 10**x
    print('Ten to the', x, '=', y)
tentothe(3)
```

Output:
```
Ten to the 3 = 1000
```

Libraries and modules

- Most programming languages nowadays make use of libraries.

  NB: Libraries are made up of modules.

- Roughly speaking (for most intents and purposes), modules and libraries offer supplementary collections of functions.

  Examples: Statistical, Machine-learning, Image Processing, etc.

- Such libraries make a programmer's life considerably easier as they do not need to reinvent the wheel.

- If unavailable – as was the case when I was coding The Logic Calculator in C# – projects slow down or even grind to a halt.

Libraries and modules in Python

- Python is the language of choice by the majority of scientists (both social and natural).

- Among the most commonly used scientific libraries are: Math, NumPy, SciPy, Matplotlib, Keras and TensorFlow.

  NB: Needless to say there is overlap between some of these.

- To gain access to a library/module in Python, we first need to import it using the namesake instruction.

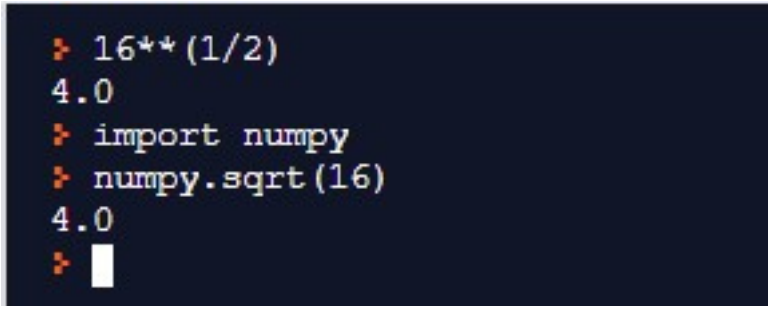
- Type ‘import numpy’ into either the interactive or the script window. We are now ready to use supplementary functions.

Functions in NumPy: Sqrt(x)

- Recall that functions in libraries/modules are written on the basis of Python's own built-in functions.
- That means that you can perform the same tasks using those built-in functions. Consider the task of getting square roots.

Python without NumPy
16**(1/2)

Python with NumPy
numpy.sqrt(16)



NB: In script mode, we need to add a print instruction to see any output.

- Aside from being mnemonic, *this* NumPy instruction is not even shorter to write. This changes with more complex tasks.

Functions in NumPy: Max(x)

- Type the following code into the script window:

```
import numpy as np
age = [16, 27, 82, 31, 57]
x = np.max(age)
print ('The oldest person is', x, 'years old.')
```

NB: The ‘as np’ clause allows us to shorten expressions using NumPy from numpy.max(x) to np.max(x).

- NumPy function max(x) returns the largest value from an array x of values. An array is a set of data (usually of the same type).

```
The oldest person is 82 years old.
>
```

Saving your work

- You can use any simple text editor, e.g. Notepad in Windows, to write and edit your code.
- To make the file Python-readable just save it with a '.py' suffix, e.g. if you name the file 'one', you should save it as 'one.py'.
- I will upload all the code presented in this lecture on Moodle in the coming days.
- Finally, if you want Python on your computer, just go to www.python.org and follow the installation instructions.

The End