Hello, world!

Spandan Madan

Introductions

What do you work on?

How would you like to use programming?

Motivations mentioned by you all...

- Data Analysis
- Building a website
- Machine Learning
- Building an app
- ….

My Motivation

- Learning how to communicate specialized ideas to people who come very different contexts, skill sets as me.

- Exposure to problems faced by people from different professions - and solving them = Fun

- Enabling my friends to do things by giving them a new (hopefully useful) skill set.

The anatomy of our lectures

Every Lecture:

- A story
- A short presentation by me.
- 2 Flipped classroom presentations (5-10 mins each).
- Let’s build something?
- Computational Thinking : What would be the steps involved in doing X? (interactive)

What will we learn in this course?

- We will learn to learn programming.

- We will build somethings together.

- We will all build one thing for our own selves.

First Lecture

Popular courses on YouTube:

- “You will learn to store 2 values in variables, and swap them”.

We will be able to check the urban legend:

Nicolas Cage appearing in more movies leads to more people drowning in swimming pools.

What is Computer Science

Wrong question….

- Let’s try to relate it to *your* context. The more important question is what can computer science do for you. And what gives it the power to do so.

Making use of *free labor* using an *algorithm*

Unless you count the electricity you’re spending….

Free Labor

- Tiny little workers, not very intelligent.

- Hard working, but you must learn to speak their tongue!
- The instructions are mostly very, very boring:
    - Store this information
    - Give me that information
    - Show me that information
    - Add/subtract/multiply/divide these two numbers

Like Lego Blocks

- Every piece of digital technology!
- It’s fun to see how the pieces fit.
- It’s even more fun to build something yourself :)

Algorithm - the process of putting lego blocks together

Breaking down what you want into sub-tasks which are:

- Simpler
- Reusable for other tasks
- Get the job done without errors (and hopefully fast).

We instruct our minions to do these sub-tasks. Thus, an algorithm is a step of instructions which get the computer to do something we desire.

For ex: the algorithm of building one tower using lego blocks can be reused over and over again in the previous castle.

What is each lego block though?

Depends who you ask (because things work at multiple levels):

- Electrical engineer : transistors
- Hard core programmers : circuits in
- Computer Scientists : blocks of logic
- The python programmer : “objects”.

Objects - the glue of (most) programming

- Objects - Suppose I told you I have a pet dog.
- Now what are some questions you can ask about the dog?
- Attributes like:
  - Name?
  - Fur color?
  - Male/Female?
- Actions like:
  - Can I play catch with the dog?

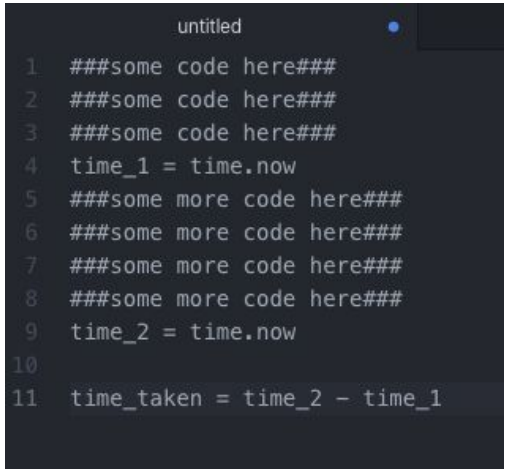
A more abstract example of an object

Object : A thing which has some attributes and lets you do certain operations on it.

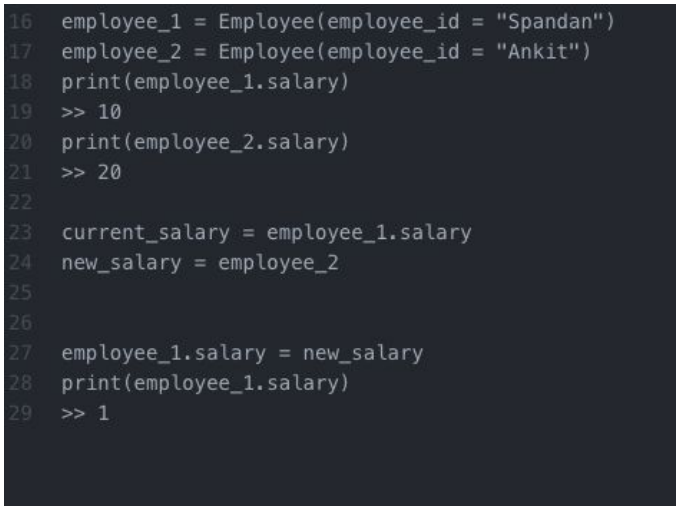
For ex:

- Line object (geometric example):
  - Attributes: start point, end point.
  - Operations: change these points to modify the object.
- Employee object:
  - Attributes: Salary, role in company.
  - Operations: Assign to a task in company, or change salary.
- Time object:
  - Attributes: get current time
  - Operations: Get time difference between two time objects.

Examples (the anatomy of python code)



Employee_object, time_1, time_2 : objects
time_1, time_2 : variables
Employee() : Class
print(): statement
employee_id = “spandan” : Argument



Employee_object, time_1, time_2 : objects
time_1, time_2 : variables
Employee() : Class
print(): statement
employee_id = “spandan” : Argument

Recap

Objects: A thing which have some attributes, and certain operations we can perform on it. (time.now, time_1 - time_2)

Variables: A container to store values

Statements: An instruction to the computer

Arguments: relevant information given in brackets

Class: helps create objects of a certain *kind.* Example: employee_1, employee_2. The class specifies what attributes and operations are associated with the object

From variables to data structures

What can I store in variables?

Well, what would you like to store?

- Numbers
- Names (string)
- employee_IDs (string)
- List of names?

  Ex: “all_employees” variable?

  Useful operation for this object: Check if you have an employee with a name Tim.

Data structures - Storing the info you want to work with

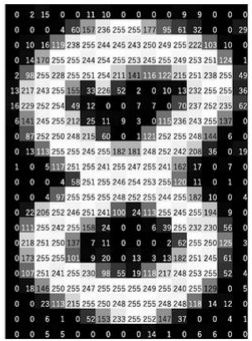
- Lists: [1,2,3] or [‘hi’,‘bye’]
- Dictionary: Key-Value pairs.
  - Ex: Employee_name_to_salary
    - Attribute: look up salary
    - Modify: modify salary

Confusing, but important : In built vs declared objects - everything is an object!

Dictionaries, lists are also objects. Though we don’t need to call a class to create objects of this kind (For ease).

What if I don’t want to work with “information”?

- Sadly, everything is information, or data.
- At the base, computer represents everything numerically.
- Strings, numbers, every kind of information gets stored as numbers.
- Foreshadowing : everything is binary, why? Transistors can only be on or off. Passing current through them you can check if they’re on or off, i.e. are they 0 or 1.

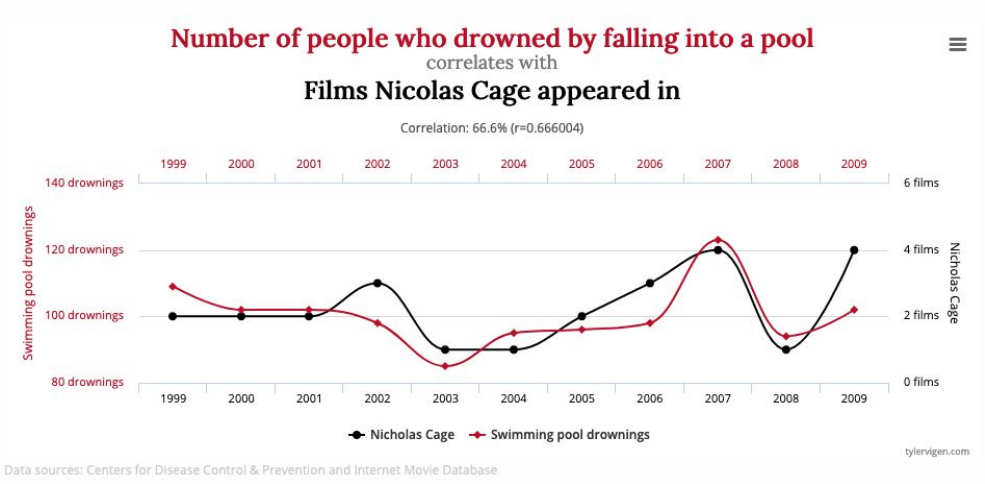


The programmer’s mindset

So, for any task you are interested in try to think of the following:

- What information do I need to use for the task? How would I need to modify it?
- What object will I need to store this information?
- What kind of operations must that object have, to allow me to perform?
- How will different objects interact with each other?

Let’s build something?

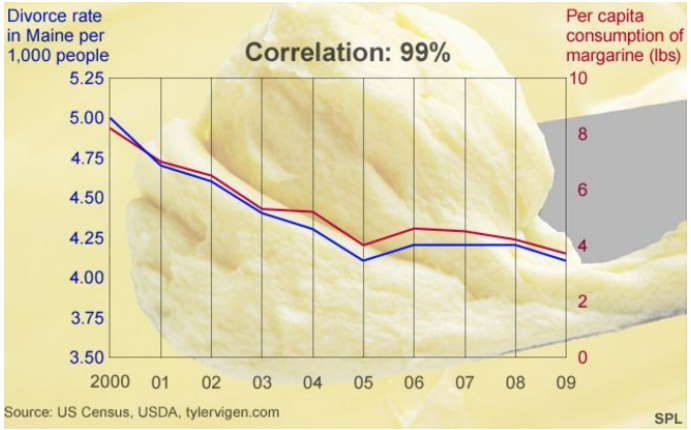


BBC Article!

https://www.bbc.com/news/magazine-27537142

Findings include:

- Margarine leads to divorces
- Pirate shortage caused global warming
- Facebook is causing the greek debt crises



Assignment : to use the tools today to plot COVID-19 growth in countries

- Pulling content from Github.
- Uploading it to colab
- Read file and plot the numbers.

Learning objects covered: modules, classes, objects, lists, file handlers, io, plotting, and more.

Getting set up - Github and Colab

- Where do I practice my code? https://colab.research.google.com

- Where's everything stored? https://github.com/Spandan-Madan/learning_to_speak_python

- Will I get future emails : Not much, everything will be on github. Also allows those who are not interested in continuing to exit without being bombarded by emails :)
- Lectures will be on youtube!

Thank you!

- Hope you enjoyed the lecture :)