

# Data Structures & Programmatic Thinking

## Introduction

Pepe García

# About

The Professor

Pepe García

# About

## The Professor

Pepe García

[jgarciah@faculty.ie.edu](mailto:jgarciah@faculty.ie.edu)

# About

## The Professor

Pepe García

[jgarciah@faculty.ie.edu](mailto:jgarciah@faculty.ie.edu)

Lead Engineer @ 47deg

# About

## The Professor

Pepe García

[jgarciah@faculty.ie.edu](mailto:jgarciah@faculty.ie.edu)

Lead Engineer @ 47deg

Ask me anything

# About

## The Course

- 11 sync sessions

# About

## The Course

- 11 sync sessions
- 3 async sessions

# About

## The Course

- 11 sync sessions
- 3 async sessions
- 2 assignments (1 individual, 1 group)



## The Course

- 11 sync sessions
- 3 async sessions
- 2 assignments (1 individual, 1 group)
- 1 final exam

## Grading

Criteria	Score %
Final Exam	20 %
Individual Work	40 %
Workgroups	20 %
Class Participation	20 %

## Grading

The grading for this course will either be **Non Graded Satisfactory** or **Non Graded Unsatisfactory**.

If you get 50% or more in the overall score, you get **NGS**, and **NGU** otherwise.

## Participation

Please, raise your hand at any point in class if you want to ask something, make an useful comment, or answer a question. (if remote, use Zoom's raise hand feature, so that it's easier to track it)

# Learning Objectives

In this course we will build the fundamentals for the rest of the courses in the masters that rely on programming. We will:

# Learning Objectives

In this course we will build the fundamentals for the rest of the courses in the masters that rely on programming. We will:

- Learn What's programming

# Learning Objectives

In this course we will build the fundamentals for the rest of the courses in the masters that rely on programming. We will:

- Learn What's programming
- Understand how computers execute programs

# Learning Objectives

In this course we will build the fundamentals for the rest of the courses in the masters that rely on programming. We will:

- Learn What's programming
- Understand how computers execute programs
- Learn the basics of Python



# Plan for this session

- Learn about software

# Plan for this session

- Learn about software
- Understand what are algorithms and data structures

# Plan for this session

- Learn about software
- Understand what are algorithms and data structures
- Install Anaconda

# Language

Throughout this course we will use Python as our programming language, but there are many more!

# Language

There are several ways for categorizing programming languages.

## Language classification

Language	Paradigm	Execution	Purpose
Python	imperative	interpreted	general
Java	object oriented	compiled	general
Javascript	imperative	interpreted	general
Haskell	functional	compiled	general
SQL	declarative	interpreted	specific
HTML	declarative	interpreted	specific

# Language

There are several ways for categorizing programming languages.

## Language classification

Language	Paradigm	Execution	Purpose
<b>Python</b>	<b>imperative</b>	<b>interpreted</b>	<b>general</b>
Java	object oriented	compiled	general
Javascript	imperative	interpreted	general
Haskell	functional	compiled	general
SQL	declarative	interpreted	specific
HTML	declarative	interpreted	specific

# Language

## Python

Python is one of the most used languages right now. Its applications range from Data Science to Web servers

# How do we write code?

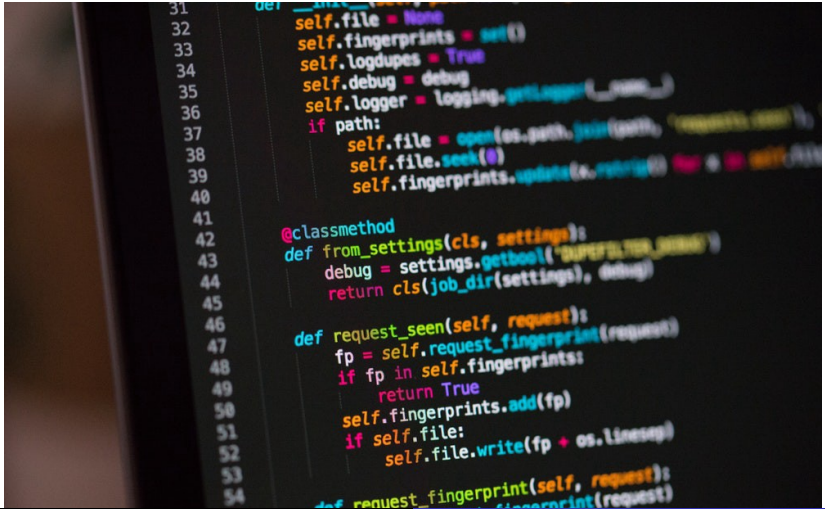


# How do we write code?

Coding is basically putting words together following a programming language specification.

# How do we write code?

We can put these words directly in a text file and then execute it as a program.



```
31     def __init__(self, path):
32         self.file = None
33         self.fingerprints = set()
34         self.logdupes = True
35         self.debug = debug
36         self.logger = logging.getLogger(__name__)
37         if path:
38             self.file = open(os.path.join(path, 'requests.json'),
39                             'w')
40             self.file.seek(0)
41             self.fingerprints.update(self._get_fingerprints())
42
43     @classmethod
44     def from_settings(cls, settings):
45         debug = settings.getbool('SUPERFINGER_DEBUG')
46         return cls(job_dir(settings), debug)
47
48     def request_seen(self, request):
49         fp = self.request_fingerprint(request)
50         if fp in self.fingerprints:
51             return True
52         self.fingerprints.add(fp)
53         if self.file:
54             self.file.write(fp + os.linesep)
```

# How do we write code?

Or we can feed these words directly into the programming language console.

## Python console

Let's see how do code looks in the console!

# Install Anaconda platform

Now we will install the Anaconda platform in our computers.

- 1 go to <https://www.anaconda.com/products/individual>
- 2 Go to the bottom of the page, to the **Anaconda Installers** section, and download the graphical installer for the 3.7 version **for your operating system**.
- 3 In the installer, when you're given the option to install the PyCharm IDE, or Visual Studio Code, you can ignore it, we're not going to use it.

## Checkpoint

Anybody is lost or has problems installing the software?

# Programs

# Programs

What is a program?

A program is a piece of software with a specific task.



# Programs

## What is a program?

A program is a piece of software with a specific task. This task can be something **big**, like handling a nuclear reactor, or something **small** like Ctrl-c/Ctrl-v.

# Programs

## What is a program?

A program is a piece of software with a specific task.

This task can be something **big**, like handling a nuclear reactor, or something **small** like Ctrl-c/Ctrl-v.

There are two main components of programs, **algorithms** & **data structures**.

# Algorithms

# Algorithms

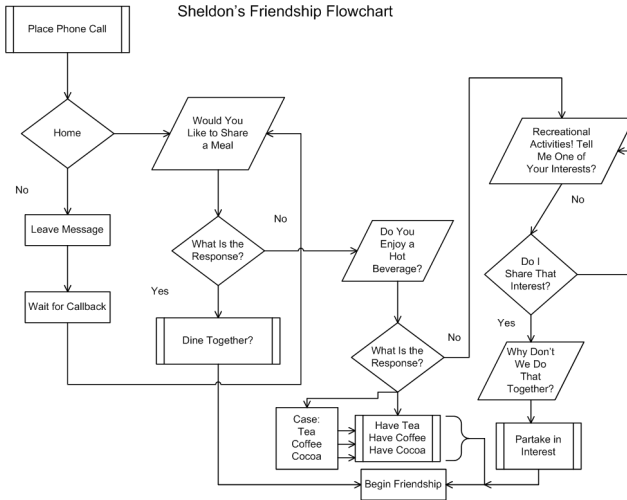
What is an algorithm?

# Algorithms

What is an algorithm?

An algorithm is a sequence of steps that guide the computer in how to solve a problem

# Algorithms



link to the video

# Algorithms

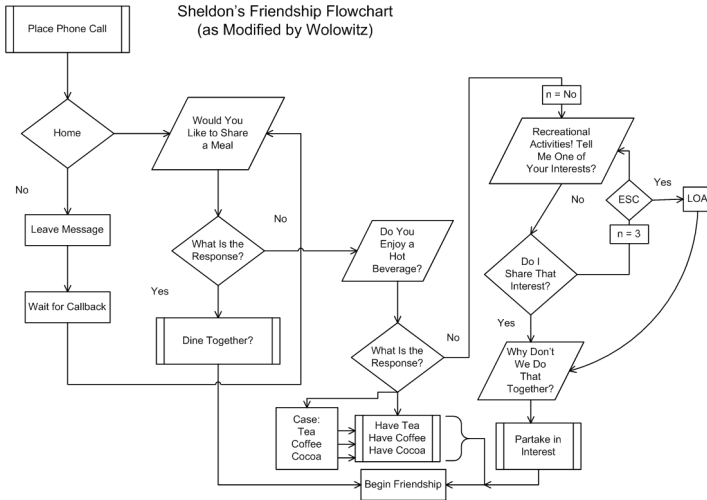
What's wrong with this algorithm? why did Wolowitz need to fix it?

What's wrong with this algorithm? why did Wolowitz need to fix it?

There was a **bug**, an infinite loop



# Algorithms



What other cases of bugs do we know?

Business

## Knight Shows How to Lose \$440 Million in 30 Minutes

By [Matthew Philips](#)

August 2, 2012, 11:10 PM GMT+1



<https://www.bloomberg.com/news/articles/2012-08-02/knight-shows-how-to-lose-440-million-in-30-minutes>

Spyder is the editor we will use in this course for programming in python, let's investigate it for a bit!

# Recap

# Recap

- We'll use Python for learning programming in this course.

# Recap

- We'll use Python for learning programming in this course.
- Algorithms, like cooking recipes, will guide our program to perform what we want.

# Recap

- We'll use Python for learning programming in this course.
- Algorithms, like cooking recipes, will guide our program to perform what we want.
- We'll use the Spyder editor to program in Python



# Recommended reading

**What Is Code** is a great essay by Paul Ford. (it's a bit long, you don't need to read it for tomorrow)

<https://www.bloomberg.com/graphics/2015-paul-ford-what-is-code/>