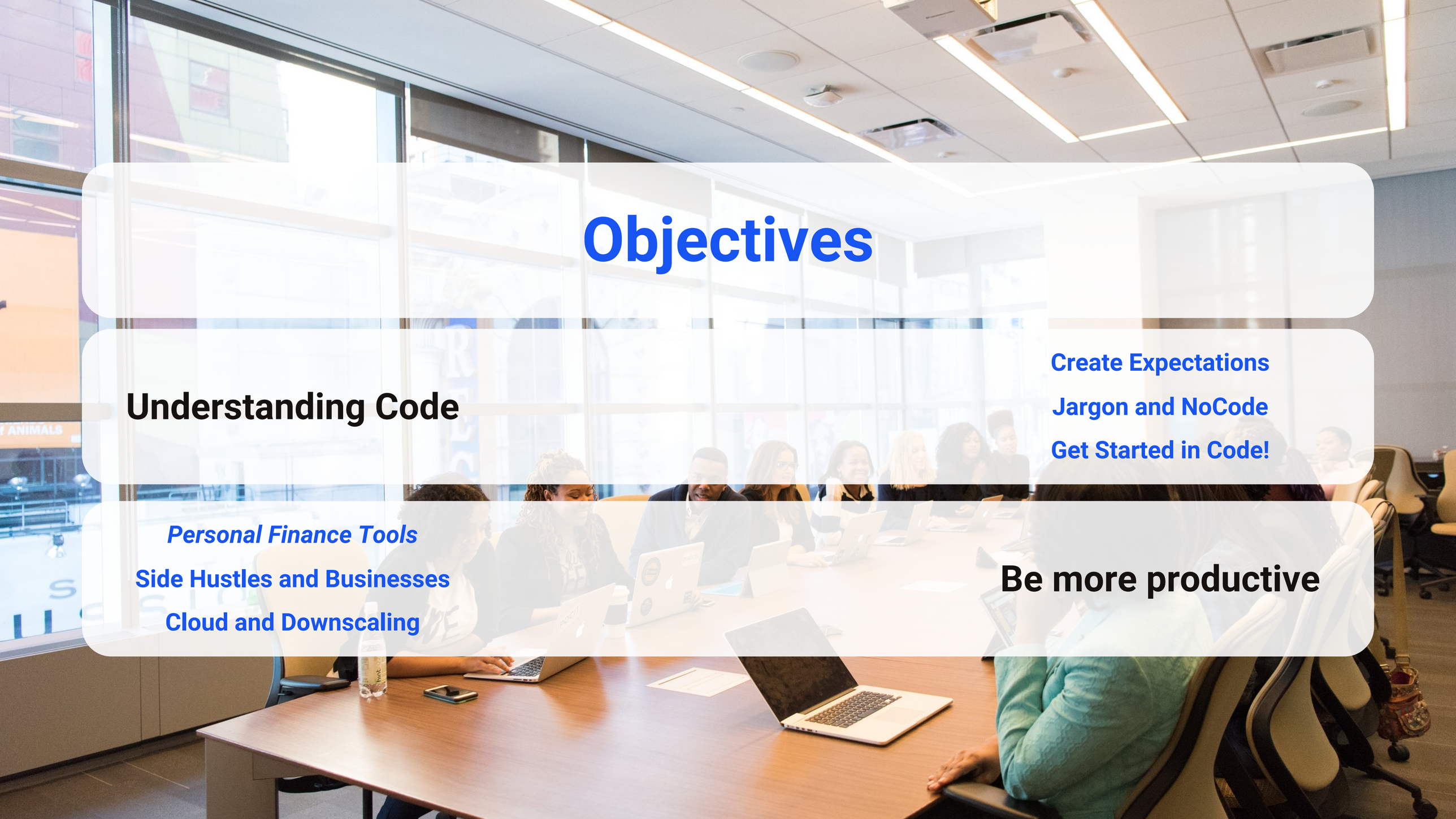
Python

Crack the Code !

An Introduction to python and Tech through Personal Finance Tools



Objectives

Understanding Code

Create Expectations

Jargon and NoCode

Get Started in Code!

Personal Finance Tools

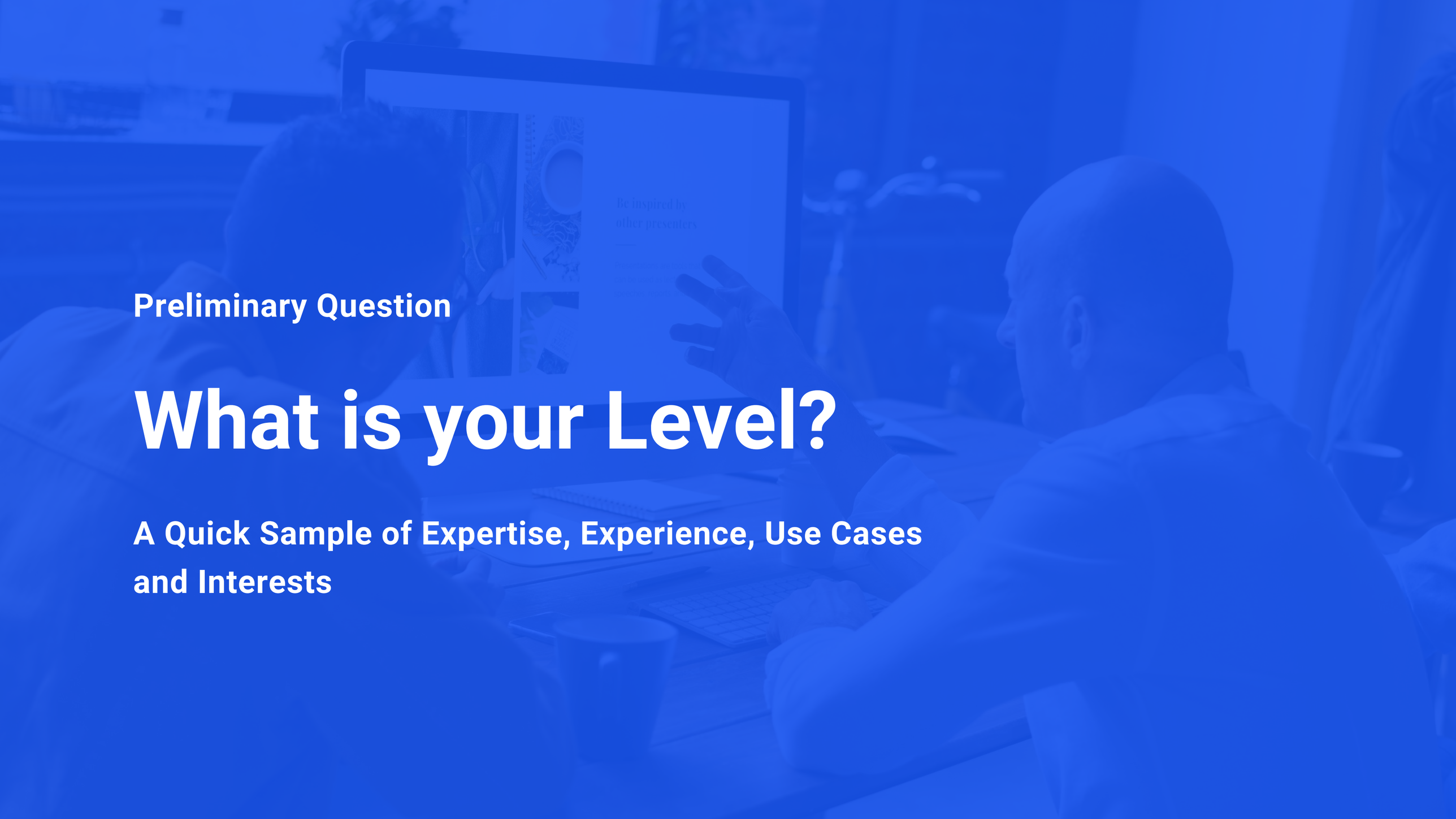
Side Hustles and Businesses

Cloud and Downscaling

Be more productive

Sessions

Session 1	Situating Python, Basics, Control Flow
Session 2	Loops, Data Structures and NumPy
Session 3	SciPy, Pandas, MatPlotLib and Prep
Session 4	Portfolio Design and Automation

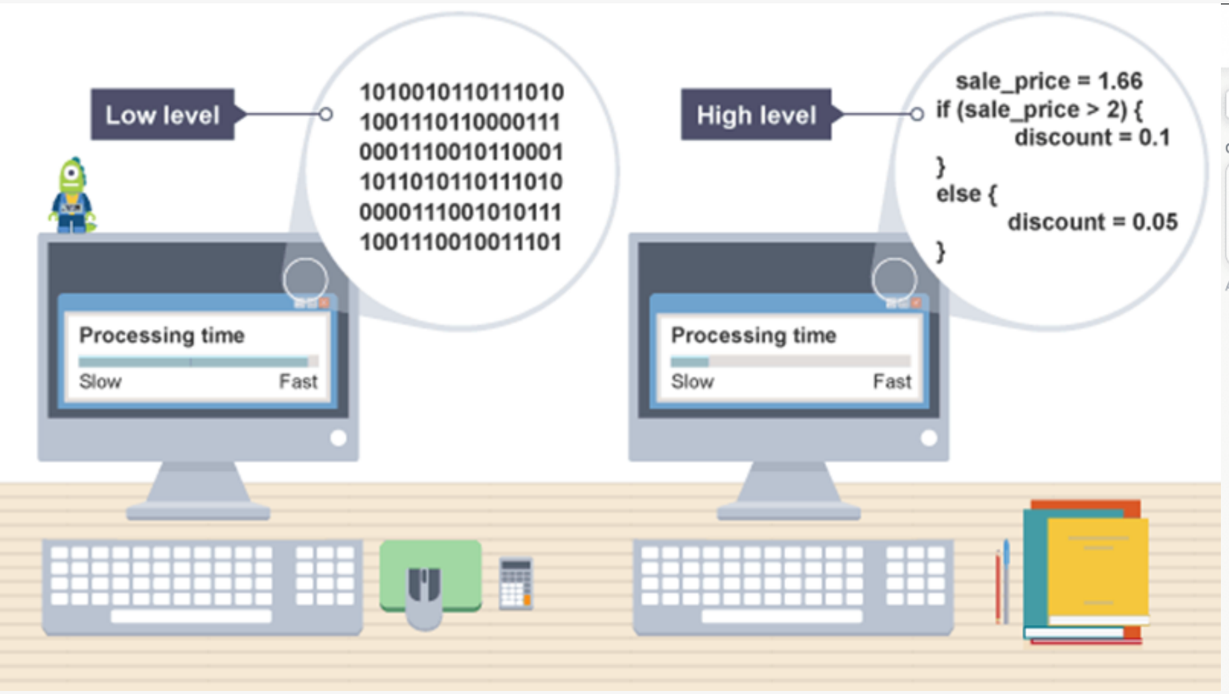


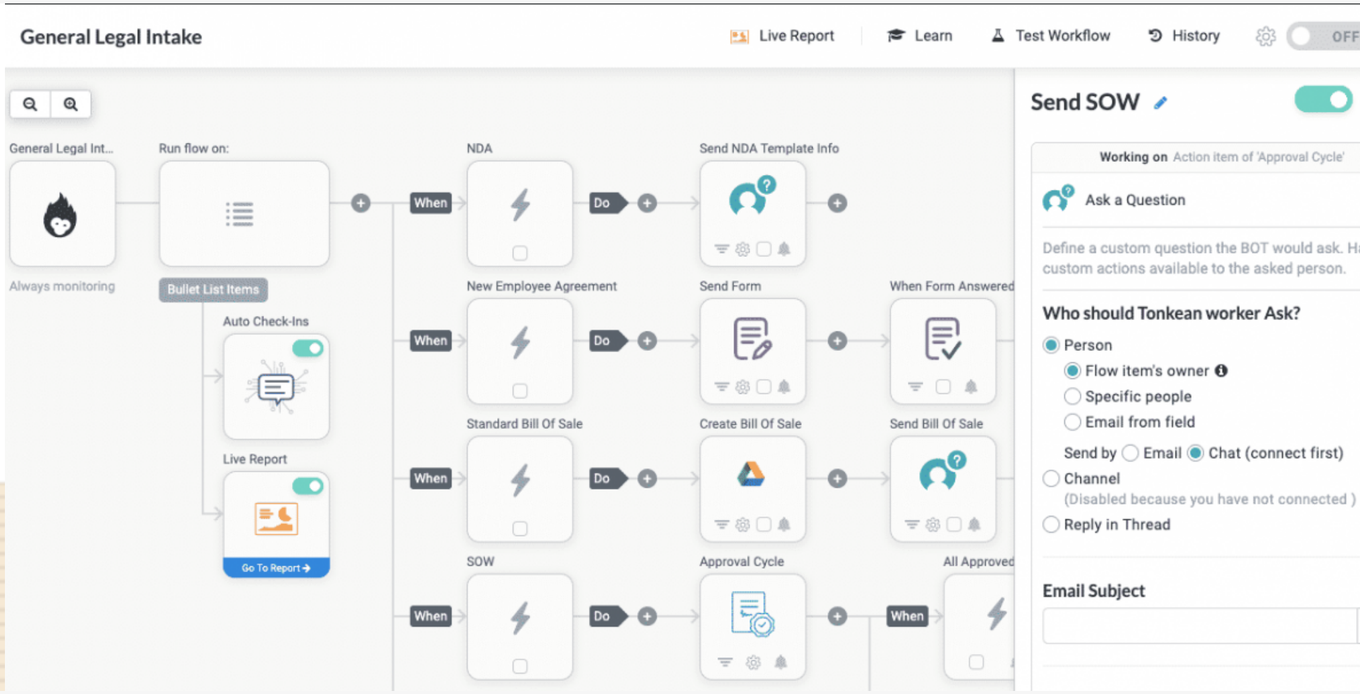
Preliminary Question

What is your Level?

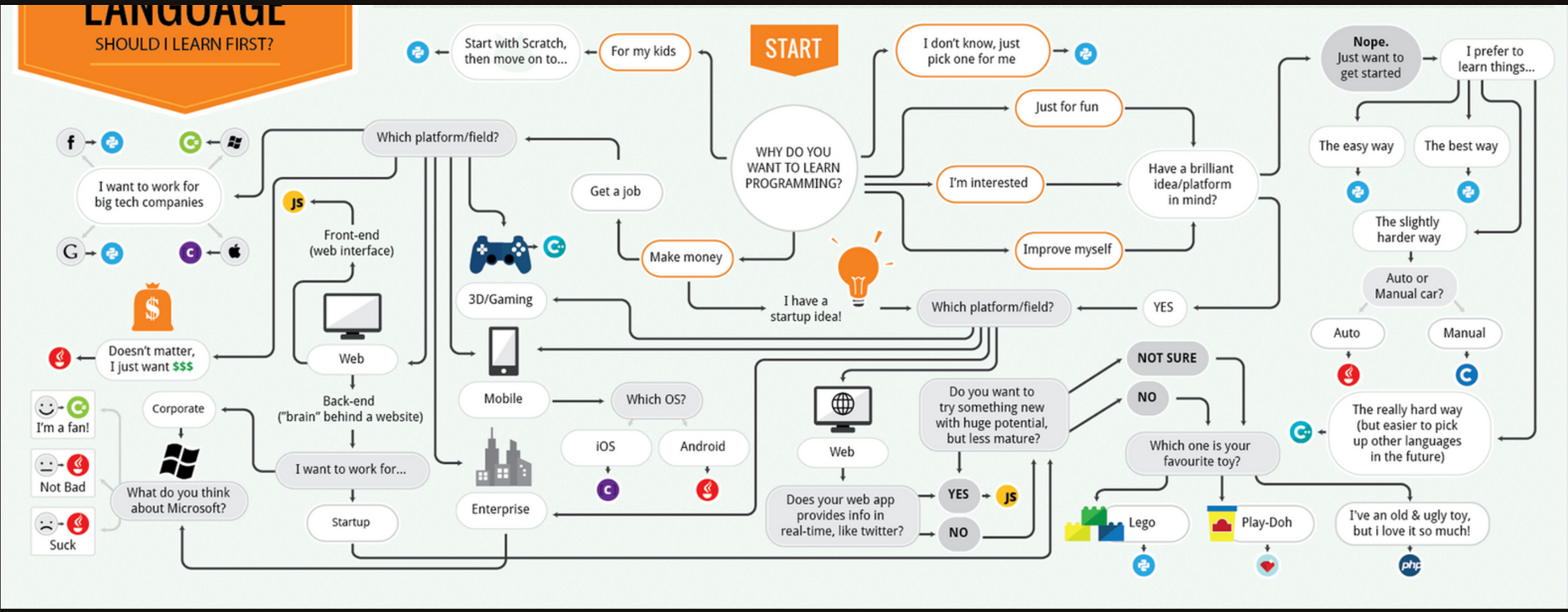
A Quick Sample of Expertise, Experience, Use Cases and Interests

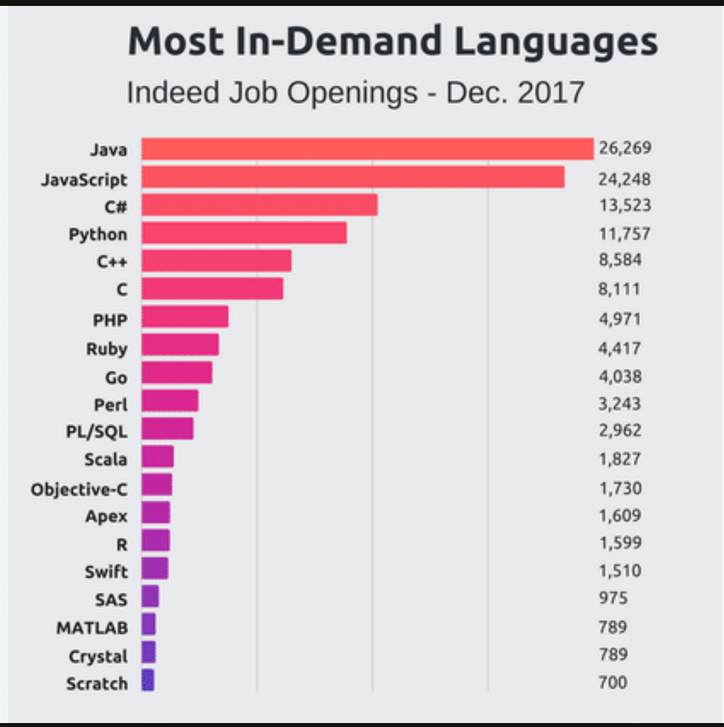
Language of Choice, A Recommendation and the future



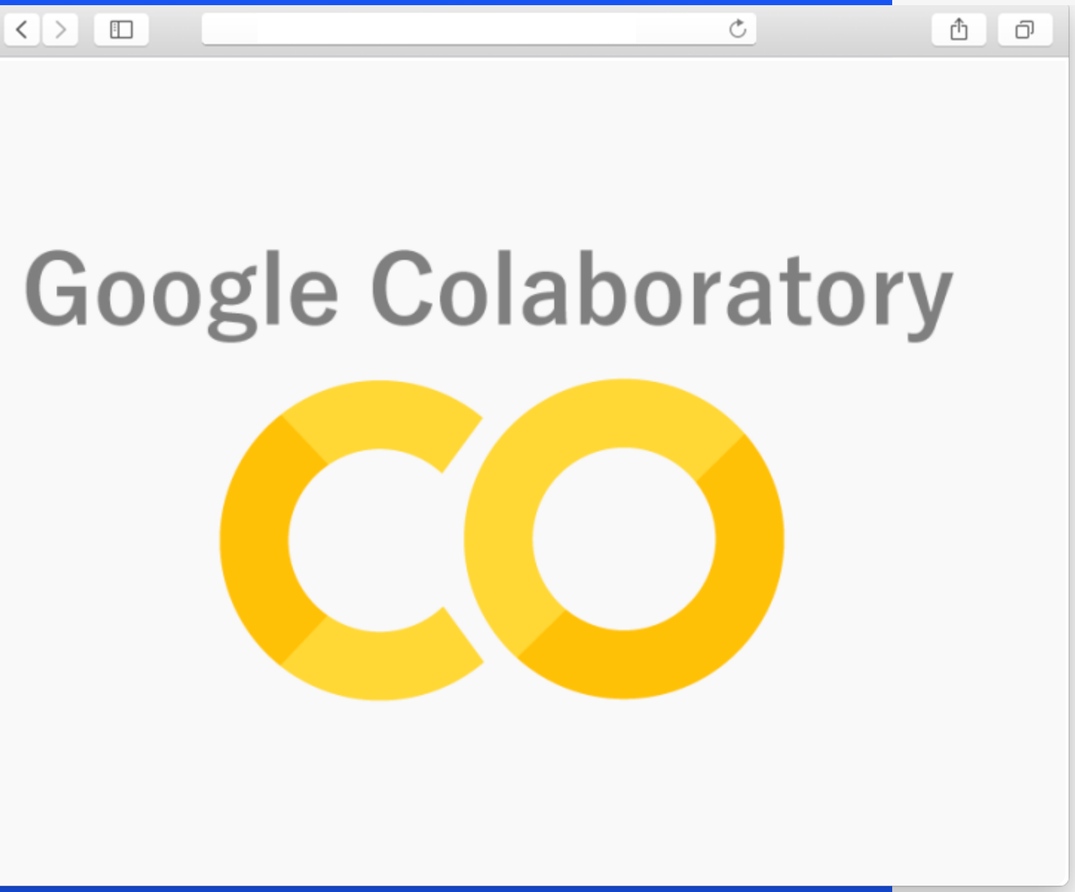


The Right tool for the Job





Google Colaboratory



Cloud Notebooks - Text and Code

Web-based interface to a document that contains runnable code, visualizations, and narrative text - Experimentation and Exploration - VSCode!.

Google's hosted runtime - Resources

Cloud Resources and Virtualisation. This is why any investment in hardware especially for code should be seen skeptically.

Let's Check it out!

www.colab.research.google.com
For those more advanced - research.google.com - Tools&Download - ML!

Coding and the Computer

A Dumb, Obedient Worker

Comes from the history of computers as mathematicians, The idea being adding machines. If you can do it by hand you can get something to program it.

*The computer listens to you, does **exactly** what you ask it to – be careful of your wording.*

Basics of Code - Math

+, -, /, *

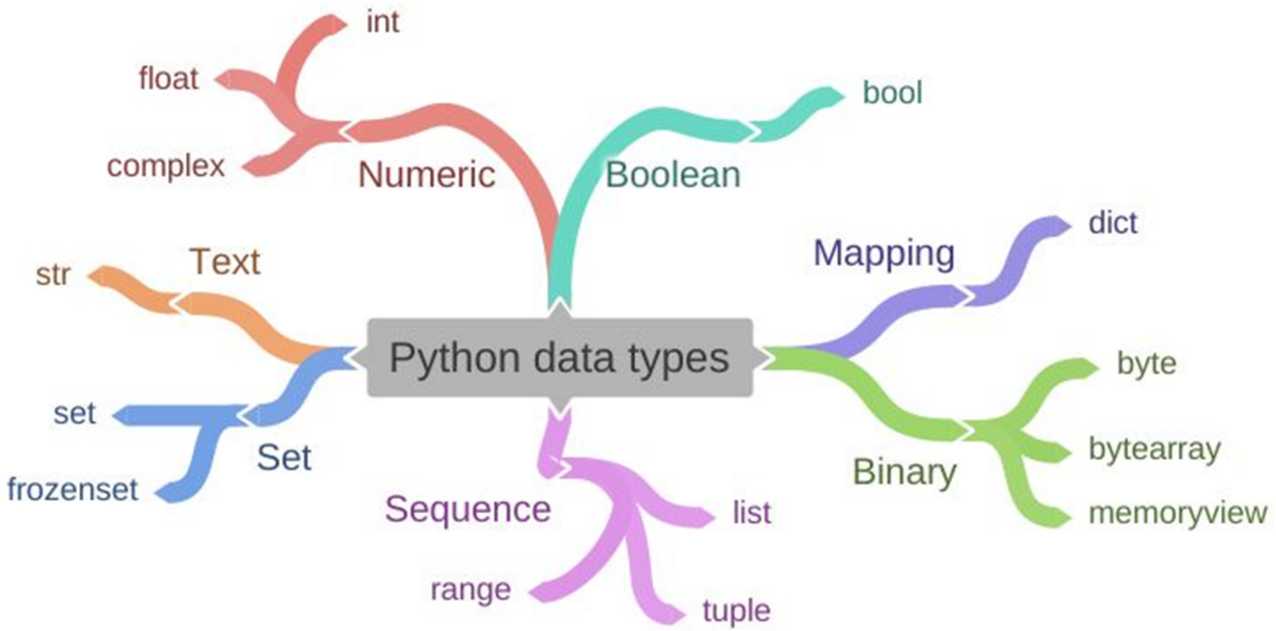
Special Signs %, **, //

Brackets work - *(PEMDAS)*

Decimal points work!

Lets try it out!

Blitz-Time!



Basics of Code - Data Types

My Coders:

Input Optimization

Input Validation

Basics of Code - Text

Different Data Types Give Different Methods

“ ” <- Text also ' '

Returns and prints – Hold

A Programming Rite of Tradition -> Hello World

Lets go over them!

Exercise - Basic

Create a program that displays your name and complete mailing address formatted in the manner that you would usually see it on the outside of an envelope. Your program does not need to read any input from the user.

Variables!

Basis of Programming (consts)

Something whose value changes

A Box with a Label holding Whatever

Some Rules to prevent Uncertainty

Why are they needed?

Let's See!

Exercise - Basic

```
a = 1 + 2 * 4

b = 1 + 1

c = a * b
```

Exercise - Basic

```
a = 2

b = a * 2

c = a + 1

d = a + b + c + 1
```

Basics of Code - Functions

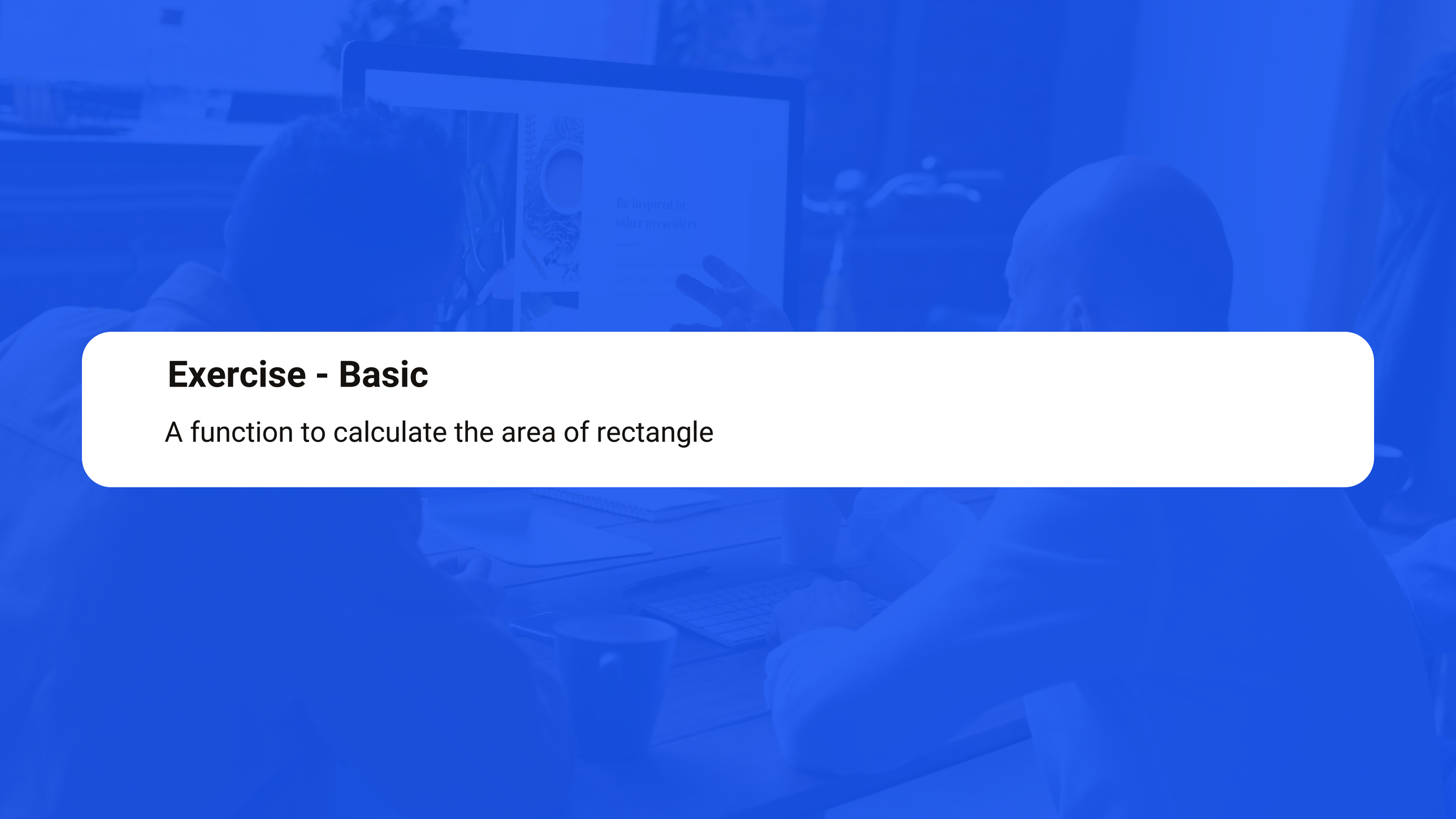
Algorithms

Functions (Rule of 3):
- Take This - Arguments
- Do that - Script
- Give it Back - Returns

Every function needs a name

"I need you to do __x" -> def

Let's try!



Exercise - Basic

A function to calculate the area of rectangle

Exercise - Basic

A function to convert temperature from Celsius to Fahrenheit

Basics of Code - Input

Input()

A function like any other - Prompts

Let's try It Out!

Basics of Code - Input

Input()

A function like any other - Prompts

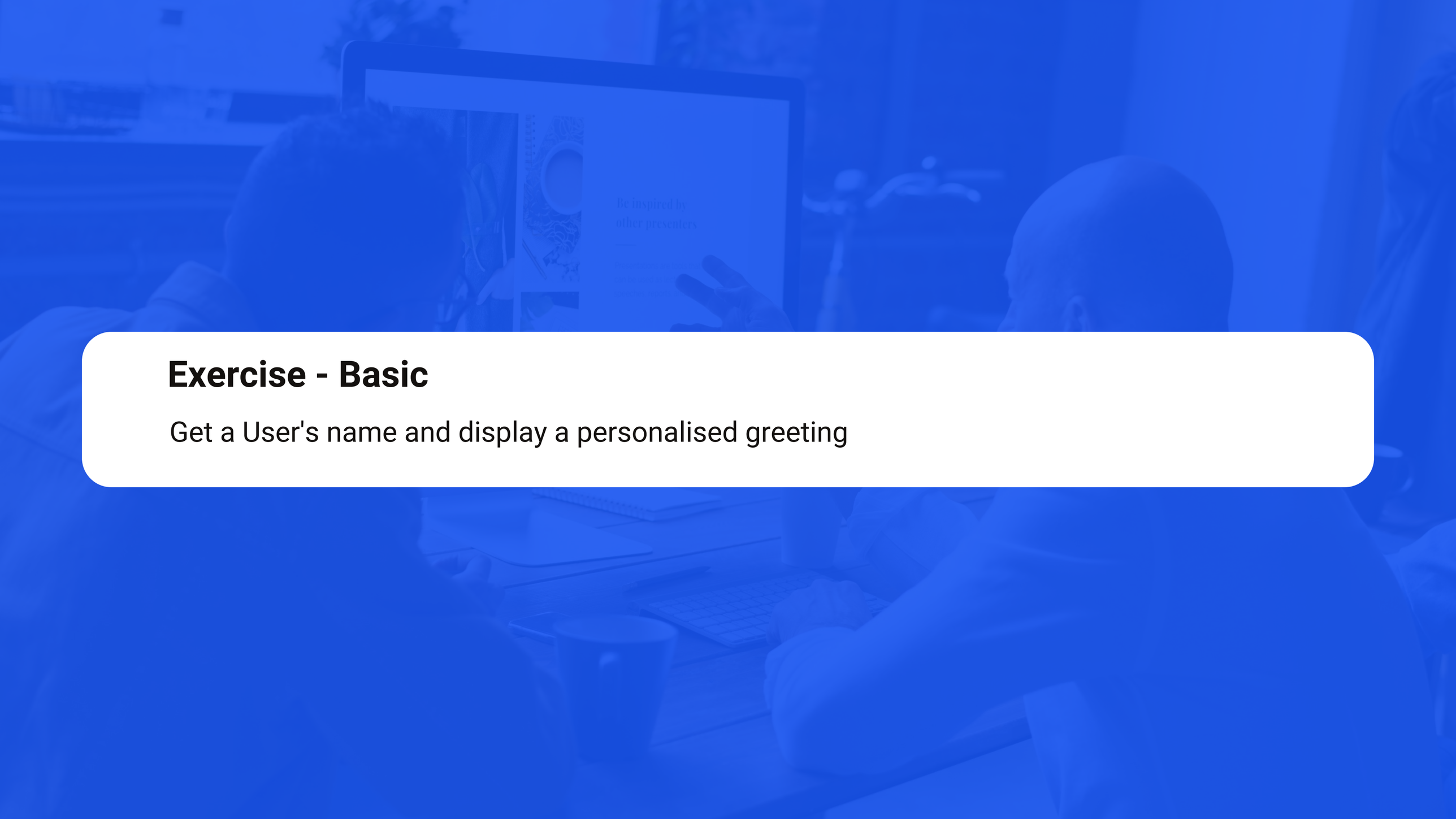
Let's try It Out!

Problem is User Types you keep Text - Numbers?

Data Types - The story of coercion

- *Str*
- *Int*
- *Float*
- *Bool*

Let's try It Out!



Exercise - Basic

Get a User's name and display a personalised greeting

Exercise - Advanced

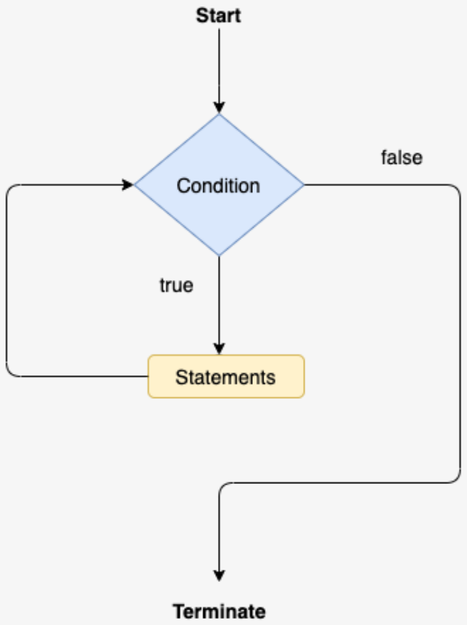
Pretend that you have just opened a new savings account that earns 4 percent interest per year. The interest that you earn is paid at the end of the year, and is added to the balance of the savings account. Write a program that begins by reading the amount of money deposited into the account from the user. Then your program should compute and display the amount in the savings account after 1, 2, and 3 years.

Exercise - Advanced

Consider the software that runs on a self-checkout machine. One task that it must be able to perform is to determine how much change to provide when the shopper pays for a purchase with cash.

Write a program that begins by reading a number of cents from the user as an integer. Then your program should compute and display the denominations of the coins that should be used to give that amount of change to the shopper. The change should be given using as few coins as possible. Assume that the machine is loaded with pennies, nickels, dimes, quarters

Basics of Code - Control



Allows your program to do "different" things

If - Condition - Literally "**If**"

If - Condition - Else - **Otherwise**

If - Condition - Elif - condition - **This or That**

If - Condition - Else - Elif - **These things in these cases.**

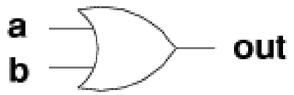
Let's try It Out!

Basics of Code - Conditions

AND

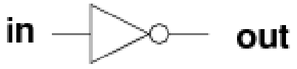
a	b	out
0	0	0
0	1	0
1	0	0
1	1	1

OR



a	b	out
0	0	0
0	1	1
1	0	1
1	1	1

NOT



in	out
0	1
1	0

Basic Comparisons - >,<, ==,>=, <=

Note: "in"

More Complex Comparisons -

AND, OR, NOT

Let's try It Out!

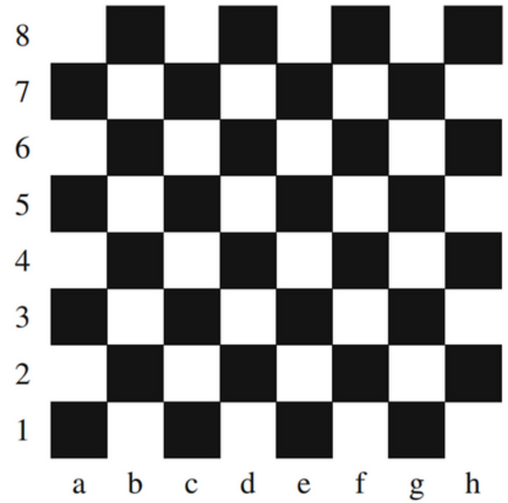
Exercise - Advanced

It is commonly said that one human year is equivalent to 7 dog years. However this simple conversion fails to recognize that dogs reach adulthood in approximately two years. As a result, some people believe that it is better to count each of the first two human years as 10.5 dog years, and then count each additional human year as 4 dog years.

Write a program that implements the conversion from human years to dog years described in the previous paragraph. Ensure that your program works correctly for conversions of less than two human years and for conversions of two or more human years. Your program should display an appropriate error message if the user enters a negative number.

Exercise - Advanced

Positions on a chess board are identified by a letter and a number. The letter identifies the column, while the number identifies the row, as shown below:



Write a program that reads a position from the user. Use an if statement to determine if the column begins with a black square or a white square. Then use modular arithmetic to report the color of the square in that row. For example, if the user enters `a1` then your program should report that the square is black. If the user enters `d5` then your program should report that the square is white. Your program may assume that a valid position will always be entered. It does not need to perform any error checking.

Exercise - Advanced

A roulette wheel has 38 spaces on it. Of these spaces, 18 are black, 18 are red, and two are green. The green spaces are numbered 0 and 00. The red spaces are numbered 1, 3, 5, 7, 9, 12, 14, 16, 18, 19, 21, 23, 25, 27, 30 32, 34 and 36. The remaining integers between 1 and 36 are used to number the black spaces.

Many different bets can be placed in roulette. We will only consider the following subset of them in this exercise:

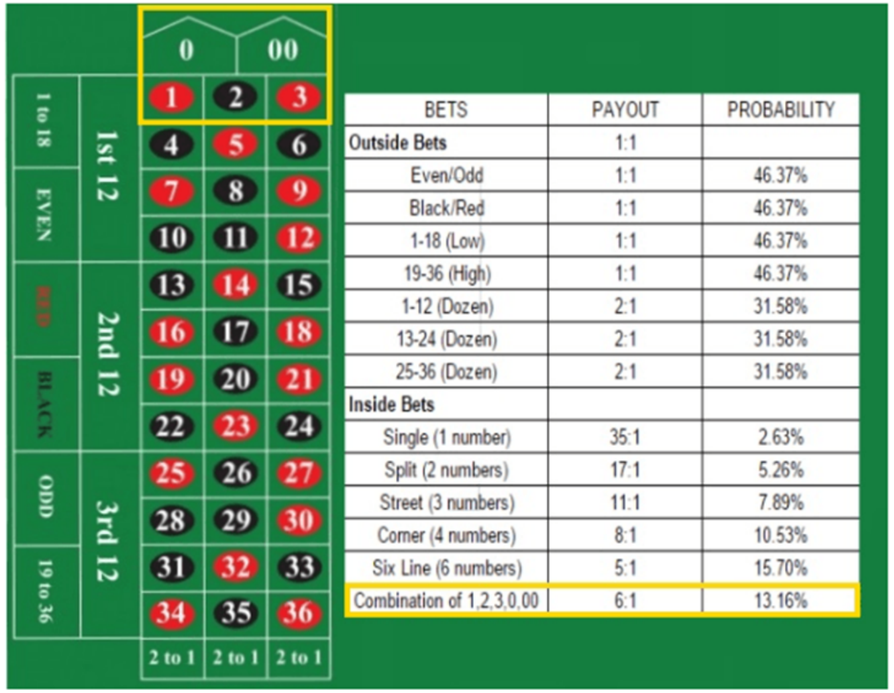
- Single number (1 to 36, 0, or 00)
- Red versus Black
- Odd versus Even (Note that 0 and 00 do **not** pay out for even)
- 1 to 18 versus 19 to 36

```
The spin resulted in 13...
Pay 13
Pay Black
Pay Odd
Pay 1 to 18
```

If the simulation results in 0 or 00 then your program should display `Pay 0` or `Pay 00` without any further output.

Exercise - Super Advanced!

- We'll see this later:
  - Import Math
  - Math.Random()



BETS	PAYOUT	PROBABILITY
Outside Bets	1:1	
Even/Odd	1:1	46.37%
Black/Red	1:1	46.37%
1-18 (Low)	1:1	46.37%
19-36 (High)	1:1	46.37%
1-12 (Dozen)	2:1	31.58%
13-24 (Dozen)	2:1	31.58%
25-36 (Dozen)	2:1	31.58%
Inside Bets		
Single (1 number)	35:1	2.63%
Split (2 numbers)	17:1	5.26%
Street (3 numbers)	11:1	7.89%
Corner (4 numbers)	8:1	10.53%
Six Line (6 numbers)	5:1	15.70%
Combination of 1,2,3,0,00	6:1	13.16%

A roulette wheel has 38 spaces on it. Of these spaces, 18 are black, 18 are red, and two are green. The green spaces are numbered 0 and 00. The red spaces are numbered 1, 3, 5, 7, 9, 12, 14, 16, 18, 19, 21, 23, 25, 27, 30 32, 34 and 36. The remaining integers between 1 and 36 are used to number the black spaces.

Many different bets can be placed in roulette. We will only consider the following subset of them in this exercise:

- Single number (1 to 36, 0, or 00)
- Red versus Black
- Odd versus Even (Note that 0 and 00 do **not** pay out for even)
- 1 to 18 versus 19 to 36

Write a program that simulates a spin of a roulette wheel by using Python's random number generator. Display the number that was selected and all of the bets that must be payed. For example, if 13 is selected then your program should display:

```
The spin resulted in 13...
Pay 13
Pay Black
Pay Odd
Pay 1 to 18
```

If the simulation results in 0 or 00 then your program should display `Pay 0` or `Pay 00` without any further output.

Functional Roulette!