Introduction to Python Programming

Learn fundamental programming concepts in a beginner friendly language

Pearson

Set up

GitHub repository

- Go to https://github.com/ariannedee/intro-to-python
- Follow the installation instructions
    - Install Python 3.6 or higher
    - Install an IDE (I'll be using PyCharm Community)
    - Download the code

Resources widget

- Download the PDF of these slides
- Download the PyCharm Reference PDF

Video link

Pearson

Set up

GitHub repository

- Go to https://github.com/ariannedee/intro-to-python
- Follow the installation instructions
  - Install Python 3.6 or higher
  - Install an IDE (I'll be using PyCharm Community)
  - Download the code

Resources widget

- Download the PDF of these slides
- Download the PyCharm Reference PDF

Video link

Introduction

Today’s schedule

- **Introduction and set-up** (20 mins)
- **First code** (30 mins)
    - Break
- **Learn programming basics** (90 mins)
    - Break
- **Control the flow with conditionals** (30 mins)
    - Break
- **Work with lists and loops** (40 mins)
- **What to learn next** (5 mins)

Pearson

Questions and breaks

- Use group chat throughout class
  - Only ask questions relevant to current discussion
  - If it's too specific or if I need to do research, put in the Q&A
  - Anyone can answer

- 3 Breaks (10 mins each)
  - Step away or work through code
  - I'll answer questions in the Q&A feature
  - Ask general or more in-depth questions

- Email more in-depth questions at arianne.dee.studios@gmail.com

Poll

- How much programming do you already know?
  - Absolutely none
  - A little bit
  - A moderate amount
  - A lot

Poll (multi-choice)

- What are your eventual goals with learning Python
  - Career change
  - Better understanding and communication
  - Use it in my current career (as a non-developer)
  - Use it in my current career (as a developer)
  - For fun
  - Other

Pearson

Introduction

Installation

Set up

GitHub repository

- Go to https://github.com/ariannedee/intro-to-python
- Download project code
- Follow the installation instructions

Resources widget

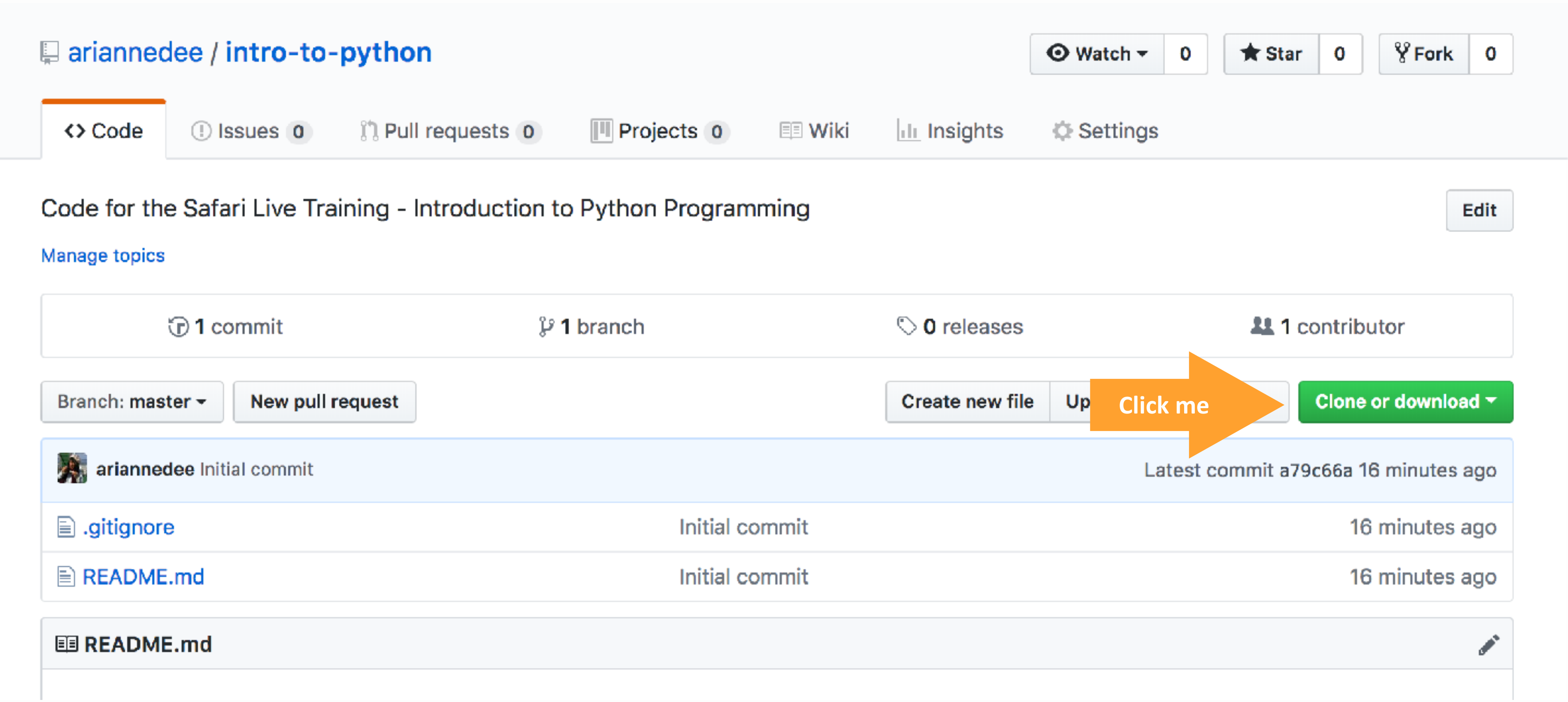
- Download the PDF of these slides
- Download the PyCharm Reference PDF

Video link

Pearson

Install links

- Download the code
  - https://github.com/ariannedee/intro-to-python
- Install Python 3.6+ for your operating system
  - https://www.python.org/downloads/
- Download the free, community edition of PyCharm
  - https://www.jetbrains.com/pycharm/download/
  - Or use another IDE, like VS Code or Spyder


ariannedee / intro-to-python

Code for the Safari Live Training - Introduction to Python Programming

Manage topics

1 commit | 1 branch | 0 releases | 1 contributor

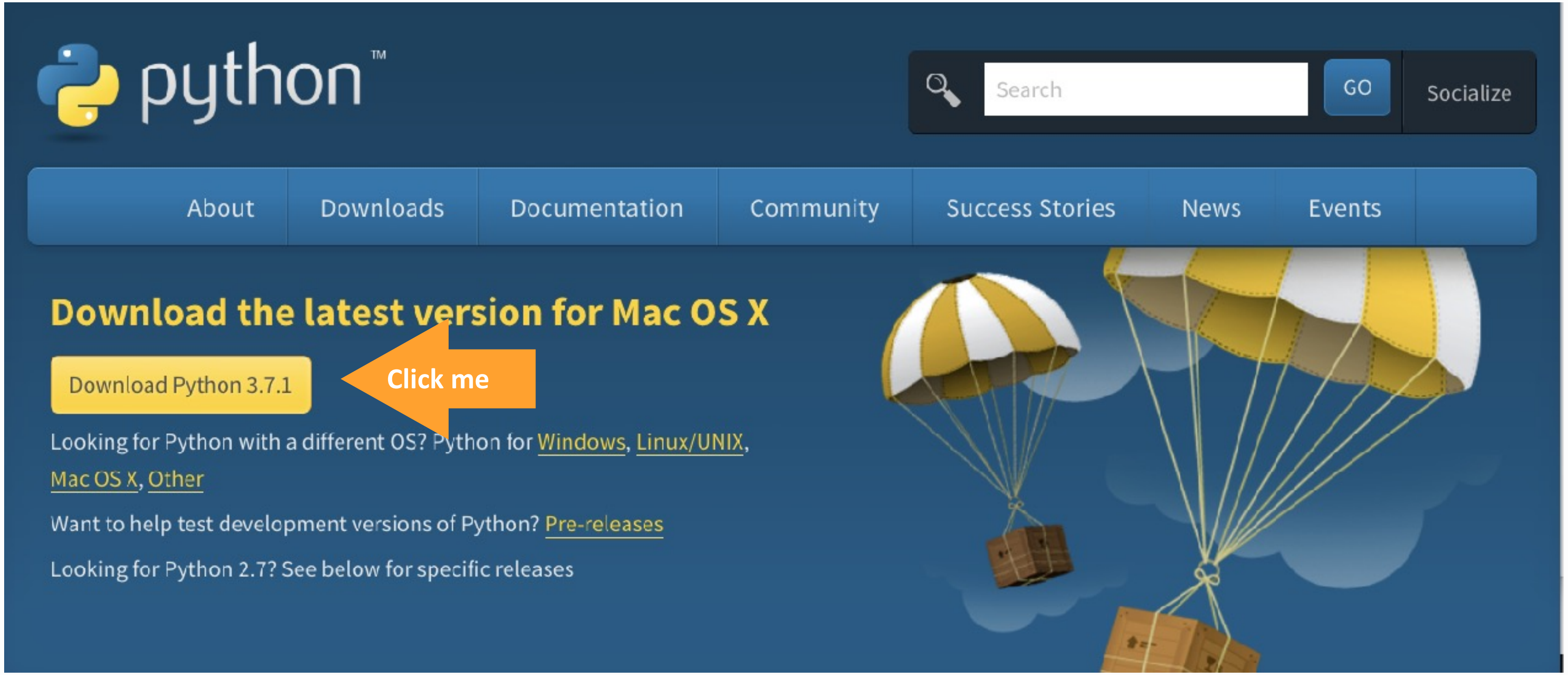
ariannedee Initial commit — Latest commit a79c66a 16 minutes ago

.gitignore	Initial commit	16 minutes ago
README.md	Initial commit	16 minutes ago

README.md

https://github.com/ariannedee/intro-to-python

www.python.org/downloads



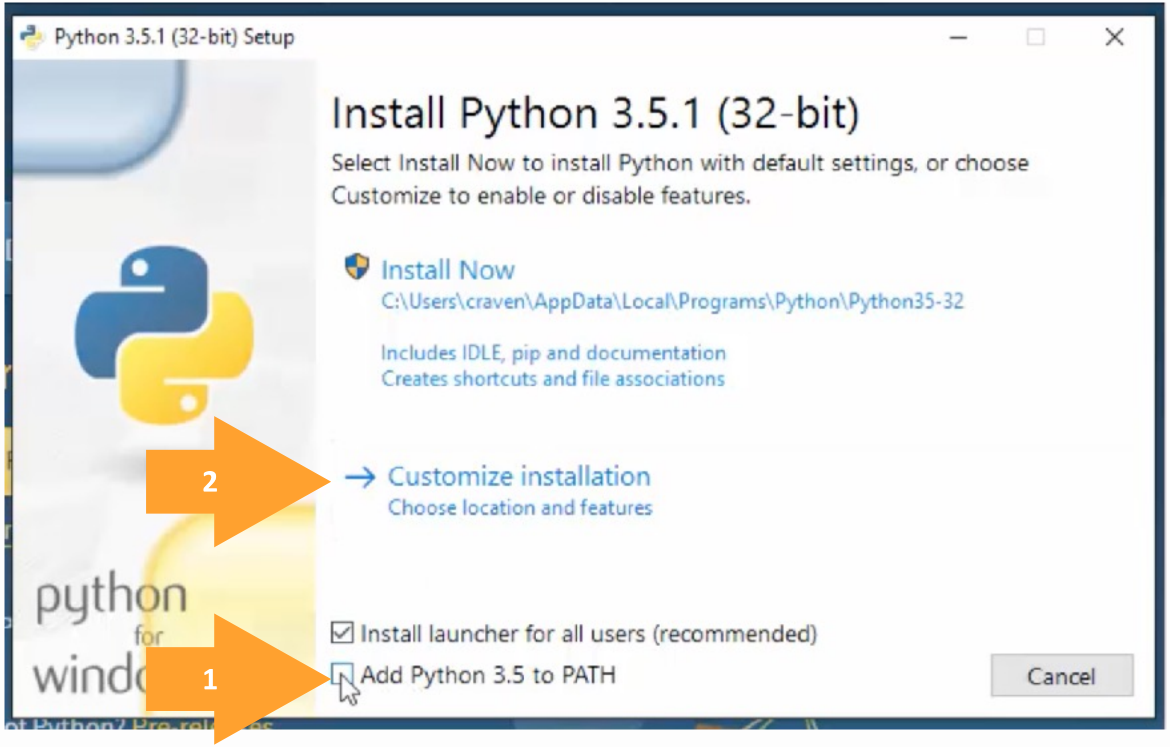
Download the latest version for Mac OS X

Looking for Python with a different OS? Python for Windows, Linux/UNIX, Mac OS X, Other

Want to help test development versions of Python? Pre-releases

Looking for Python 2.7? See below for specific releases

On Windows



Python 3.5.1 (32-bit) Setup

Install Python 3.5.1 (32-bit)

Select Install Now to install Python with default settings, or choose Customize to enable or disable features.

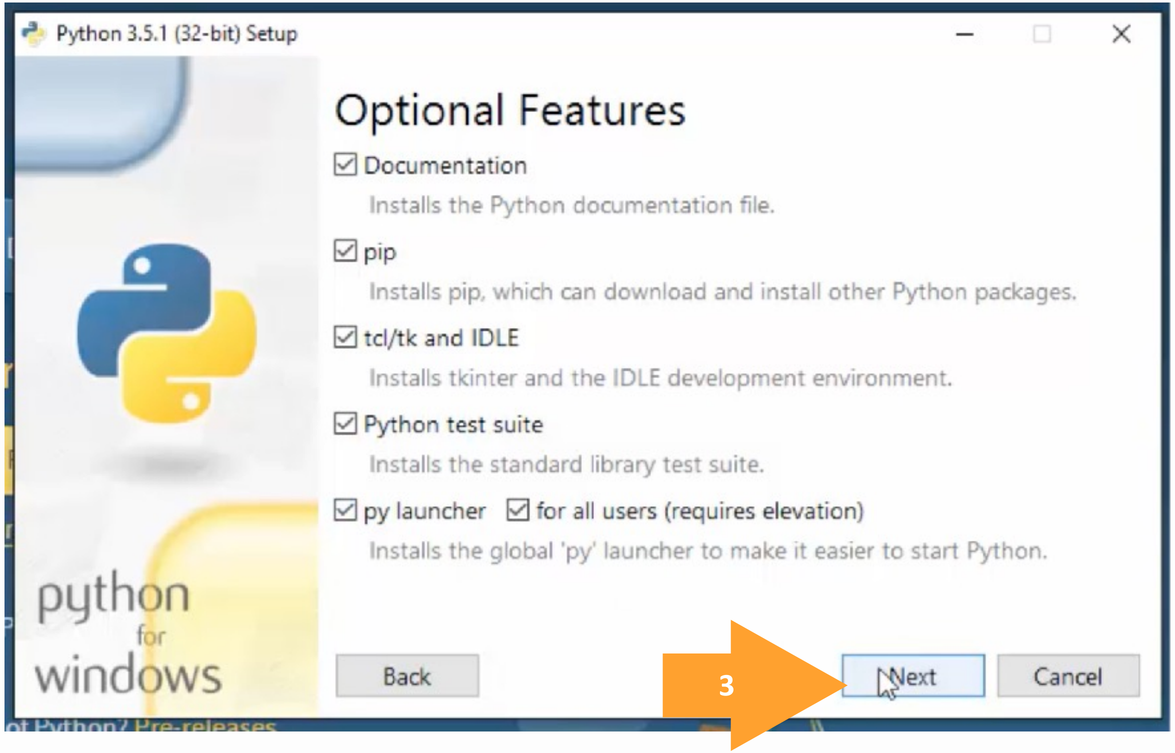
Install Now
C:\Users\craven\AppData\Local\Programs\Python\Python35-32

Includes IDLE, pip and documentation
Creates shortcuts and file associations

Customize installation
Choose location and features

☑ Install launcher for all users (recommended)
☐ Add Python 3.5 to PATH

On Windows



Python 3.5.1 (32-bit) Setup

Optional Features

☑ Documentation
Installs the Python documentation file.

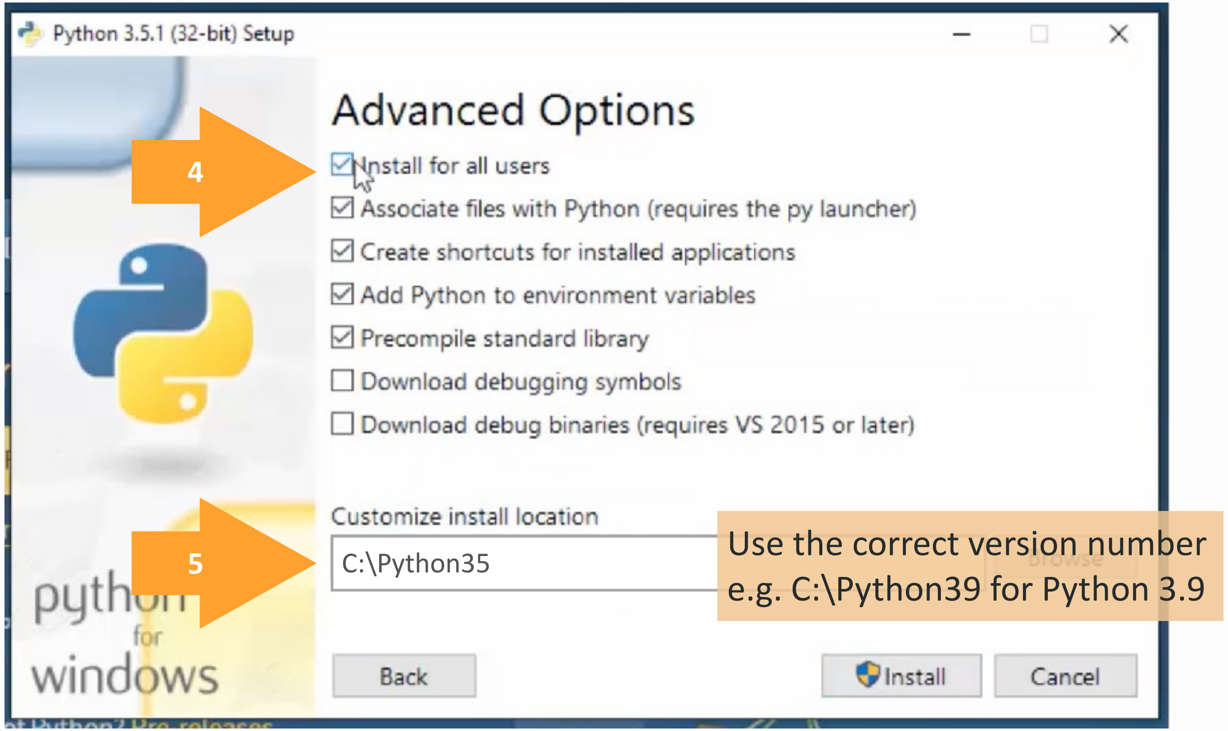
☑ pip
Installs pip, which can download and install other Python packages.

☑ tcl/tk and IDLE
Installs tkinter and the IDLE development environment.

☑ Python test suite
Installs the standard library test suite.

☑ py launcher ☑ for all users (requires elevation)
Installs the global 'py' launcher to make it easier to start Python.

On Windows



Python 3.5.1 (32-bit) Setup

Advanced Options

- [✓] Install for all users
- [✓] Associate files with Python (requires the py launcher)
- [✓] Create shortcuts for installed applications
- [✓] Add Python to environment variables
- [✓] Precompile standard library
- [ ] Download debugging symbols
- [ ] Download debug binaries (requires VS 2015 or later)

Customize install location

C:\Python35

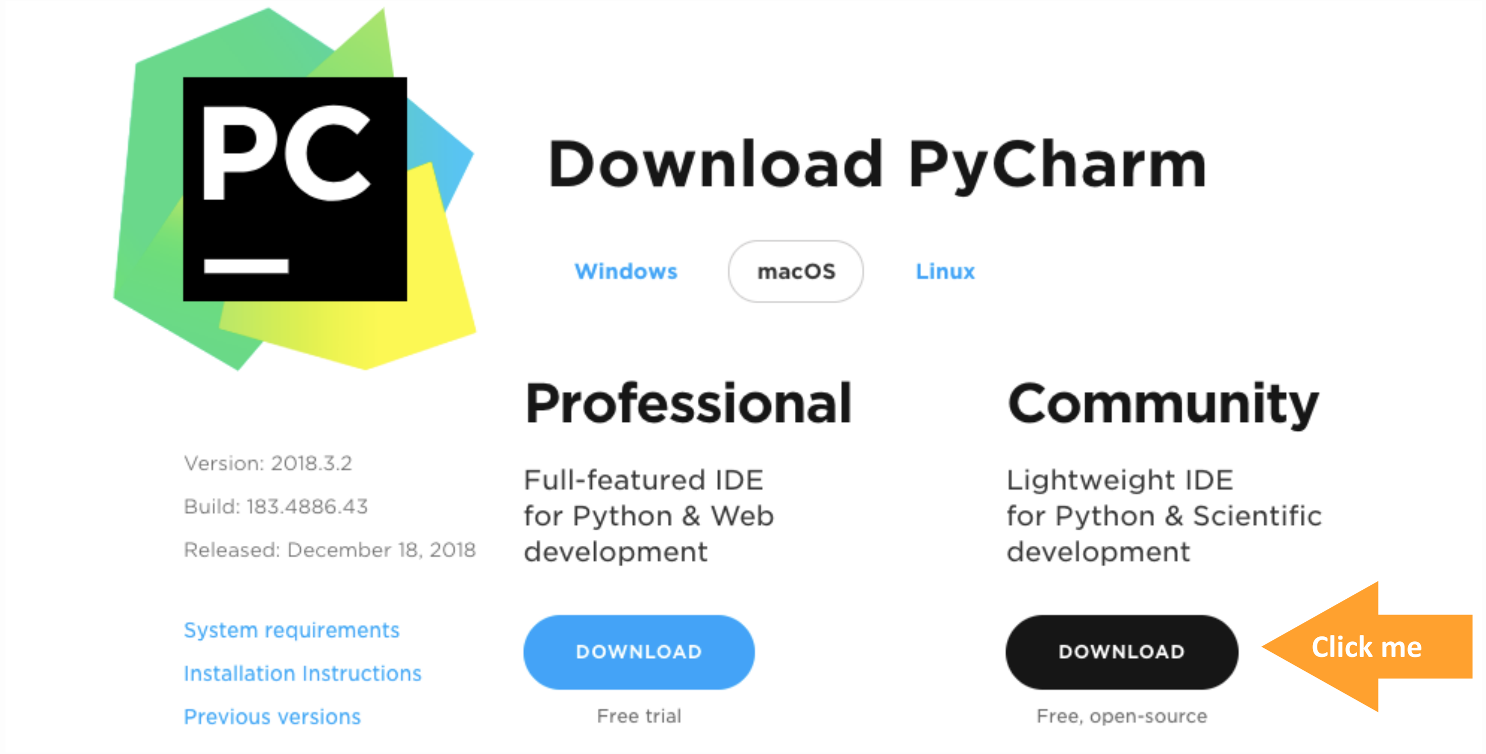
Use the correct version number
e.g. C:\Python39 for Python 3.9

If you already installed Python

Follow the instructions to add Python to your PATH

Link

www.jetbrains.com/pycharm/download/



Download PyCharm

Professional

Full-featured IDE for Python & Web development

Free trial

Community

Lightweight IDE for Python & Scientific development

Free, open-source

Version: 2018.3.2

Build: 183.4886.43

Released: December 18, 2018

Integrated Development Environment (IDE)
PyCharm

Why we're using PyCharm

- Handles **Python** out of the box
  - Syntax highlighting
  - Error highlighting
  - Code suggestions
- Better for **beginners** who don't know the command line
- Full-featured for **professional** Python developers

Alternatives to PyCharm

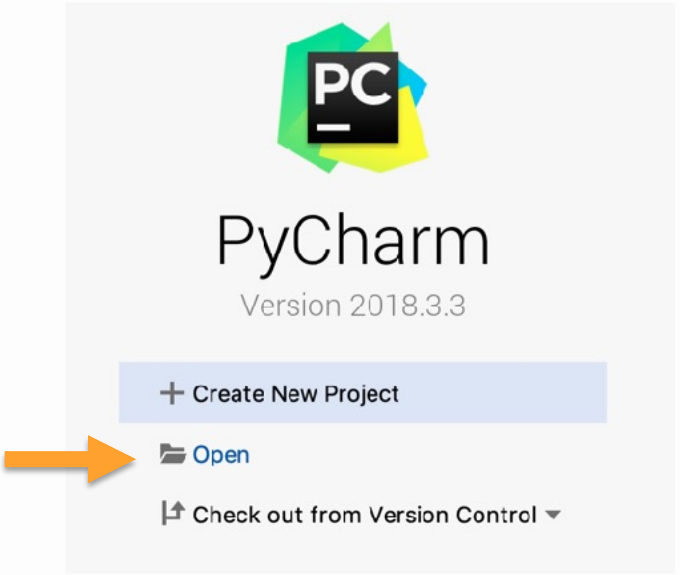
- Anaconda
  Data scientists
- Thonny
  Absolute beginners
- Sublime text, VS code, Atom
  Multi-lingual programmers
  Requires plug-ins to fully support Python
- Notepad, Notepad++, Vim, Emacs
  Old-school programmers
- https://realpython.com/python-ides-code-editors-guide/

First code

Hello world!

Video link

Open project files

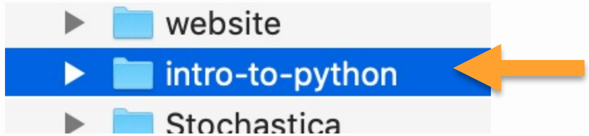

PyCharm

Version 2018.3.3

+ Create New Project

Open

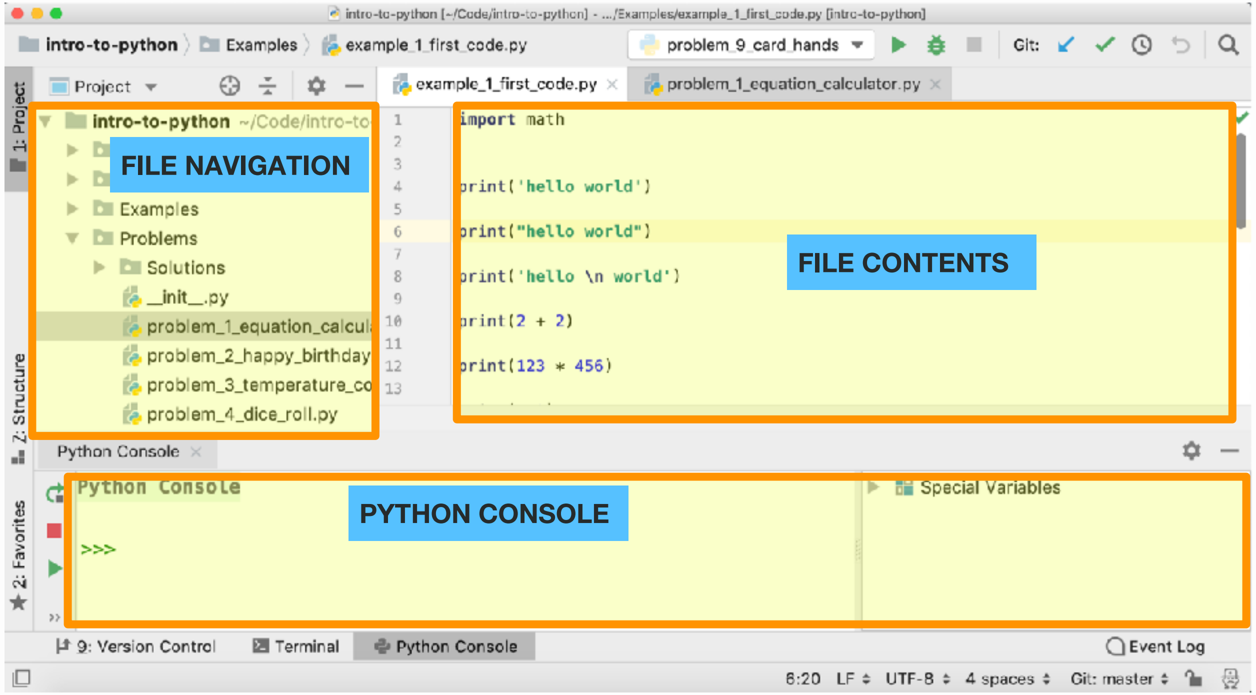
Check out from Version Control


website

intro-to-python

Stochastica

PyCharm Layout

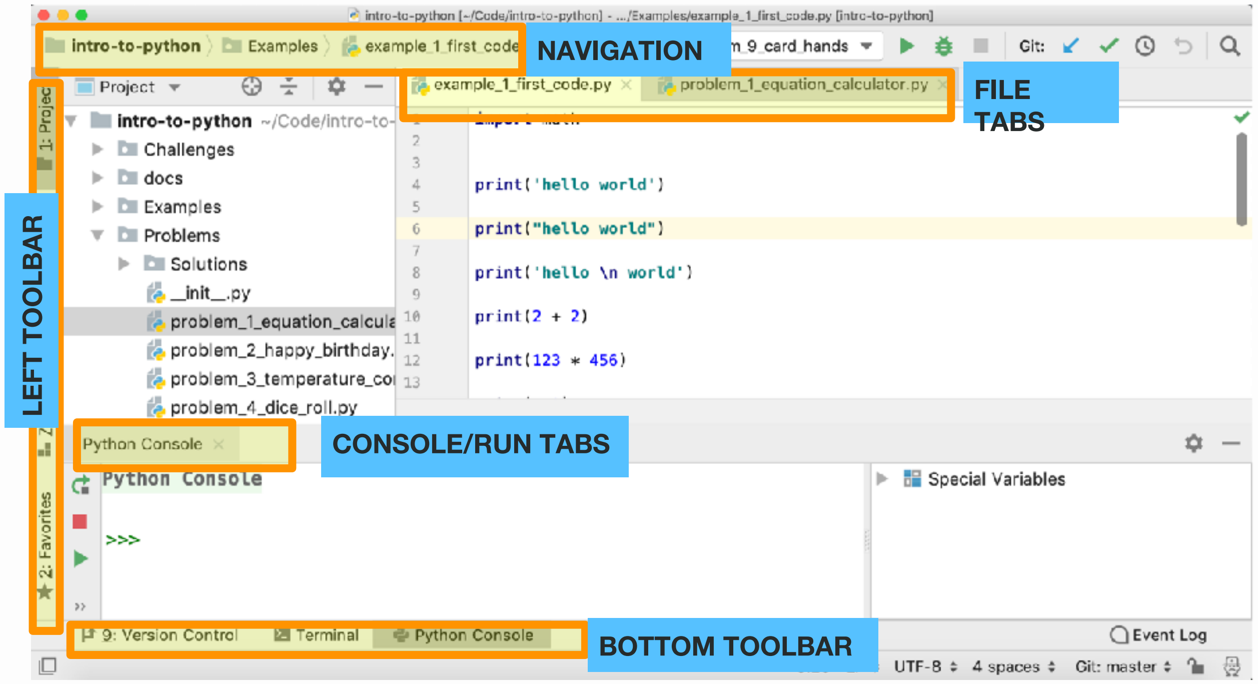


```
import math


print('hello world')

print("hello world")

print('hello \n world')

print(2 + 2)

print(123 * 456)
```

```
Python Console
>>>
```

Reference: page 2

PyCharm Toolbars

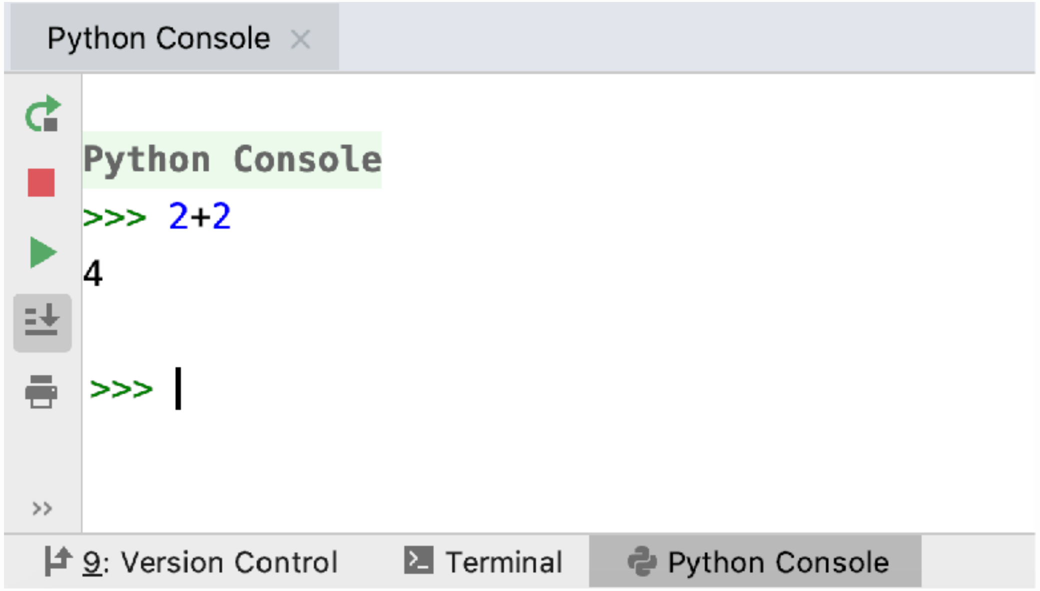


```
import math


print('hello world')

print("hello world")

print('hello \n world')

print(2 + 2)

print(123 * 456)
```

```
Python Console

>>>
```

Reference: page 2

Run code in the console

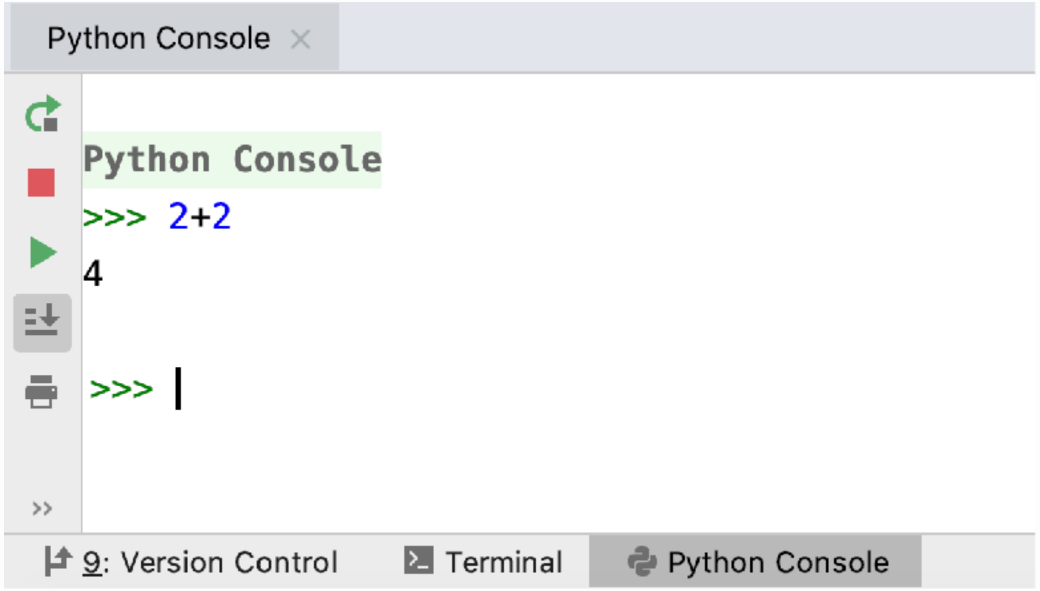


```
Python Console
>>> 2+2
4

>>> 
```

If PyCharm doesn’t recognize Python3

Follow the online instructions or refer to pages 4-6 of the reference document (in the resources widget)

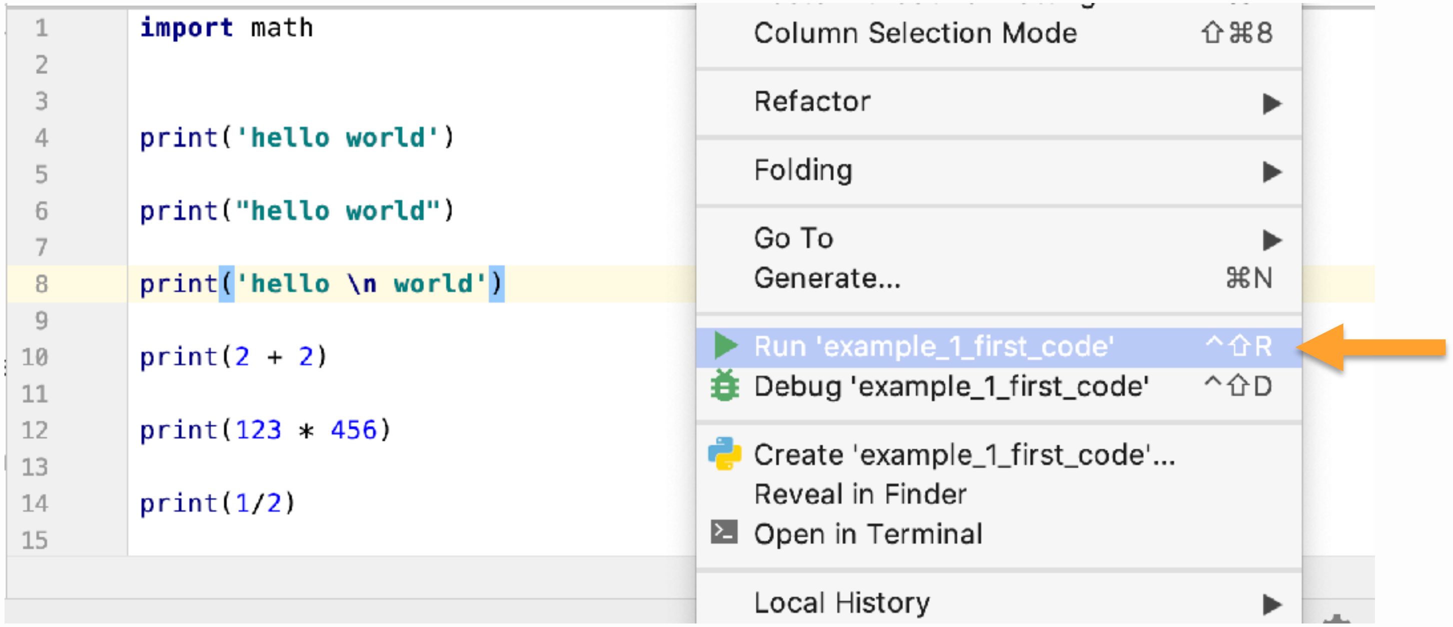
Run code in the console



```
Python Console
>>> 2+2
4

>>> 
```

Run code from a file

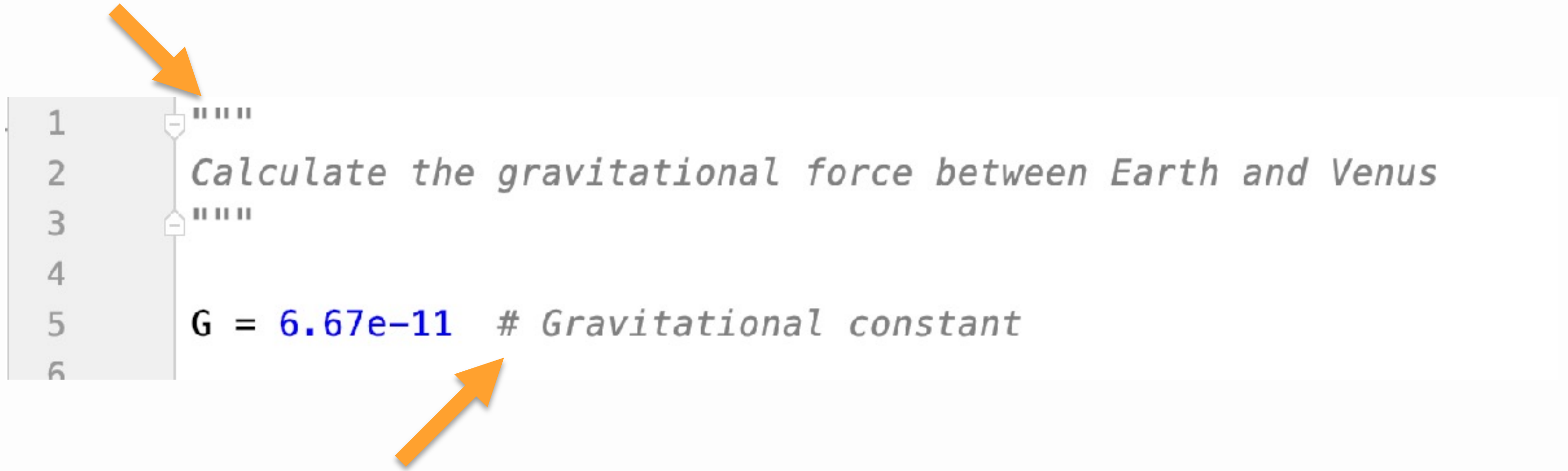
Right click in file



```
import math


print('hello world')

print("hello world")

print('hello \n world')

print(2 + 2)

print(123 * 456)

print(1/2)
```

Column Selection Mode ⇧⌘8
Refactor
Folding
Go To
Generate... ⌘N
Run 'example_1_first_code' ^⇧R
Debug 'example_1_first_code' ^⇧D
Create 'example_1_first_code'...
Reveal in Finder
Open in Terminal
Local History

Comments



```
"""
Calculate the gravitational force between Earth and Venus
"""

G = 6.67e-11  # Gravitational constant
```

Problem #1

Gravitational force calculator

$$F_g = \frac{G m_1 m_2}{r^2}$$

Other ways to run Python code

- IDLE
- Terminal / Command Prompt

Checking python versions

Open Command Line (PC) or Terminal (Linux, Mac)

```
python --version
```

or

```
python3 --version
```

or

```
python3.10 --version
```

or

```
py --version
```

- One of those commands should return:
    - `Python 3.10.x`

Pearson

Checking Python version

Open Command Line (PC) or Terminal (Linux, Mac)

Try:

- `python --version`
- `python3 --version`
- `python3.11 --version`
- `py --version`

- One of those commands should return:
  - `Python 3.11.x`

Pearson

Running Python in the command line

Use command from previous slide

- `python, python3.11, py`

Open the Python Console

- `python3.11`

Run a file

- `python3.11 <filename.py>`

First code

About Python

Video link

“Hello World” in different languages

- Roughly from high - low level of abstraction

Python

```
print("Hello World")
```

Pearson

JavaScript

```
console.log("Hello World!");
```

C#

```
using System;

class Program
{
    static void Main(string[] args)
    {
        Console.WriteLine("Hello, world!");
    }
}
```

Java

```
class HelloWorldApp {
    public static void main(String[] args) {
        System.out.println("Hello World!"); // Prints the string to the
console.
    }
}
```

Pearson

C++

```
#include <iostream>

int main()
{
    std::cout << "Hello, world!\n";
    return 0;
}
```

Assembly

```
    global  _main
    extern  _printf

    section .text
_main:
    push    message
    call    _printf
    add     esp, 4
    ret
message:
    db  'Hello, World', 10, 0
```

Machine Code

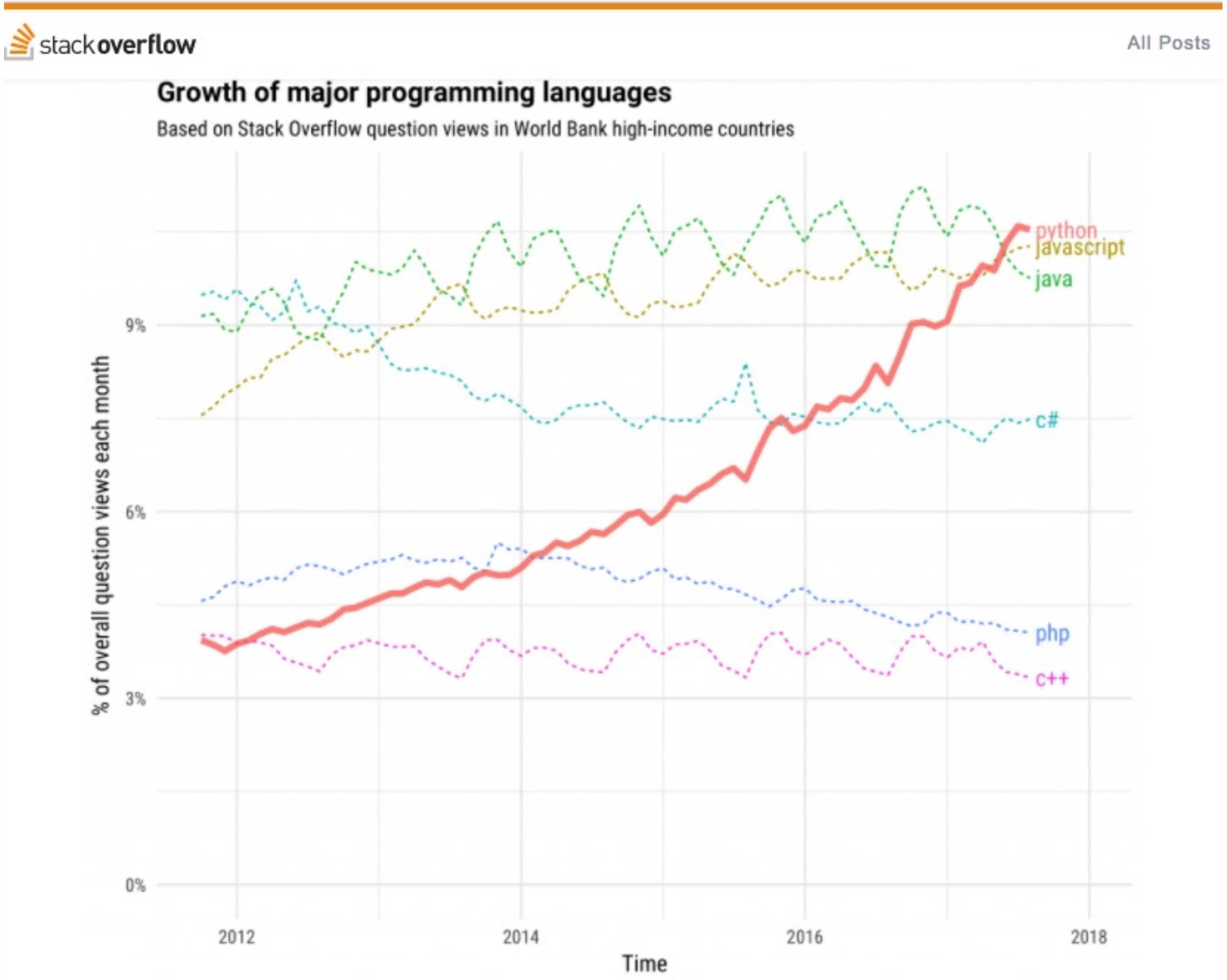
```
b8    21 0a 00 00   #moving "!\n" into eax
a3    0c 10 00 06   #moving eax into first memory location
b8    6f 72 6c 64   #moving "orld" into eax
a3    08 10 00 06   #moving eax into next memory location
b8    6f 2c 20 57   #moving "o, W" into eax
a3    04 10 00 06   #moving eax into next memory location
b8    48 65 6c 6c   #moving "Hell" into eax
a3    00 10 00 06   #moving eax into next memory location
b9    00 10 00 06   #moving pointer to start of memory location into
ecx
ba    10 00 00 00   #moving string size into edx
bb    01 00 00 00   #moving "stdout" number to ebx
b8    04 00 00 00   #moving "print out" syscall number to eax
cd    80            #calling the linux kernel to execute our print to
stdout
b8    01 00 00 00   #moving "sys_exit" call number to eax
cd    80            #executing it via linux sys_call
```

Python

- High-level language
    - Is closer to English than most others

- Simple syntax
    - Easy to learn and get stuff done

- Open source
    - Everything is free, lots of things are well-maintained


stackoverflow
All Posts
Growth of major programming languages
Based on Stack Overflow question views in World Bank high-income countries
% of overall question views each month
9%
6%
3%
0%
2012
2014
2016
2018
Time
python
javascript
java
c#
php
c++

Great for

- Prototyping
- Scripting (automation tasks, managing servers)
- Data analysis and machine learning
- Teaching
- Low - medium traffic web apps
- RaspberryPi

Not great for

- High speed applications
- Multi-threaded applications
- Mobile development
- Easy to learn, hard to master and progress

Where is it used?

Web apps

Data analysis

More common options

- **Desktop apps**
  - Java, Swift/Objective-C *(Mac)*, C# *(Windows)*, JavaScript *(with Electron)*
- **Mobile apps**
  - Kotlin/Java*(Android)*, Swift/Objective-C *(iOS)*, C# *(with Unity)*, JavaScript *(with React Native)*
- **High speed, high reliability, multi-threading**
  - C/C++, Go, Rust

About Python

Video link

Pearson

About Python

- Released in 1990
- Created by Guido Van Rossum
- Python Enhancement Proposals (PEPs)

About Python

- Released in 1990
- Created by Guido Van Rossum
- Python Enhancement Proposals (PEPs)

Pearson

About Python

- Released in 1990
- Created by Guido Van Rossum
- Python Enhancement Proposals (PEPs)

Style Guide (PEP 8)

Indentation

Use 4 spaces per indentation level.

Tabs or Spaces?

Spaces are the preferred indentation method.

Tabs should be used solely to remain consistent with code that is already indented with tabs.

Zen of Python (PEP 20)

Beautiful is better than ugly
Explicit is better than implicit
Simple is better than complex
Complex is better than complicated
Readability counts
…

Try typing “`import this`” into the interpreter.
Next, try typing “`import antigravity`”.

“Code is more often read than written.”

- Guido van Rossum

More about Python

- Why you should learn Python
  - https://yourstory.com/mystory/interesting-facts-about-python-language
- Python Developer Survey 2019
  - https://www.jetbrains.com/lp/python-developers-survey-2019/
- StackOverflow developer survey 2019
  - https://insights.stackoverflow.com/survey/2019#technology
- Python fun facts
  - https://data-flair.training/blogs/facts-about-python-programming/

Skills for programmers

- Structured problem solving
- Redefine success
- Learn how to love learning
- Empathy

Video link

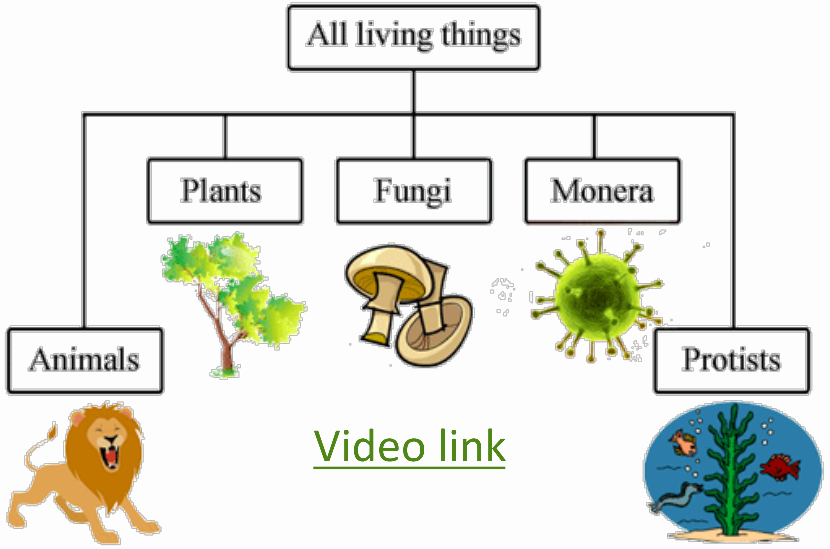
Question & Answer

Learn programming basics

Fundamental concepts

- Types
- Variables
- Errors
- Functions and methods
- Libraries
- Comparisons
- Conditionals (if/else/elif)
- Looping (while/for)
- Lists

Types



Video link

Types

- String - **`str`** - a string of characters, e.g. “Hello 123!”
- Integer - **`int`** - whole number
- Float - **`float`** - with decimal place
- Boolean - **`bool`** - True or False
- None - **`NoneType`** - nothing (nil or null in other languages)

Pearson

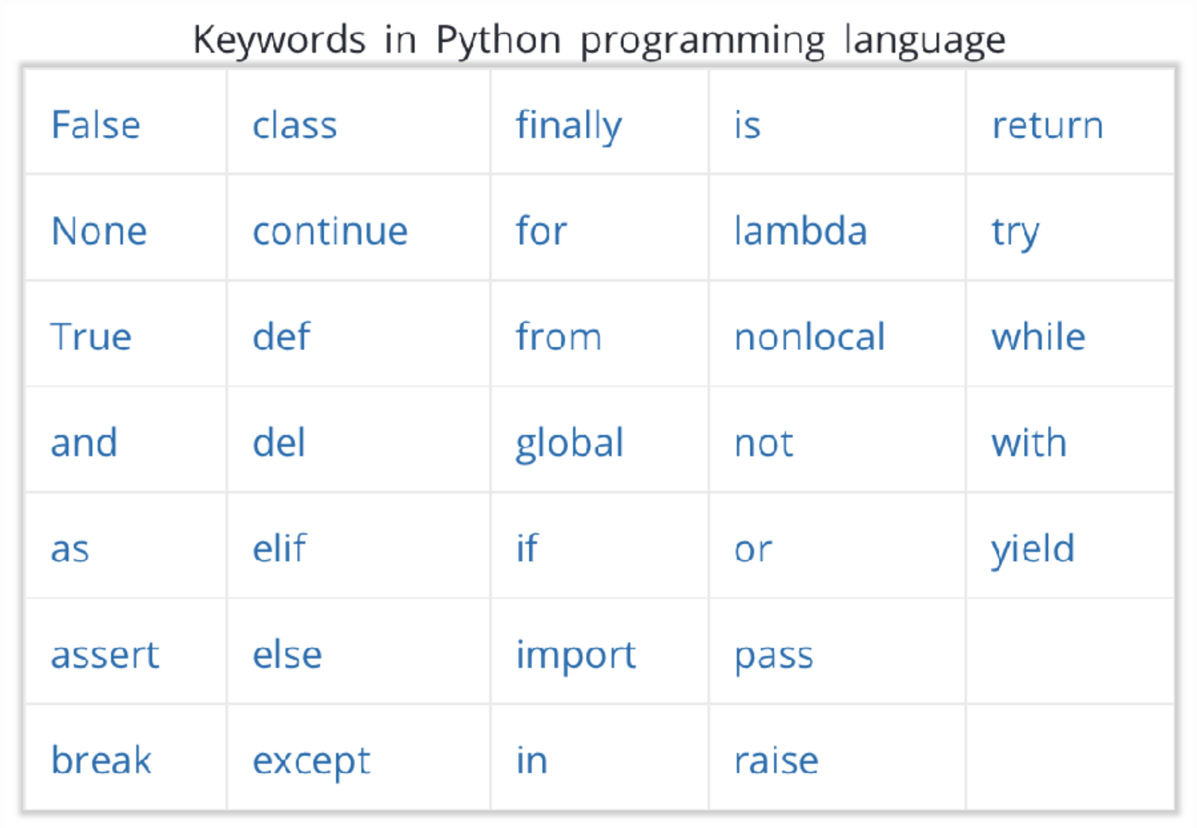
Variables

Video link

Pearson

- Starts with letter or underscore
  - `name`
  - `_name`
- Followed by letters, numbers, or underscores
  - `name_1`
- Case sensitive
  - `name_1`, `Name_1`, and `name1` are all different
- Readable and descriptive
  - `name` instead of `n`

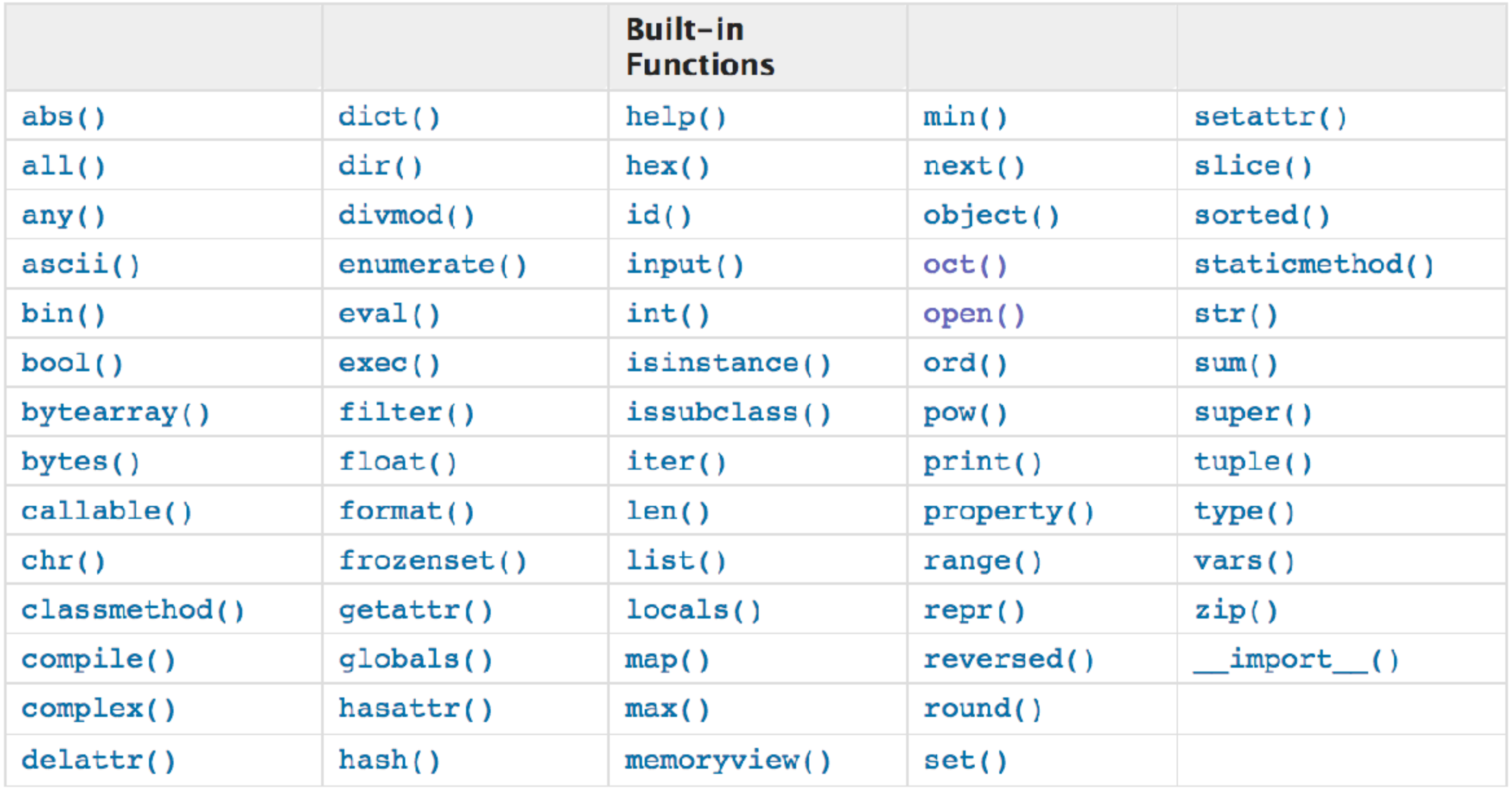
Pearson



Keywords in Python programming language

False	class	finally	is	return
None	continue	for	lambda	try
True	def	from	nonlocal	while
and	del	global	not	with
as	elif	if	or	yield
assert	else	import	pass	
break	except	in	raise	

https://www.programiz.com/python-programming/keyword-list



		Built-in Functions		
abs()	dict()	help()	min()	setattr()
all()	dir()	hex()	next()	slice()
any()	divmod()	id()	object()	sorted()
ascii()	enumerate()	input()	oct()	staticmethod()
bin()	eval()	int()	open()	str()
bool()	exec()	isinstance()	ord()	sum()
bytearray()	filter()	issubclass()	pow()	super()
bytes()	float()	iter()	print()	tuple()
callable()	format()	len()	property()	type()
chr()	frozenset()	list()	range()	vars()
classmethod()	getattr()	locals()	repr()	zip()
compile()	globals()	map()	reversed()	__import__()
complex()	hasattr()	max()	round()	
delattr()	hash()	memoryview()	set()	

https://docs.python.org/3.6/library/functions.html

Problem #2

Sing Happy Birthday

Happy birthday to you

Happy birthday to you

Happy birthday dear {name}

Happy birthday to you

Problem #3

Temperature converter

$$T(°C) = (T(°F) - 32) \times 5 / 9$$

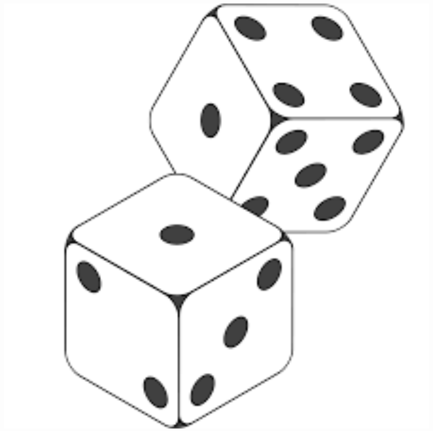
Errors

Video link

Pearson

Problem #4

Dice simulator



Libraries

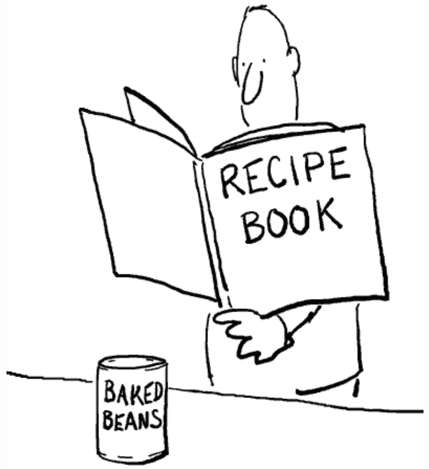
Video link

External libraries

- Use `pip` or PyCharm to install
- Search the Python Package Index (PyPI) https://pypi.org/
- In command line:
  - `pip install <package_name>`
- Video: Next Level Python - Lesson 3.2

Pearson

Functions



Video link

Problem #5

Circle stats

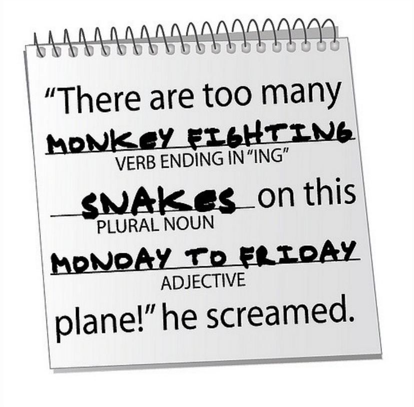
What is your circle's radius?

Methods

Video link

Challenge #1

Write a Mad Libs program



Video link

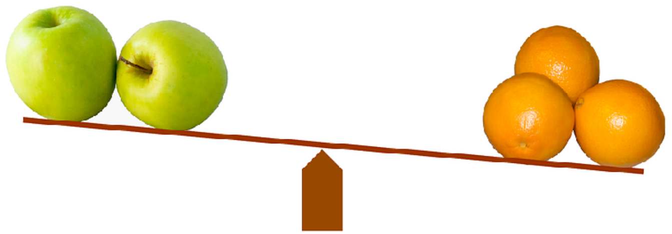
Fundamental concepts

- ☑ Variables
- ☑ Types
- ☑ Errors
- ☑ Functions and methods
- ☑ Libraries
- ☐ Comparisons
- ☐ Conditionals (if/else/elif)
- ☐ Looping (while/for)
- ☐ Data structures (list, dictionary, set, tuple)
- ☐ Exceptions

Control the flow with conditionals

What’s True? What’s False?

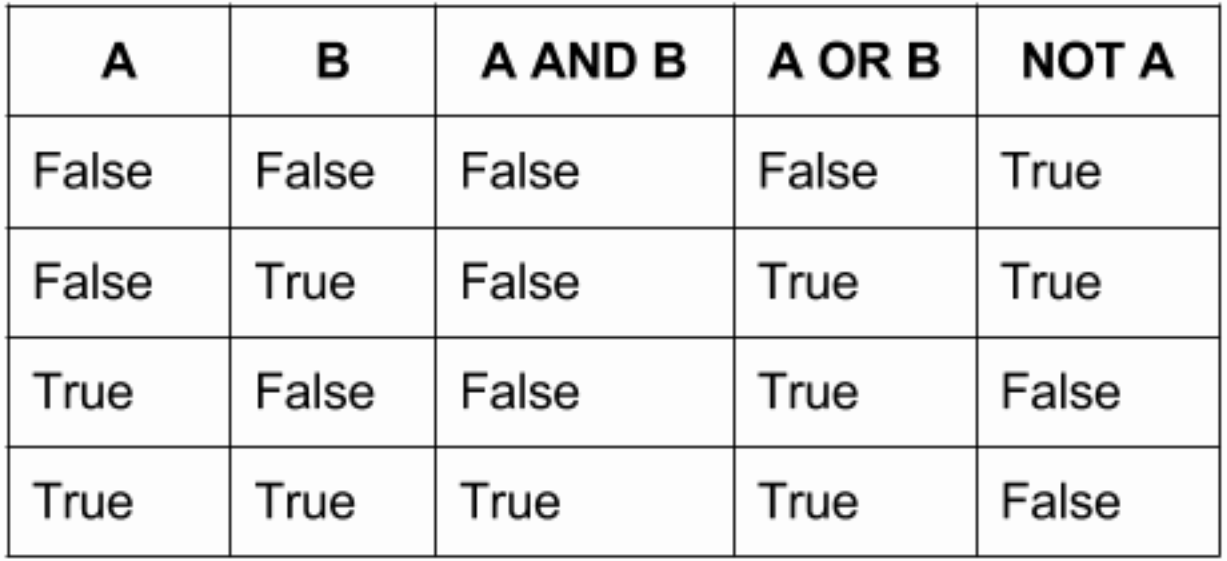
Comparisons



Video link

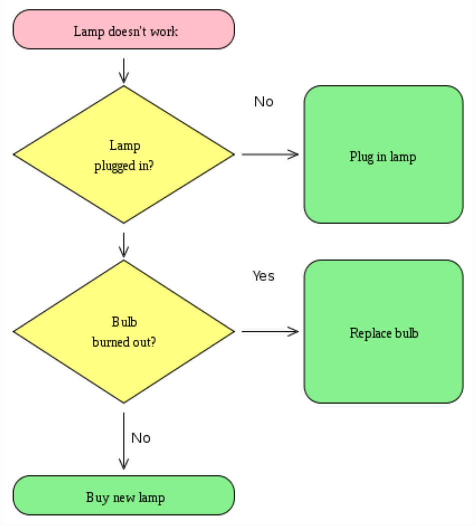
Pearson

Logical Operators



A	B	A AND B	A OR B	NOT A
False	False	False	False	True
False	True	False	True	True
True	False	False	True	False
True	True	True	True	False

Conditionals



Video link

Problem #6

Lucky guess

Guess a number between 1 and 10

Challenge #2

Number guessing game

Guess a number between 1 and 20

Tell them if the answer is higher or lower than their guess

Give the user 4 tries to get it right

Video link

Pearson

Fundamental concepts

- [x] Variables
- [x] Types
- [x] Errors
- [x] Functions and methods
- [x] Libraries
- [x] Comparisons
- [x] Conditionals (if/else/elif)
- [ ] Looping (while/for)
- [ ] Lists

Pearson

Work with lists and loops

Don’t repeat yourself!

While loops

Video link

Problem #7

New Year countdown

Happy New Year! 🎉

Challenge #2b

Number guessing game

Keep it DRY

Don't Repeat Yourself

Video link

Lists

Video link

For loops

Video link

Pearson

Problem #8

Vowel counter

Find the bug

Pearson

Problem #9

Deal a hand of 5 cards

Pearson

`while`

`for`

Challenge #3

Word guessing game

Create a list of words

Choose one at random

The user guesses the word, one letter at a time

They have 6 wrong guesses before they lose

Video link

Fundamental concepts

- ☑ Variables
- ☑ Types
- ☑ Errors
- ☑ Functions and methods
- ☑ Libraries
- ☑ Comparisons
- ☑ Conditionals (if/else/elif)
- ☑ Looping (while/for)
- ☑ Lists

What to learn next

Software engineering skills

- Handling exceptions
- Testing
- Debugging
- Refactoring

More data structures

- Dictionaries (`dict`)
- Sets
- Tuples

Intermediate Concepts

- Object-oriented programming
  - **Classes**
  - Inheritance

- Intermediate Python
  - **List comprehension**
  - **Nested functions**
  - Decorators
  - Lambda functions

Other essential skills

- Importing external libraries (`pip`)
- Creating virtual environments (`virtualenv` / `pipenv`)
- **Reading/writing to files**
- Making API requests

Word guess bonus solutions

- 3b: word_guess_from_file
  - Choose random word from a separate text file
- 3c: word_guess_with_nested_functions
  - Inner functions can use variables from outer function
- 3d: word_guess_with_classes
  - Creates a WordGame class and define methods on it
- 3e: word_guess_refactored
  - Uses list comprehension
  - Adds some validation so only 1 letter guesses are allowed
  - challenge_3e_word_guess_tests.py tests this version

Specialization

- Data Analysis (video link)
  - **Jupyter** notebooks, **Anaconda** distribution
  - **Pandas**, **NumPy** for manipulating data
  - **Matplotlib** or **Seaborn** for visualizations

- Web development (video link)
  - **Django** or **Flask** frameworks
  - API creation with **Django Rest Framework**, **Graphene**

- Scripting (Beyond the Basics live training)
  - Command line, bash
  - Web scraping with **beautiful-soup**, API requests with **requests**

How to learn them

- Tutorials
- Documentation
- **Books**
- **Live Trainings**
- **Videos**
- Courses
- Bootcamps

Next steps

- Project Euler math problems - https://projecteuler.net/
- Pick small projects that are appropriate for your level
  - More text-based games
  - Choose your own adventure stories
- Learn intermediate concepts
- Learn Object-Oriented programming (e.g. class)
  - GUIs for your programs (graphical user interface)
  - Make small games with PyGame - PDF tutorial

Recommended follow-up by me

Live Trainings

- Programming with Python: Beyond the Basics
- Python Environments and Best Practices

OR

- Hands-on Python Foundations in 3 Weeks

Videos

- Next Level Python LiveLessons

Books

- **Treading on Python Volume 1**: Foundations of Python
- **Python Crash Course**: A Hands-On, Project-Based Introduction to Programming
- **Automate the Boring Stuff with Python**: Practical Programming for Total Beginners
- **Learn Python the Hard Way**

Beginner Live Trainings by Arianne

- **Introduction to Python Programming**
  - Variables, functions, conditionals, lists, loops
  - Skill level – 1/10

- **Programming with Python: Beyond the Basics**
  - Dictionaries, exceptions, files, HTTP requests, web scraping
  - Skill level – 2/10

- **Python Environments and Best Practices**
  - Virtual envs, testing, debugging, PyCharm tips, git, modules
  - Skill level – 2/10

- **Hands-on Python Foundations in 3 Weeks**
  - Multi-week course that covers most of the above material
  - Skill level 1-3

Intermediate Live Trainings by Arianne

- **Object-Oriented Programming in Python**
  - Classes, dunder methods, and decorators
  - Skill level – 3/10
- **Python Data Structures and Comprehensions**
  - Overview of data structures from the standard library, Numpy and Pandas
  - Skill level – 3/10
- **Introduction to Django: a web application framework for Python**
  - Building web apps in Django – starting a project and high-level overview
  - Skill level – 4/10
- **Learn GraphQL in 4 Hours**
  - GraphQL APIs in Django and Node.js
  - Skill level – 5/10

Pearson

Video courses by Arianne

- **Introduction to Python LiveLessons**
  - Lessons 1-4 is the same content as this class
  - Lessons 5-7 is further content
  - Link
- **Next Level Python LiveLessons**
  - Setting up Python projects with virtual environments and git
  - More fundamentals (dictionaries, exceptions, file handling)
  - Testing, debugging, and understanding modules
  - Create a web scraper
  - Link
- **Rethinking REST: A hands-on guide to GraphQL and Queryable APIs**
  - Link

Thanks!

Questions?

Email me at

arianne.dee.studios@gmail.com

Pearson