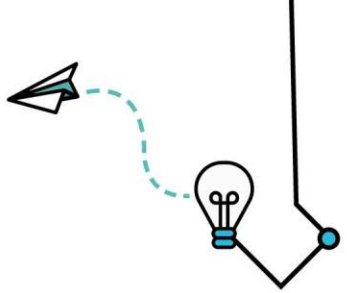
LINKIT

BUILDING IT TOGETHER

PYTHON – DAY 1

ANIS BOUDIH

NAZLI ANDER



WELCOME

Who are we?

BUILDING IT TOGETHER

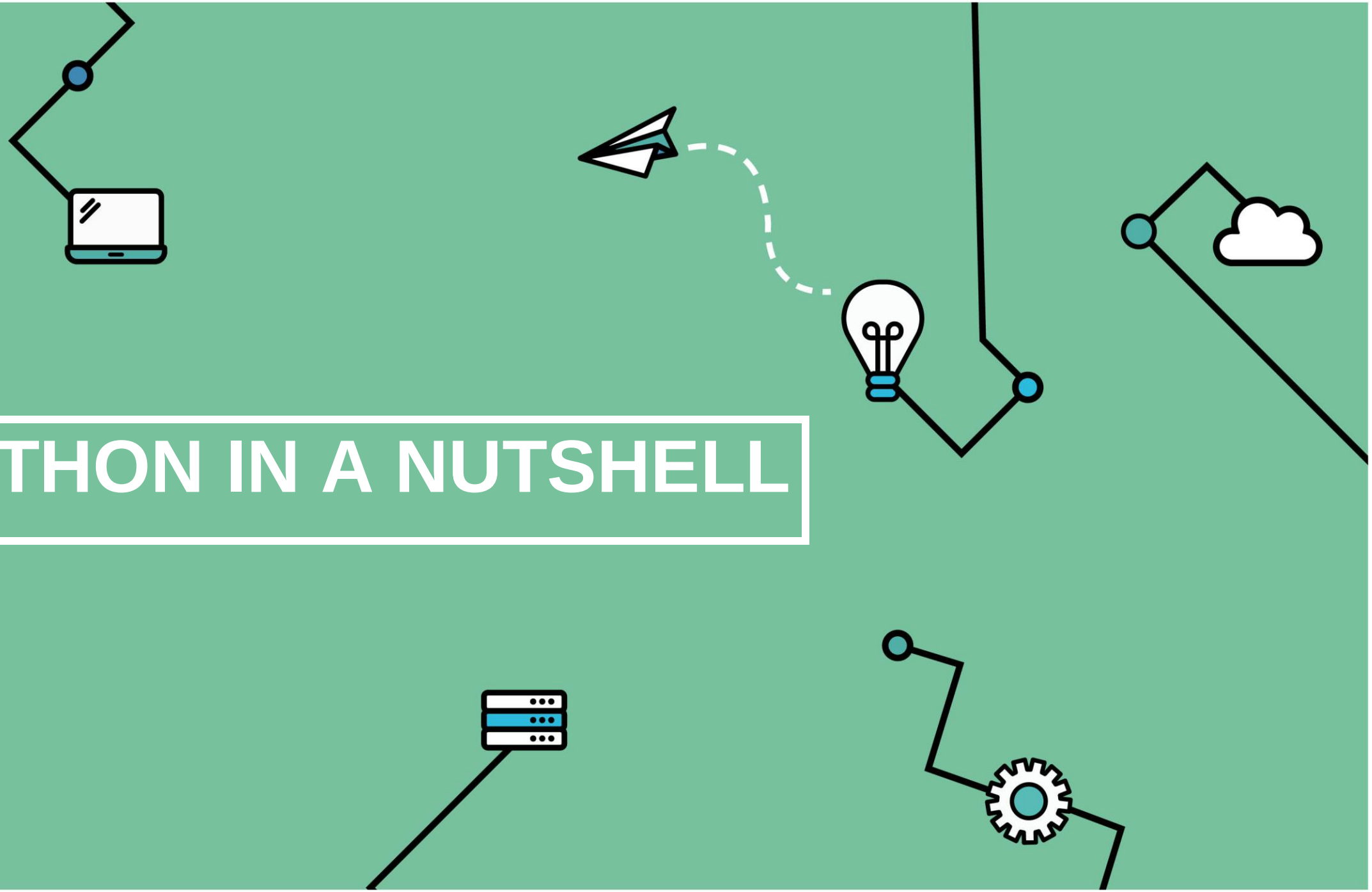
GET TO KNOW YOU

- Your name
- Your background (education, professionally and nationality)
- Your proficiency with programming languages and more specifically with python
- Why did you choose for the bootcamp?

AGENDA FOR TODAY

- Intro to Python in +/- 5 minutes
- Basic concepts in Python such as:
  - Data types
  - Naming conventions
  - Loops
  - Functions
  - Data structures
- Hands-on exercise: Exploring Gutenberg.org with Python

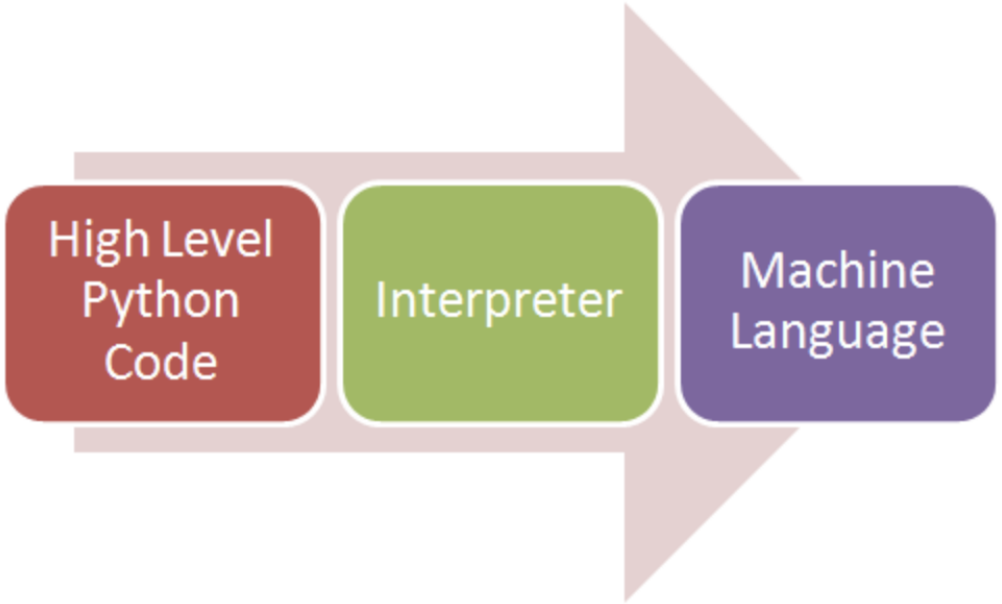
PYTHON IN A NUTSHELL



WHAT IS PYTHON?

- High-level programming language
- Interpreted language
- Readable language :)

INTERPRETED LANGUAGE



THE READABILITY...

"Hello, World"

- C

```
#include <stdio.h>

int main(int argc, char ** argv)
{
   printf("Hello, World!\n");
}
```

- Java

```
public class Hello
{
   public static void main(String argv[])
   {
      System.out.println("Hello, World!");
   }
}
```

- now in Python

```
print "Hello, World!"
```

WHY PYTHON?

- Easy to code and to understand
- It's suitable for both OO and functional programming
- Rapid development of applications
- Large number of packages available
- Large community and books available for support
- All type of applications can be implemented in Python

WHY NOT PYTHON?

- Python is slower than C++, C#, Java
- White-space and indenting constraint
- Not suitable for low-level system and hardware interaction
- Dynamic typing

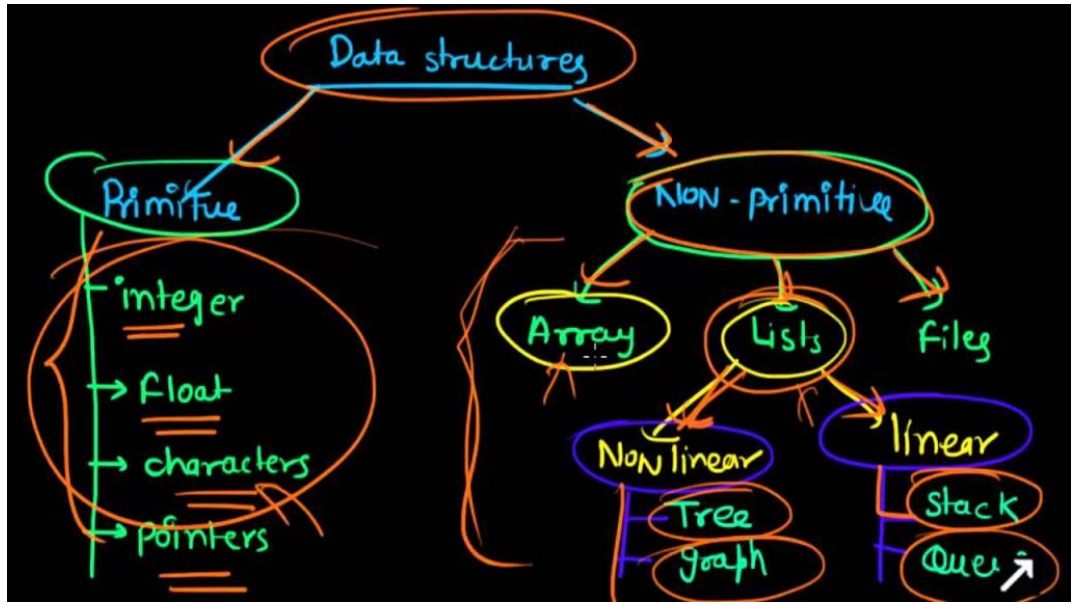
TIME TO LEARN PYTHON!

PRIMITIVE DATA STRUCTURES

- Simplest forms of representing data, hence *primitive*
    - Integer
    - Float
    - String
    - Boolean

NON-PRIMITIVE DATA STRUCTURES

- How to store, insert, delete or access data efficiently led to the development of data structures.
- Other words: data structures provide a way of organizing and storing data so that it can be accessed and modified efficiently
- Important concept in Computer Science.
- Four major data structures used in Python: **List, Tuples, Dictionaries, Sets**



"non-primitive structures are designed to organize and manage sets of primitive data"

INTEGER

- Just as in mathematics, an integer is just a whole number.
- Can be a positive or negative value.
- Can be zero.
- If you want to convert a float or string to an integer, you can use the ***int()*** function; will return the value between the parentheses as an integer (rounding down if necessary).

FLOAT

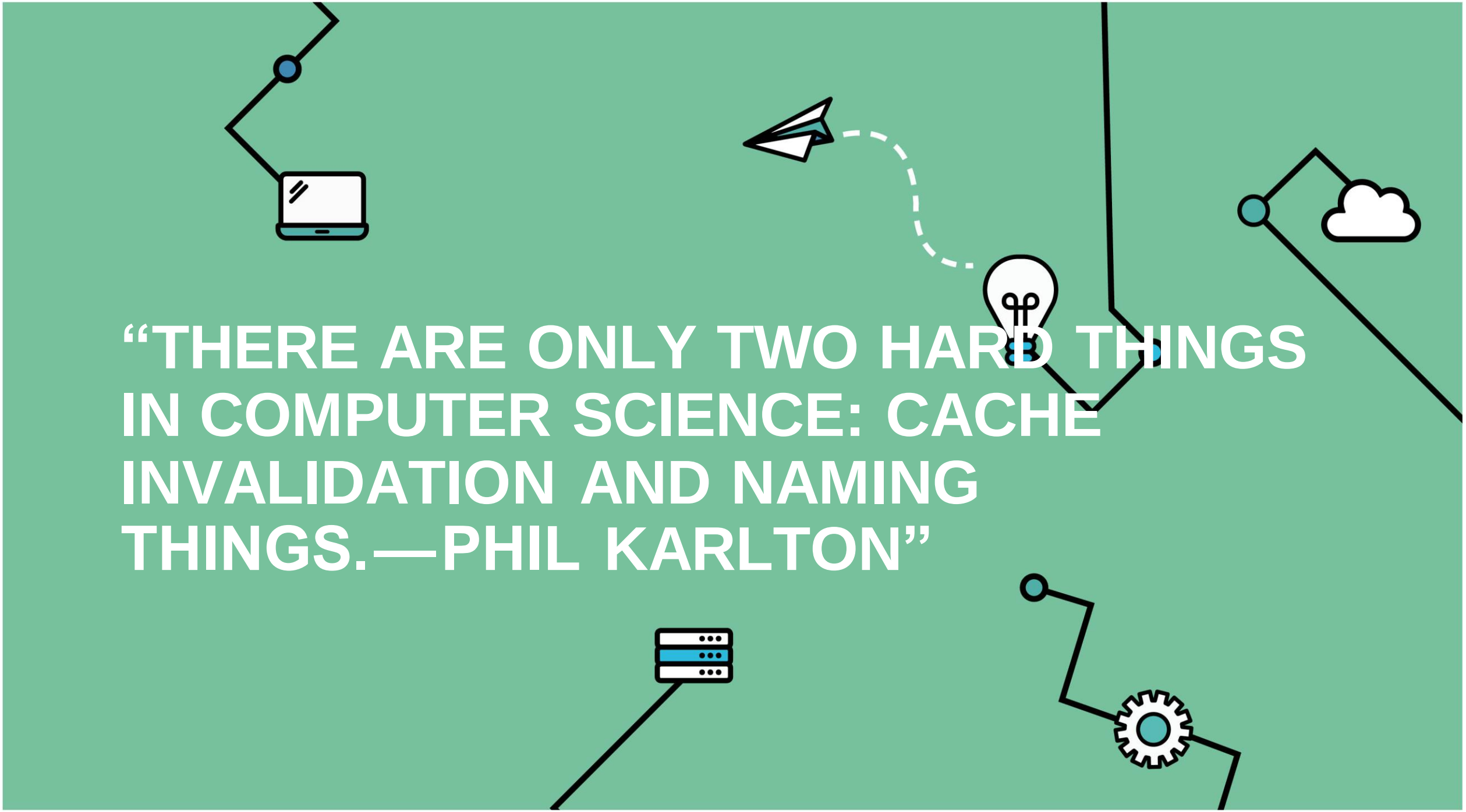
- Float represents real numbers
- Are numbers with decimals
- The period . Is used as a separator
- Floats have a maximum precision, however in most cases, floats are 'accurate enough'
- Use ***float()*** function to cast a value to a float; will return the value between the parentheses as a float (adding .0 if necessary)

STRING

- A string is a text, consisting of zero or more characters.
- Text data must be enclosed by either single or double quotes
- Strings are immutable
- Use ***str()*** function to cast a value to a string
- There are a LOT of useful built-in string functions

BOOLEAN

- A Boolean expression (or logical expression) evaluates to one of two states **true** or **false**.
- Python provides the Boolean type that can be either set to ***False*** or ***True***.
- The data type is *bool*
- Every value can be interpreted as a Boolean value, regardless of its data type
- Booleans are returned by Comparison Operators (<, <=, >, >=, ==, !=)
- Membership Operators((not) in) return Booleans as well


“THERE ARE ONLY TWO HARD THINGS IN COMPUTER SCIENCE: CACHE INVALIDATION AND NAMING THINGS.—PHIL KARLTON”

NAMING CONVENTIONS (1)

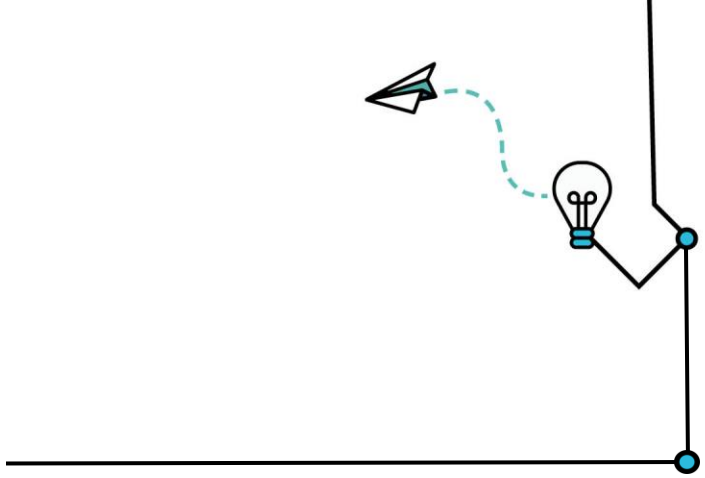
```
In [6]: a = 3.14159265
        b = 7.5
        c = 8.25
        d = a * b * b * c / 3
        print( d )

485.96511304687505
```

```
In [7]: pi = 3.14159265
        radius = 7.5
        height = 8.25
        volume_of_cone = pi * radius * radius * height / 3
        print( volume_of_cone )

485.96511304687505
```

NAMING CONVENTIONS (2)



Some "Reserved" words are:

```
and       del       from      not       while
as        elif      global    or        with
assert    else      if        pass      yield
break     except    import    print
class     exec      in        raise
continue  finally   is        return
def       for       lambda    try
```

ts, and/or

d

#5

Put underscore between words, if a variable name consists of multiple words

"An exception to choosing meaningful variable names is choosing names for "throw-away" variables"

CONDITIONS

The syntax of the `if` statement is as follows:

```
if <boolean expression>:
    <statements>
```

Note the colon (`:`) after the boolean expression, and the fact that `<statements>` is indented.

Then this expression is going to evaluate to False, thus it will not be executed

- Boolean-expressions evaluate to True or False, there are no other Boolean values.
- Boolean expressions can be combined with logical operators. There are three logical operators, ***and***, ***or***, and ***not***

ITERATIONS (1)

- If you want to perform repetitious tasks in Python, you should consider using loops
- There're two types of loops in Python, ***while*** loop and ***for*** loop.
- A while loop is very similar to an "if-statement"
- CAUTION: watch out for accidentally creating an endless loop

"Computers do not get bored, If you want the computer to repeat a certain task hundreds of thousands of times, it does not protest. "

ITERATIONS (2)

- An alternative way of implementing loops are ***for*** loops
- ***for*** loops tend to be easier and safer to use
- However, ***for*** loops cannot be applied to all iteration problems in contrary to ***while*** loops
- ***for*** loops are applied on a collection of items, and it will process these items, in order, one by one.

"Computers do not get bored, If you want the computer to repeat a certain task hundreds of thousands of times, it does not protest. "

FUNCTIONS IN PYTHON

- A function is a block of reusable code that performs some action, just like in any other programming language
- You "call" a function with some parameters if the function requires them and it will **return** a result
- Not all functions 'return' a result e.g. a value that you can use in your code, for example ***print()***
- We may consider a function as a "black box"

FUNCTIONS IN PYTHON (2)

- Besides those "basic" functions that we just introduced. Python also offers modules that contain many more useful functions
- These modules with their accompanying functions are not ***imported*** by default, you must ***import*** the module
- Alternatively, you can import only specific functions from a module
- When you have a general problem, always investigate if there's a module that helps you solving your problem

FUNCTIONS IN PYTHON (3)

- Creating your own functions! Why would you like to do that?
    - Encapsulation
    - Generalization
    - Manageability
    - Maintainability
    - Reusability

STRING OPERATIONS

- Most of the time, you will be dealing with textual information
- Strings are immutable
- There is a collection of methods that are designed to operate on strings
- These operations **do not** change the string, but they return a changed version of the string.
- Example of operations are: ***strip(), upper(), lower(), find(), replace(), split()*** etc.

TEXT FILES

- One of the most important uses of Python for data processing is the reading, changing, and writing of text files
- Working with files might give a sense of lack of control, you can overcome this by adding ***print()*** statements
- Best practices:
- Use: ***with open('filename') as f:*** to open a file
- Use ***for line in f:***

  print(line, end='') : for reading lines from a file

LISTS

- A list is a collection of elements
- In Python, lists are recognizable by the following structure [element*]
- The elements are *ordered*, as a result, you can access each element of a list using an index
- Lists are ***mutable***, you can change the contents of a list
- Python supports several methods to change the contents of a list, such as ***insert()***, ***remove()***, ***sort()*** etc..

DICTIONARIES

- Basically, dictionaries store ***"key-value pairs".*** Any immutable data type can function as a key. A very common type to use as key is the string.
- Dictionaries are *insertion ordered* collections of elements. (python 3.7+)
- You create dictionaries using curly brackets ({}), like how you create lists using square brackets ([ ]).
- Dictionaries are like lists ***mutable*** objects

TUPLES

- Tuples are identical to lists in all respects, except for the following properties:
    - The elements of a tuple are enclosed by parentheses (()).
    - Tuples are ***immutable***

Why use a tuple instead of a list?

- Tuples are faster
- Sometimes you don't want data to be modified
- A tuple can be a dictionary key, a list not

SETS

•Sets are an unordered data structure containing no duplicate elements.

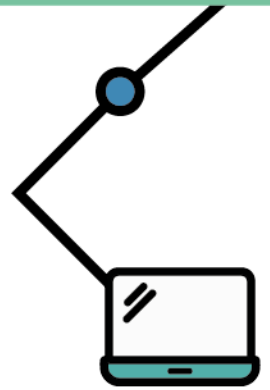
- The elements of a set are enclosed by curly brackets ({}).
- Sets are ***mutable***
- Are highly optimized for presence checking of an element in comparison to list.

HANDS-ON EXERCISES - CONTENT

- Here we are sharing a web scraper that scrapes the most popular (frequently downloaded) e-books from Gutenberg.org.
- That's a cool website that contains free books – mostly classics.
- The scraper will provide a dictionary dataset containing some interesting information about books.
- The dictionary is nested, so we will be using lots of comprehensions…
- To get more insights let's check the Jupyter Notebook.

HANDS-ON EXERCISES - QUESTIONS

1. We have a nested dictionary of book information. Write a function to create a list of tuples, including only the book_name and book_downloads.
2. Write a function to sort book_name and downloads by checking the number of the downloads. (ascending is fine)
3. Write a function to count words in each text (string). (please use simple tokenizer given in the notebook)
4. Write a function with a dictionary that is stating which words are used in the text how many times.
5. Use download_book_text to download all the popular books with their own book name. (bonus question)
6. By using the downloaded book texts in the previous step, count the words in each book. (bonus question)



LINKIT

BUILDING IT TOGETHER