Day 1

Python Bootcamp

GENERAL ASSEMBLY

Meet Your Instructor

Arwa Lokhandwala

Developer, Instructor, Speaker & Blogger

- Full Stack Developer with over 5+ years of experience developing scalable web applications
- Ex Lead Instructor for Software Engineering Immersive @GA
- Ex Telstra, Reliance Jio, BookMyShow
- Technical Blogger & Speaker

Arwa Lokhandwala @code.with.arwa

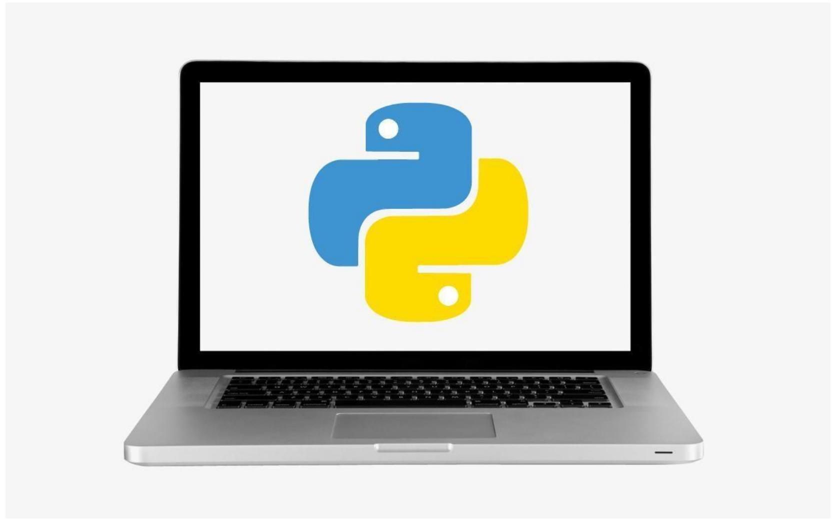
Agenda

- About Python (WHY)
- Functions (using)
- Type and variables
- Importing modules
- Control Structures with If/else
- How to Loop
- Functions (defining)
- File I/O
- Creating a Chart

About this course

Getting the most out of this course

- Make sure you have the tools you need running smoothly
- Think / ask how Python could fit into your workflow
- The exercises are guidelines, pursue your interests in during practice
- Plan how you will continue your learning



Discussion: About You

Introduce yourself:

- **Name**
- **What Languages have you coded in?**
    - None!, Excel, SQL, HTML, BASH/DOS, C, Python, etc.
- **What industries are you interested in?**
    - Finance, Technology, Medicine, Publishing, Education etc.
- **How will this course help you with your goals?**
- **Share something you recently read/watched/heard.**

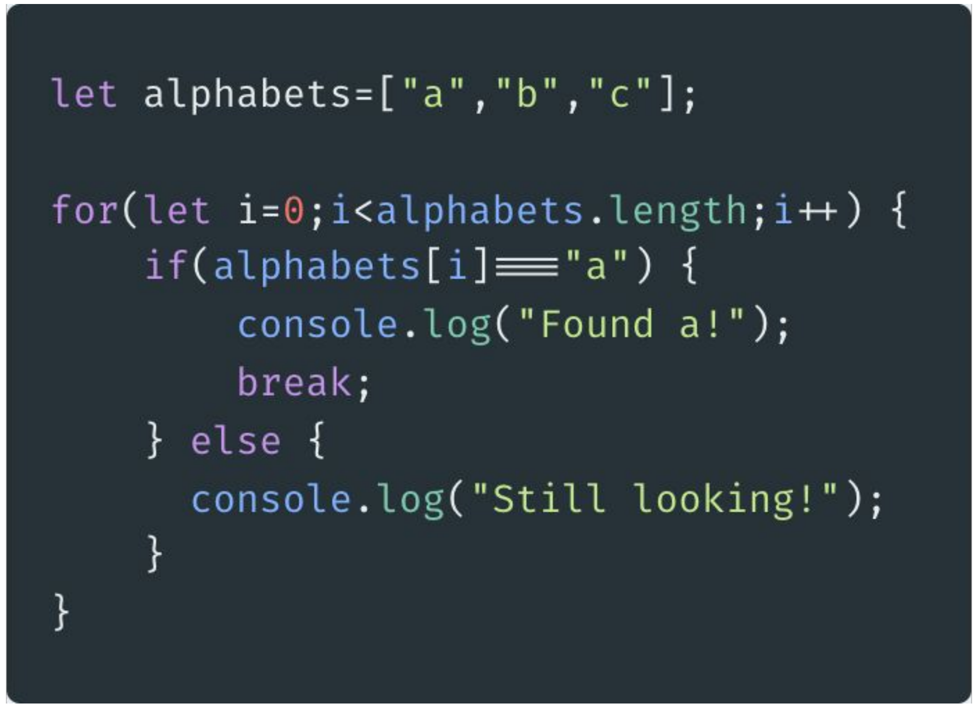
Python Bootcamp

All About Python

What is Python?

- Created by Guido Van Rossum in 1991
- Emphasizes **productivity** and code **readability**
- **Easy** to pick up and learn
- Easier for many to contribute to **production level code**
- **Readable code** means that almost anyone can read and **understand what the code is doing**

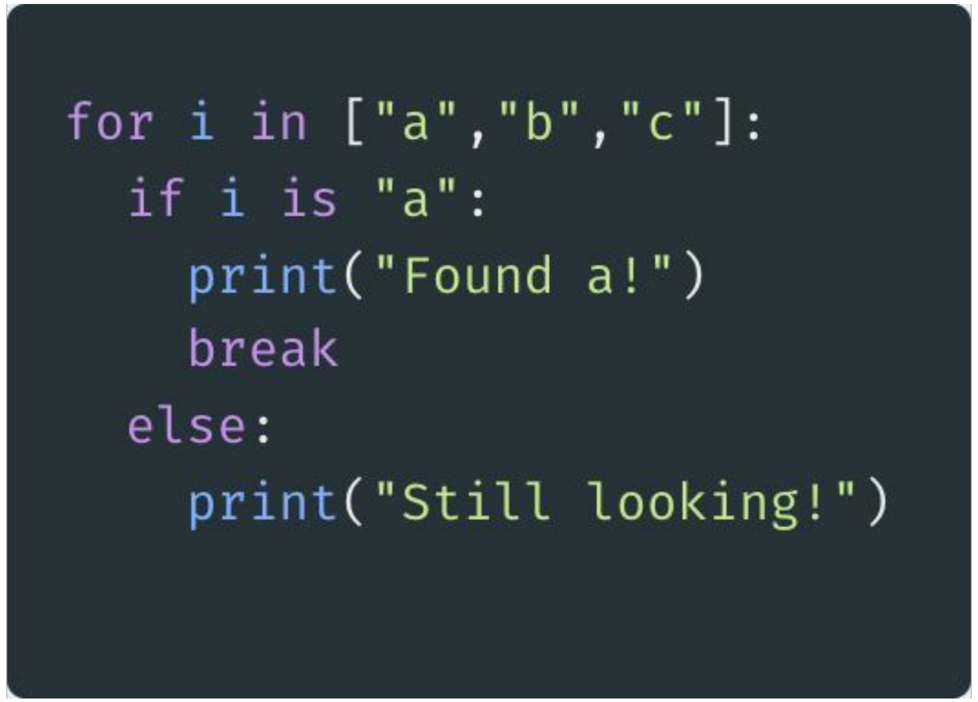
JavaScript



```javascript
let alphabets=["a","b","c"];

for(let i=0;i<alphabets.length;i++) {
    if(alphabets[i]==="a") {
        console.log("Found a!");
        break;
    } else {
      console.log("Still looking!");
    }
}
```

Python



```python
for i in ["a","b","c"]:
  if i is "a":
    print("Found a!")
    break
  else:
    print("Still looking!")
```

Why is Python readable?

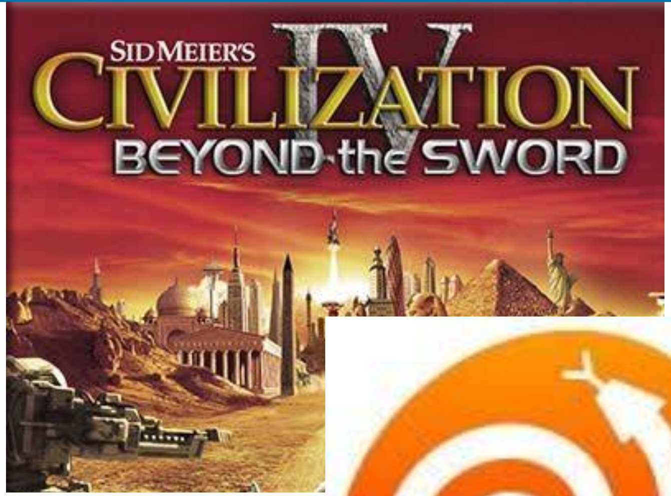
- **Interpreted language**:
  - Step by step execution for easier programming ideation
  - Write once, run anywhere
  - Performance tradeoff
- **Object-oriented (OO)**
  - Code with objects that contain data and functions to manipulate it in predefined ways
- **High-level programming**
  - Use natural language syntax where possible

Typical Programs using Python (REAL WORLD)

- **Data Science / Machine Learning:**
  - Analyse data and create predictive models.
    - Pandas, ScikitLearn, Tensorflow
- **Web Applications:**
  - Backends for website or mobile apps.
    - Django, Flask.
- **Data Engineering:**
  - Prepare data for machine learning / big data applications
    - ETL Scripts, Data Pipelines, Data Analysis
- **DevOps/SysOps:**
  - Maintain fleet of servers and live applications -
    - Orchestration tools like Ansible

Real Cases: Who uses Python?

- **Industry & Academia**
  - AstroPy
  - BioPython
- **Web Development**
  - Youtube
  - DropBox
- **Game Development**
  - Civilization IV
- **Standalone Applications**
  - BitTorrent



How to WRITE and RUN Python

WHAT	WHY	HOW TO EXECUTE CODE
SHELL	FOR DISCOVERY / QUICK EXPERIMENTS	AT THE COMMAND LINE RUN "PYTHON". THEN TYPE CODE. ENTER EXECUTES.
NOTEBOOK	FOR LEARNING AND DATA SCIENCE	CODE IS TYPED IN CELLS. TO EXECUTE A CELL, TYPE SHIFT + ENTER
CODE EDITOR	FOR BIGGER PROGRAMS	TYPE CODE IN FILE. SAVE THE FILE. THEN EXECUTE "PYTHON <FILENAME>" AT THE COMMAND LINE.

Writing code a.k.a PROGRAM

“ A program is a sequence of instructions that specifies how to perform a computation ”

- Input: Gets data or instructions from a file, a keyboard, an API, etc.
- Output: Sends data to a file, a database, a screen, etc.
- Modifies Data: Perform some mathematical operations
- Conditional Execution: Checks for condition to execute the appropriate code
- Repetition: Repeats an action a number of time, with some variations

https://greenteapress.com/thinkpython/html/thinkpython002.html

Debugging

Programs are typically full of errors, which we call **bugs.** Learning to debug is one of the most important skills to be a productive developer.

Types of Errors to look out for:

- **Syntax Errors:**
  - The simplest one. It means the program is not respecting the rules of the language. As you get experienced, you'll make fewer of these mistakes, and learn to correct them quickly
- **Runtime Errors:**
  - The program crashes while running. They are often called *Exceptions*
- **Semantic Errors:**
  - The programs runs without crashing, but it does not do what you want. These are the hardest bugs to fix.

Always remember that "There will be bugs".

JUPYTER NOTEBOOK

- Most convenient way to learn – easy to run and re-run code
- Typically used by data scientists/ data analysts
- The Notebook consists of "Cells" that are space to write and execute code
- We can see the results immediately
- We’re using an online notebook today, Google Colab

Google Colab

- Jupyter Notebook hosted by Google Colab
- Completely online, no-installation required
- Collaboration is very easy
- It’s like Google docs for your Jupyter Notebooks
- Easy to use Tensorflow and other ML libraries

https://colab.research.google.com/notebooks/intro.ipynb

Python Workshop Notebook

1. Go to https://github.com/arwalokhandwala/python-bootcamp
2. Click on day01.ipynb
3. Click on Open in Colab

What’s Next?

Create a Learning Plan

Solidify your learning:

- Go through the parts of Learn How to Think Like a Computer Scientist.
- Familiarize yourself with the language by going through A Beginner's Python Tutorial.

Practice Practice Practice! Problems to expand your skills are available at:

- HackerRank
- CodeWars

Real Cases: Expanding python

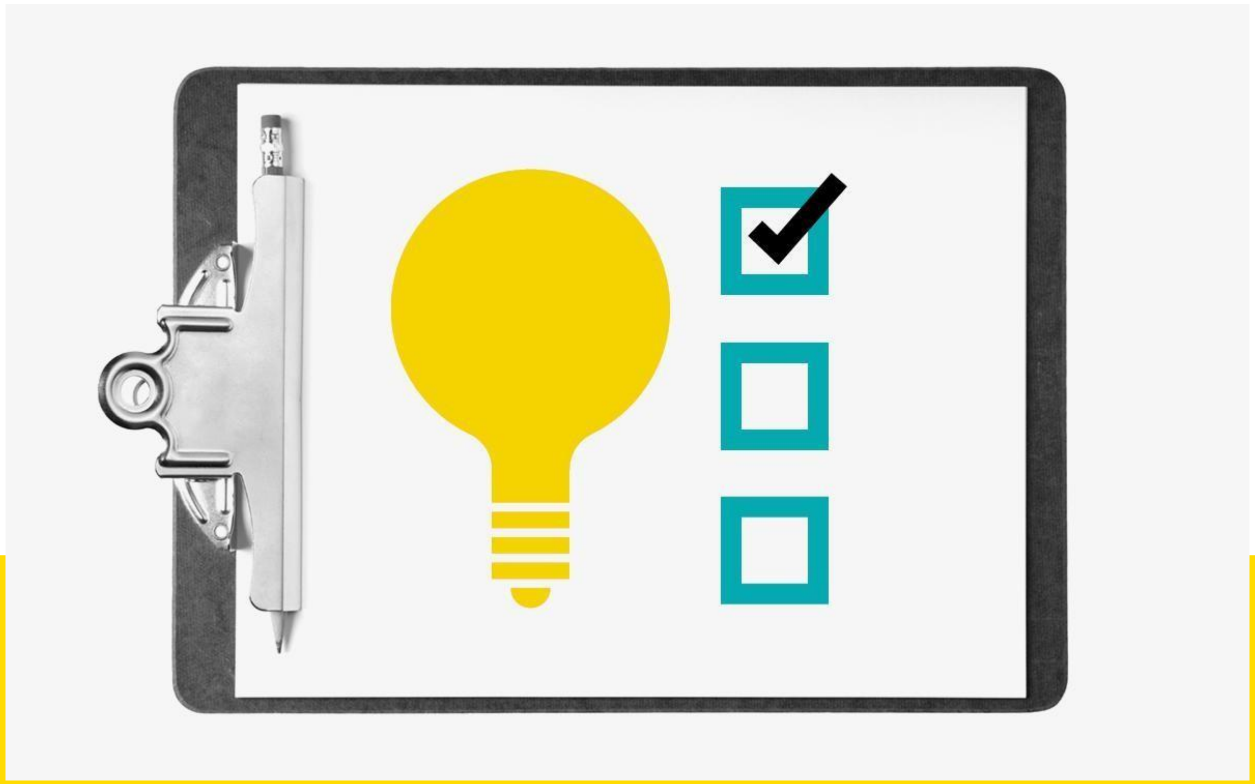
Common Packages

- Data manipulation: pandas, Numpy, scipy
- Machine Learning: scikit-learn, nltk
- Databases: psycopq2, sqlalchemy
- Visualizations: matplotlib, plotly, bokeh
- API calls / web scraping: requests, BeautifulSoup, Scrapy
- Web development: Django, Flask, Twisted, Scapy
- Game Development: Pygame, Pyglet
- Desktop App: pyQt, Tkinter

More

A Few Good References

1. Official Python Documentation
2. PEP-8 Official Guide
3. Anaconda Tutorials
4. Jupyter Documentation
5. Example Notebooks



See you next time!

Thank you!

GENERAL ASSEMBLY