



KTP Coding Workshop

Coding in Python: Basics



Schedule

01

Overview

02

**Jupyter
Notebooks**

03

Basic Syntax

04


Basic Functions

05

**Basics of
Control Flow**

06

Basics of Loops





01

Overview



Why is Python useful?

- **Easy to Learn**
 - ◆ **Syntax resembles natural language - easy to pick up and identify errors**
- **Many libraries to do many different things with**
 - ◆ **Data visualization, machine learning, web development and more**
- **Large community of users**
 - ◆ **Lots of resources to aid in learning and solution assistance**
- **Can handle large datasets**



02

Jupyter Notebooks



So How Do You Use Python?

- **Jupyter Notebooks**
 - **Open-source web-based application**
- **Google Colab**
 - **Free, cloud-based platform**
 - **You can run Python code in a Jupyter Notebook through Colab**



How to Open Google Colab

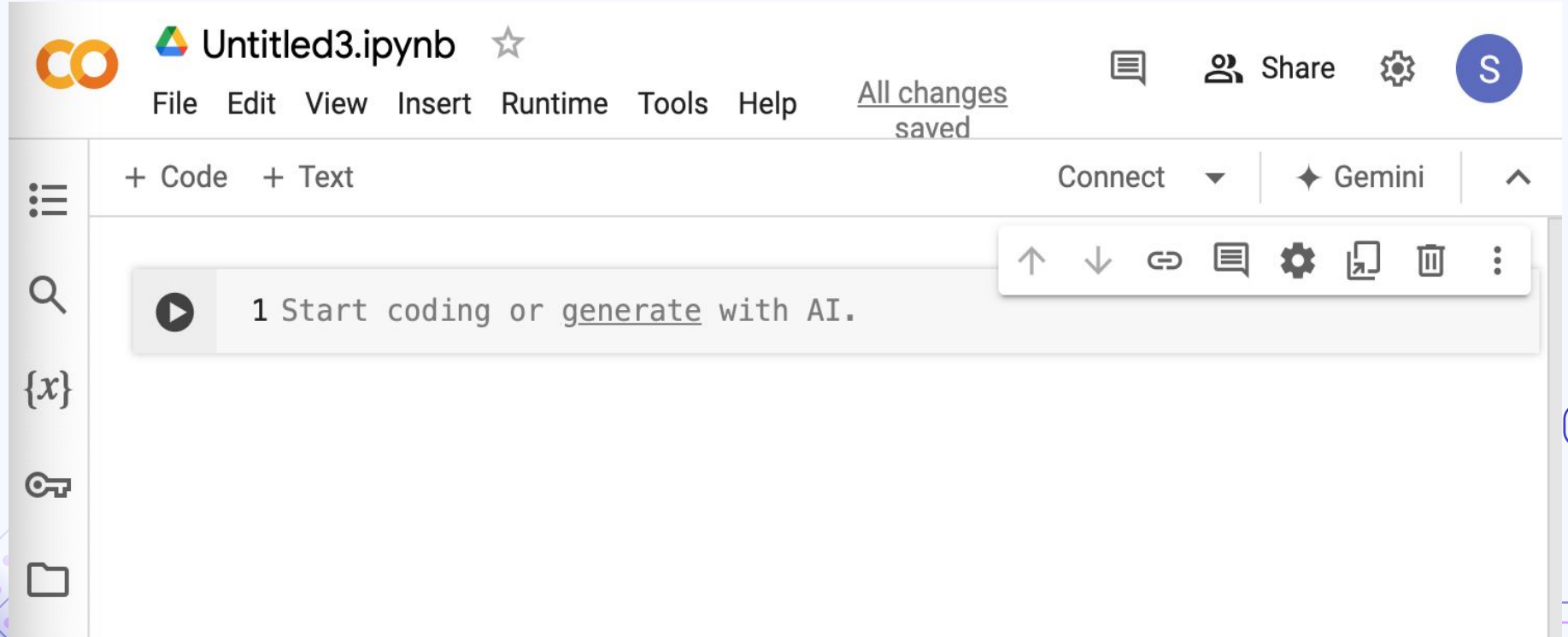
Google Colaboratory

Colab is a hosted Jupyter Notebook service that requires no setup to use and provides free access to computing resources, including GPUs and TPUs. Colab is especially well suited to machine learning, data science, and education.

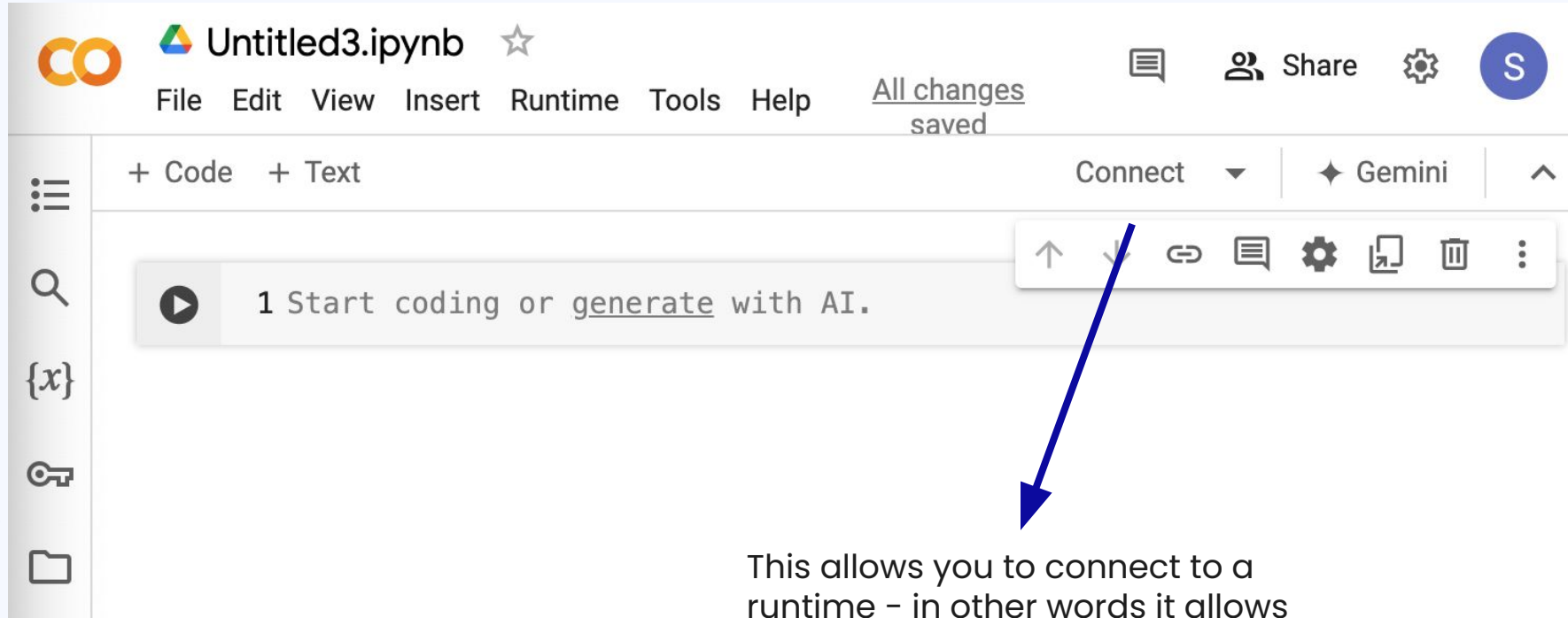
[Open Colab](#)

[New Notebook](#)

Should Look Like This Now



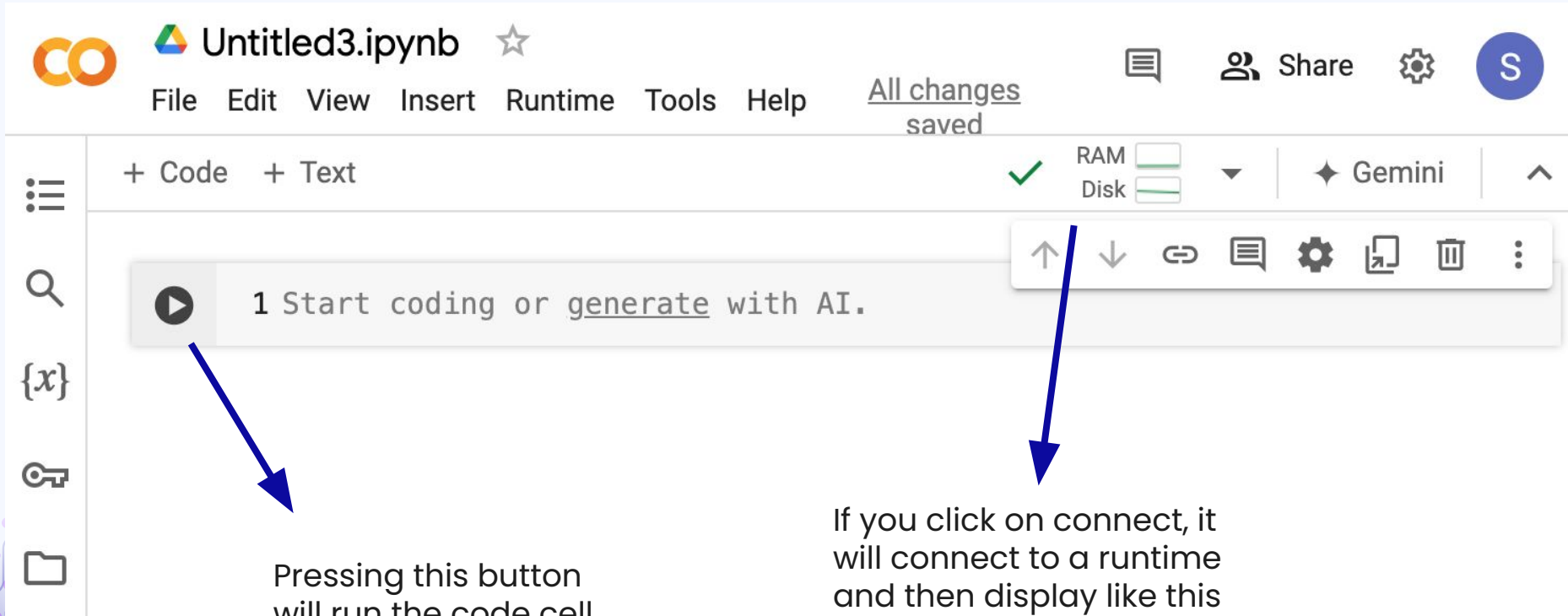
What Does Everything Mean?



The screenshot shows the Google Colab web interface. At the top, there's a header with the Colab logo, the file name 'Untitled3.ipynb', and a star icon. Below this is a menu bar with 'File', 'Edit', 'View', 'Insert', 'Runtime', 'Tools', and 'Help'. To the right of the menu bar, it says 'All changes saved'. Further right are icons for chat, share, settings, and a user profile. Below the menu bar, there's a toolbar with '+ Code' and '+ Text' buttons. To the right of these are 'Connect', 'Gemini', and an expand/collapse icon. The main code editor area contains a single cell with the text '1 Start coding or generate with AI.' and a play button icon. A blue arrow points from the 'Connect' button in the toolbar to the explanatory text below.

This allows you to connect to a runtime - in other words it allows your code to process!

What Does Everything Mean?



The screenshot displays the Google Colab web interface. At the top, the title bar shows 'Untitled3.ipynb' with a star icon, a 'Share' button, and a settings gear. Below this is a menu bar with 'File', 'Edit', 'View', 'Insert', 'Runtime', 'Tools', and 'Help'. A status indicator 'All changes saved' is visible. The main workspace contains a code cell with the text '1 Start coding or generate with AI.' and a play button icon. A blue arrow points from the play button to the text 'Pressing this button will run the code cell'. Another blue arrow points from the 'connect' button (upward arrow) in the cell's toolbar to the text 'If you click on connect, it will connect to a runtime and then display like this - now your code can run'. The toolbar also includes buttons for download, link, comment, settings, copy, and delete.

CO Untitled3.ipynb ☆

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

✓ RAM [] Disk [] Gemini ^

↑ ↓ 🔗 💬 ⚙️ 📄 🗑️ ⋮

▶ 1 Start coding or generate with AI.

Pressing this button will run the code cell

If you click on connect, it will connect to a runtime and then display like this - now your code can run

Creating Comments

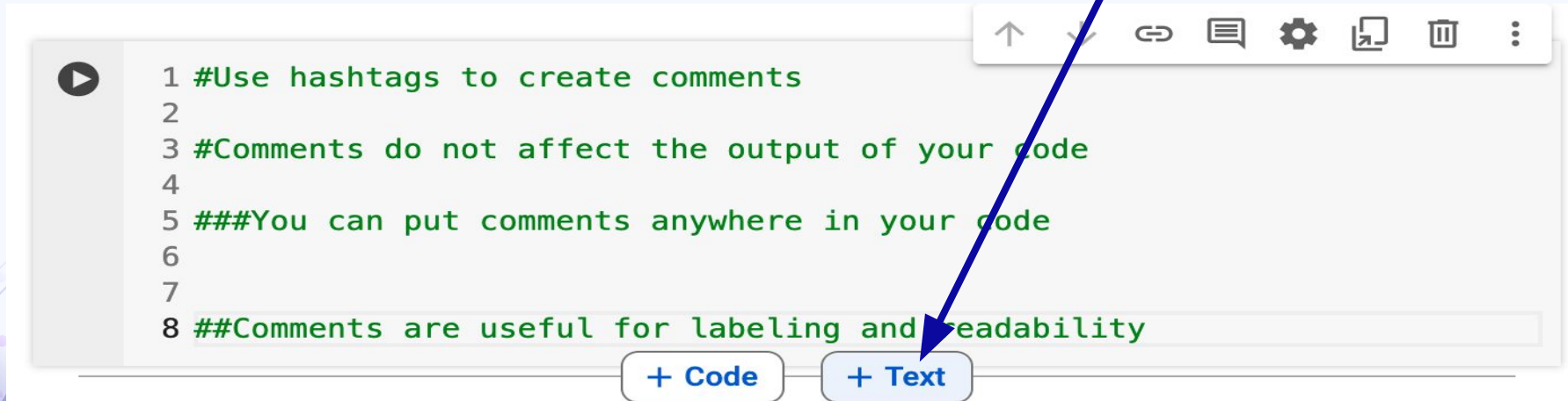


```
1 #Use hashtags to create comments
2
3 #Comments do not affect the output of your code
4
5 ###You can put comments anywhere in your code
6
7
8 ##Comments are useful for labeling and readability
```



Headers and Text Boxes

- To create a text cell
 - You can use Command MM (on a mac)
 - Hover near the top or bottom of your current cell and then click "Text"

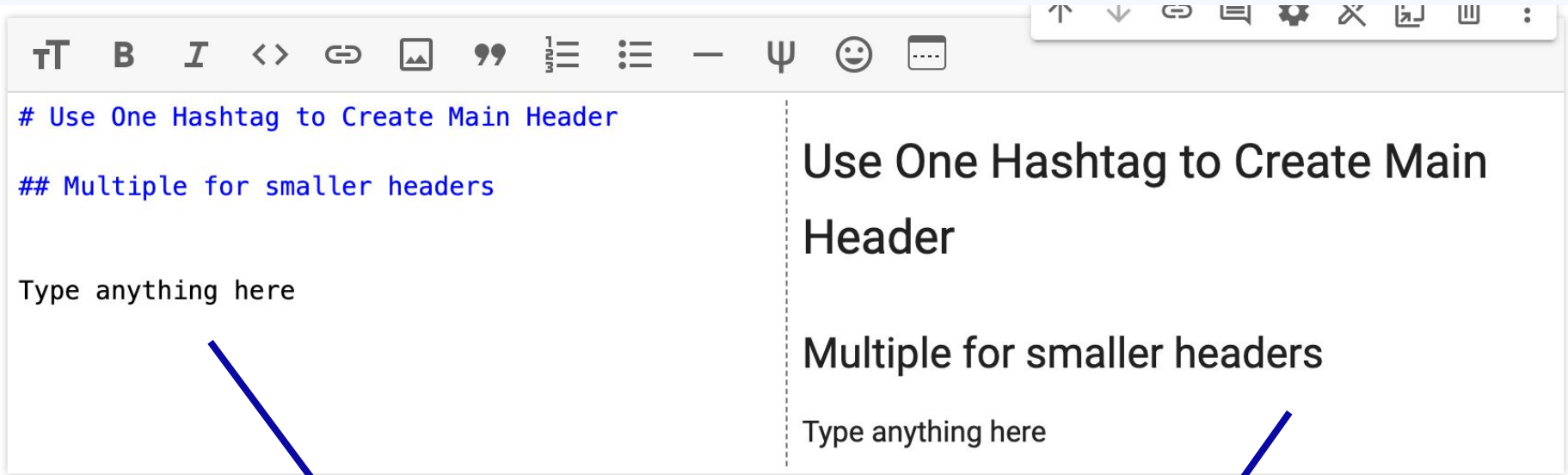


The screenshot shows a Jupyter Notebook interface. A code cell contains the following text:

```
1 #Use hashtags to create comments
2
3 #Comments do not affect the output of your code
4
5 ###You can put comments anywhere in your code
6
7
8 ##Comments are useful for labeling and readability
```

At the top right of the cell, there is a toolbar with icons for up, down, link, comment, settings, copy, delete, and a menu. A blue arrow points from the text "Add Text Box" to the "+ Text" button at the bottom right of the cell. The "+ Code" button is also visible next to it.

Headers and Text Boxes



The image shows a screenshot of a rich text editor interface. The top toolbar contains icons for text formatting (bold, italic, underline), linking, image insertion, quote, list creation, indentation, undo, redo, and a smiley face. The editor is split into two panes by a vertical dashed line. The left pane, labeled 'This is your code' with a blue arrow, contains the following text: `# Use One Hashtag to Create Main Header`, `## Multiple for smaller headers`, and `Type anything here`. The right pane, labeled 'This is the output' with a blue arrow, shows the rendered result: a large heading 'Use One Hashtag to Create Main Header', a smaller heading 'Multiple for smaller headers', and the text 'Type anything here'.

Use One Hashtag to Create Main Header

Multiple for smaller headers

Type anything here

Use One Hashtag to Create Main Header

Multiple for smaller headers

Type anything here

This is your code

This is the output



03

Basic Syntax



Data Types

int (integer)

Represents integer values
(whole numbers)

3, -5, 0 are all integer
values

float values

Represents a floating point
number (decimal)

31.0, -0.987, 45.24

str (string)

Represents text (string)

Enclosed with quotes - single or double

"Hello", 'Python is Cool', "3 is the best
number"

boolean

True or False value

Connected to comparison
operators: <, >, !=, ==

More on Boolean Values

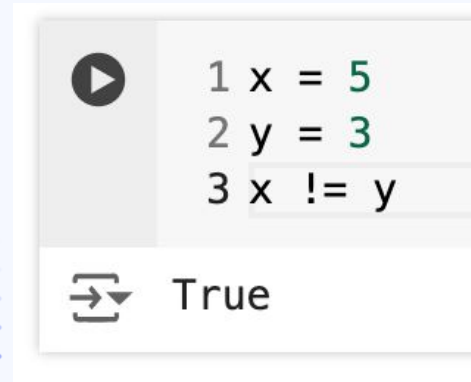
→ Syntax

- ◆ True
- ◆ False

→ Comparison Operators

- ◆ != means does not equal
- ◆ == means equal

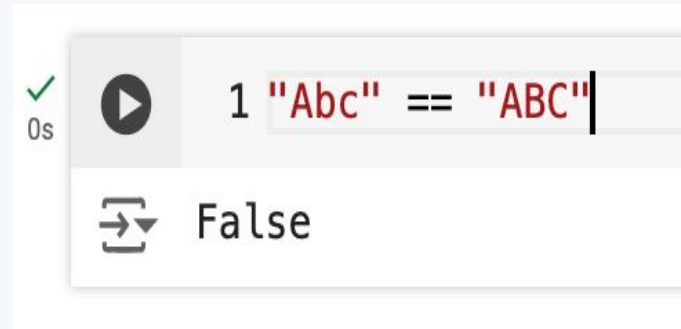
→ You can compare string, integers and float values



```
1 x = 5
2 y = 3
3 x != y
```

True

A Python REPL snippet showing three lines of code: `1 x = 5`, `2 y = 3`, and `3 x != y`. Below the code, the output `True` is displayed, indicating that the comparison `x != y` is true.



```
1 "Abc" == "ABC"
```

False

A Python REPL snippet showing a single line of code: `1 "Abc" == "ABC"`. Below the code, the output `False` is displayed, indicating that the comparison `"Abc" == "ABC"` is false. A green checkmark and the text `0s` are visible to the left of the code line.

Using 'And' and 'Or' To Compare

→ And

- ◆ True and True = True
- ◆ True and False = False
- ◆ False and False = False

→ Or

- ◆ True or True = True
- ◆ True or False = True
- ◆ False or False = False

Naming Variables

→ Assign Variables using '='

◆ Ex:

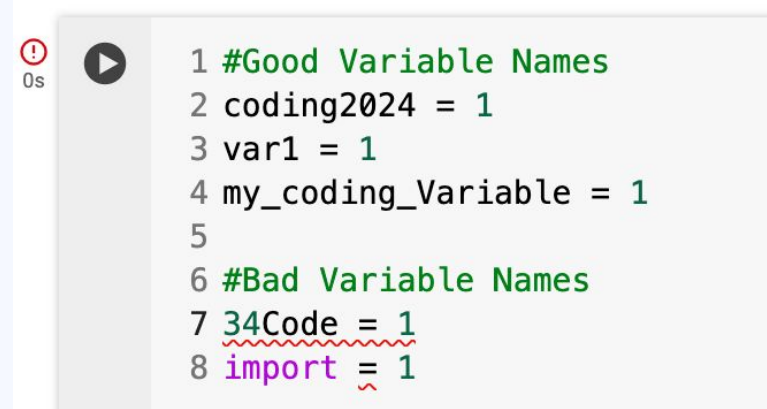
- thisVariable = 4

→ Do's

- ◆ Use letters and numbers
- ◆ Use underscores
- ◆ False or False = False
- ◆ Recognize case sensitivity

→ Don'ts

- ◆ Start with a number
- ◆ Use Python keywords as a variable name



A code editor snippet with a light yellow background. On the left, there is a red circle with an exclamation mark and the text '0s' below it, and a black play button icon. The code is as follows:

```
1 #Good Variable Names
2 coding2024 = 1
3 var1 = 1
4 my_coding_Variable = 1
5
6 #Bad Variable Names
7 34Code = 1
8 import = 1
```

The code is color-coded: comments are green, numbers are black, and variable names are black. In the 'Bad' section, '34Code' has red wavy underlines under '34' and 'Code', and 'import' has red wavy underlines under 'import'.



04

Basic Functions



Basic Functions

print()

Displays result to the screen

```
1 print("This coding camp is so fun")
```

This coding camp is so fun

len()

Returns the length of an object

```
1 len("how many characters are in this sentence?")
```

41

input()

Allows user to enter input

```
1 input("What is your name?")
```

... What is your name?

type()

Returns the data type

```
[13] 1 type(34)
```

int

More Basic Functions

sum()

Returns the sum

```
✓ 0s 1 numbers = [1, 6, 9]
    2 sum(numbers)
    ↗ 16
```

round()

Rounds a float to specified number of decimals

```
✓ 0s 1 round(3.18, 1)
    ↗ 3.2
```

max() and min()

Returns the max and min values

```
✓ 0s 1 max(numbers)
    ↗ 9
```

range()

Returns a sequence of numbers

```
✓ 0s 1 for i in range(5):
    2 | print(i)
    ↗ 0
    1
    2
    3
    4
```



05

Control Flow



Control Flow - If Statements

- **Tells python what to execute and in what order using conditional statements**
 - ◆ If, else and elif
 - ◆ Elif means “else if”
- **Syntax is very important in control flow**
 - ◆ Use of colons
 - ◆ Use of indentation

Control Flow - Conditional Statements

✓
0s



```
1 age = 22
2
3 if age < 21:
4 |   print("You are so young!")
5
6 elif age > 23:
7 | |   print("You are getting older!")
8
9 else:
10 | | |   print("You are 22")
```



You are 22

'If' checks a condition. If it is true, it will execute.

'Elif' checks additional conditions in the case that the if condition is not true

'Else' runs only if all other conditions are false



06

Loops



Control Flow - For Loops

→ **For loops allow you to iterate over a sequence**

◆ Useful if you want to print out something multiple times

✓
0s



```
1 colors = ["Blue", "Purple", "Yellow", "Green"]  
2  
3 for color in colors:  
4 |     print(color)
```



```
Blue  
Purple  
Yellow  
Green
```

Brackets are used to
create a list

Control Flow - While Loops

- **While loops will continue running - so long as a condition is true**
 - ◆ Useful if you are not sure how many times the loop will have to execute until you get the result you want

✓
4s



```
1 while True:
2     command = input("Enter command: ")
3     if command == "exit":
4         break
5     print("You did not enter the proper command, try again", command)
```



```
Enter command: hi
You did not enter the proper command, try again hi
Enter command: exit
```

SO MUCH MORE TO LEARN!

- Object oriented programming
- Lists and Tuples
- Dictionaries
- Pandas
- Importing Files
- Data Visualization
- ...MORE!!!

→ Helpful Resources

- ◆ W3 Schools
- ◆ Kaggle
- ◆ LinkedIn Learning
- ◆ Generate AI on Google Colab

THANKS FOR LISTENING!!!