Python Fundamentals

Data Boot Camp
Lesson 3.1

The Big Picture

Python Unit
Python
Pandas
Matplotlib
APIs
Unit Assessment

Visualization Unit
Web Scraping
Plotly
Leaflet
Tableau
Unit Assessment

Final Project
Project Design
Train Model &
Build Database
*Dashboards
Presentation*

01

02

03

04

05

06

Excel Unit
Excel
VBA
Unit Assessment

Databases Unit
SQL
ETL
SQLite
Unit Assessment

Advanced Topics Unit
R—Stats
Big Data—AWS
Machine Learning
Unit Assessment

Boot Camp Pointers

As you work through this module, remember the following:

01

Don't forget about your support system!

Use office hours, class recordings, Learning Assistants, and more to help boost your learning.

02

Nail down a good study schedule and work it into your daily agenda.

03

Embrace those moments of frustration! It's where you learn the most.

Module 3

This Week: Python

This Week: Python

By the end of this week, you'll know how to:

Read and extract data from CSV files.

Write data to an output file and print the file.

Recognize Python data types, like integers, floating-point decimal numbers, and strings.

Declare variables and perform mathematical operations using data types.

Create and use data structures, like lists, tuples, and dictionaries.

Create and use decision and repetition statements.

Create and use Boolean and logical operators.

This Week's Challenge

Using the skills learned throughout the week, complete an audit of election data and provide a written analysis of your findings for the election commission.

Career Connection

How will you use this module's content in your career?

Module 3

How to Succeed This Week

Quick Tip for Success:

Take full advantage of office hours and your support network. Refactoring this Challenge code might be tricky!

Don't be worried if you also need help with GitHub.

Tips for Success: Python

Keep these tips in mind:

Stay organized!

This will help you keep your focus and easily dive back into your code after a break.

Flowcharts

Consider making flowcharts to help visualize your logic flow while writing Python code.

Module 3

Today's Agenda

Today's Agenda

By completing today's activities, you'll learn the following skills:

01	Python Data Types

02	Printing Formats

Make sure you've downloaded any relevant class files!

Intro to Python and Terminal

The Mighty Python

Few things to note before we move forward:

01

We are diving into a more traditional programming language, Python.
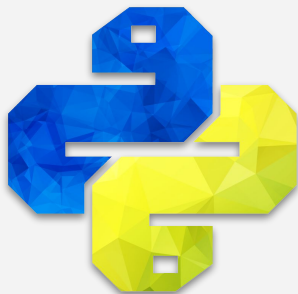
02

The fundamental concepts in Python are the same as VBA, but Python employs different syntax.

03

Check your Slack.
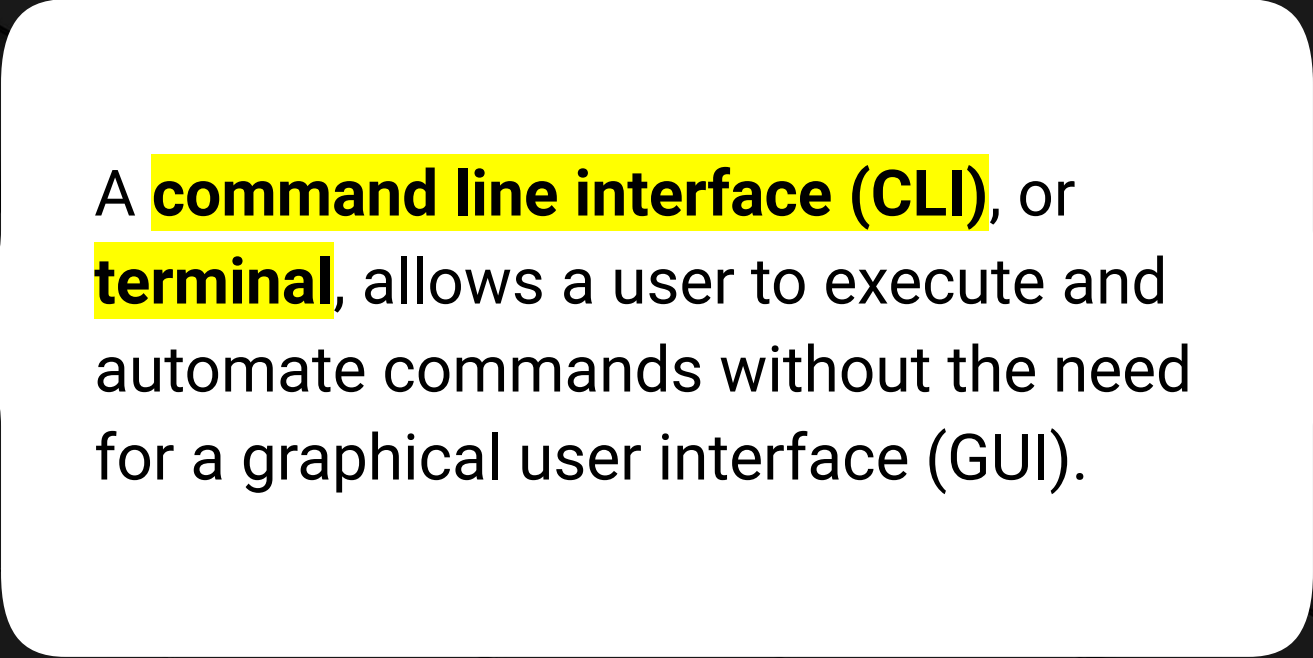
Instructor Demonstration
Installation Check
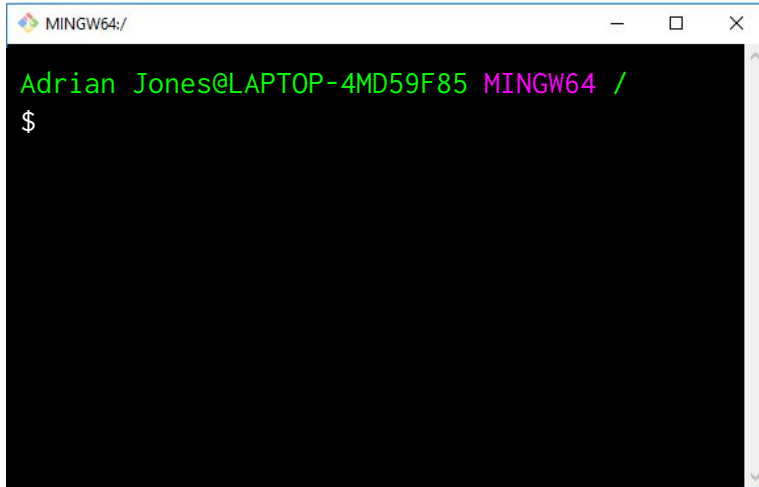
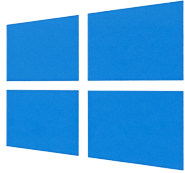Instructor Demonstration
Terminal

A **command line interface (CLI)**, or **terminal**, allows a user to execute and automate commands without the need for a graphical user interface (GUI).
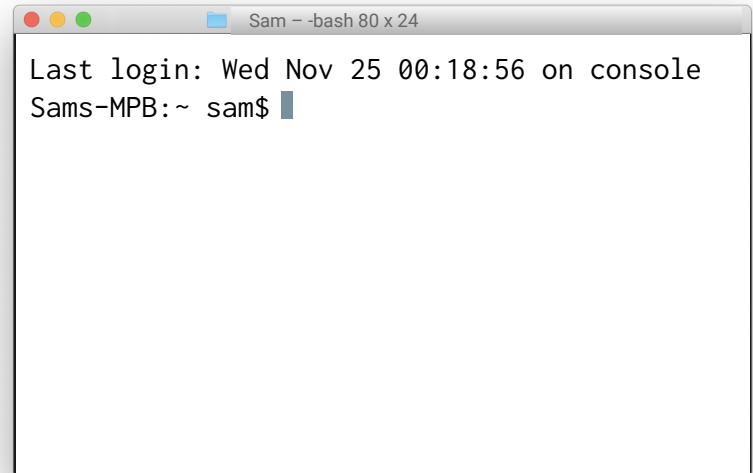
Python Code

Python code will be executed through either `git-bash` (Windows) or the Terminal (Mac)

Windows

```
MINGW64:/                                    —  □  ×

Adrian Jones@LAPTOP-4MD59F85 MINGW64 /
$
```

Mac

```
Sam — -bash 80 x 24

Last login: Wed Nov 25 00:18:56 on console
Sams-MPB:~ sam$
```

Some basic commands

`cd`	Changes the directory
`cd ~`	Changes to the home directory
`cd ..`	Moves up one directory
`ls`	Lists files in the folder
`pwd`	Shows the current directory
`Mkdir <FOLDERNAME>`	Creates a new directory with the FOLDERNAME
`touch <FILENAME>`	Creates a new file with the FILENAME
`rm <FILENAME>`	Deletes a file
`rm -r <FOLDERNAME>`	Deletes a folder, make sure to note the -r
`open .`	Opens the current directory on Macs
`explorer .`	Opens the current directory on GitBash
`open <FILENAME>`	Opens a specific file on Macs
`explorer <FILENAME>`	Opens a specific file on GitBash

Common Commands

```
bash-3.2$ mkdir PythonStuff
bash-3.2$ cd PythonStuff
bash-3.2$ touch first_file.py
bash-3.2$ open first_file.py
```

```
bash-3.2$ python first_file.py
bash-3.2$ This is my first_file.py
```

Activity Workbook: Terminal

As your review the file, think about the following questions:

Where have we used this before?

How does this activity equip us for the Challenge?

What can we do if we don't completely understand this?

Variables

Variables

Similar to values stored in VBA cells

In Python, a value is being stored and given a name

Variables can store different data types like strings, integers, and an entirely new data type called booleans, which hold True or False values.

```python
# Creates a variable with a string "Frankfurter"
Title = "Frankfurter"

# Creates a variable with an integer 80
years = 80

# Creates a variable with the boolean value of True
expert_status = True
```

Instructor Demonstration
Variables

Print Statements

We can print statements that include variables, but traditional Python formatting won't concatenate strings with other data types. This means integers and booleans must be cast as strings using the `str()` function.

```python
# Prints a statement adding the variable
print("Nick is a professional " + title)

# Convert the integer years into a string and prints
print("He has been coding for " + str(years) + " years")

# Converts a boolean into a string and prints
print("Expert status: " + str(expert_status))
```

Alternatively, the `f-string` method of string interpolation allows strings to be formatted with different data types. Demonstrate the differences by refactoring the last print statement as an 'f-string':

```python
# An f-string accepts all data types
without conversion
print(f"Expert status: {expert_status}")
```

Activity: Hello Variable World!

In this activity, you will create a simple Python application that uses variables.

Suggested Time:
15 Minutes

Activity: Hello Variable World!

Instructions:

- Create two variables called `name` and `country` that will hold strings.
- Create two variables called `age` and `hourly_wage` that will hold integers.
- Create a variable called `satisfied` that will hold a boolean.
- Create a variable called `daily_wage` that will hold the value of `hourly_wage` multiplied by 8.
- Print out statements to the console using all of the above variables.

```
HelloVariableWorld.py
You live in United States
You are 25 years old
You make 120 per day
Are you satisfied with your current wage? True
```

What is the difference between an integer and a string?

A string will be wrapped in single
or double quotes, an integer won't.

Lists

Lists

Couple of points to keep in mind before we move forward

01

Lists are the Python equivalent of arrays in VBA, functioning in much the same way by holding multiple pieces of data within one variable.

02

Lists can hold multiple types of data inside of them, as well. This means that strings, integers, and boolean values can be stored within a single list.
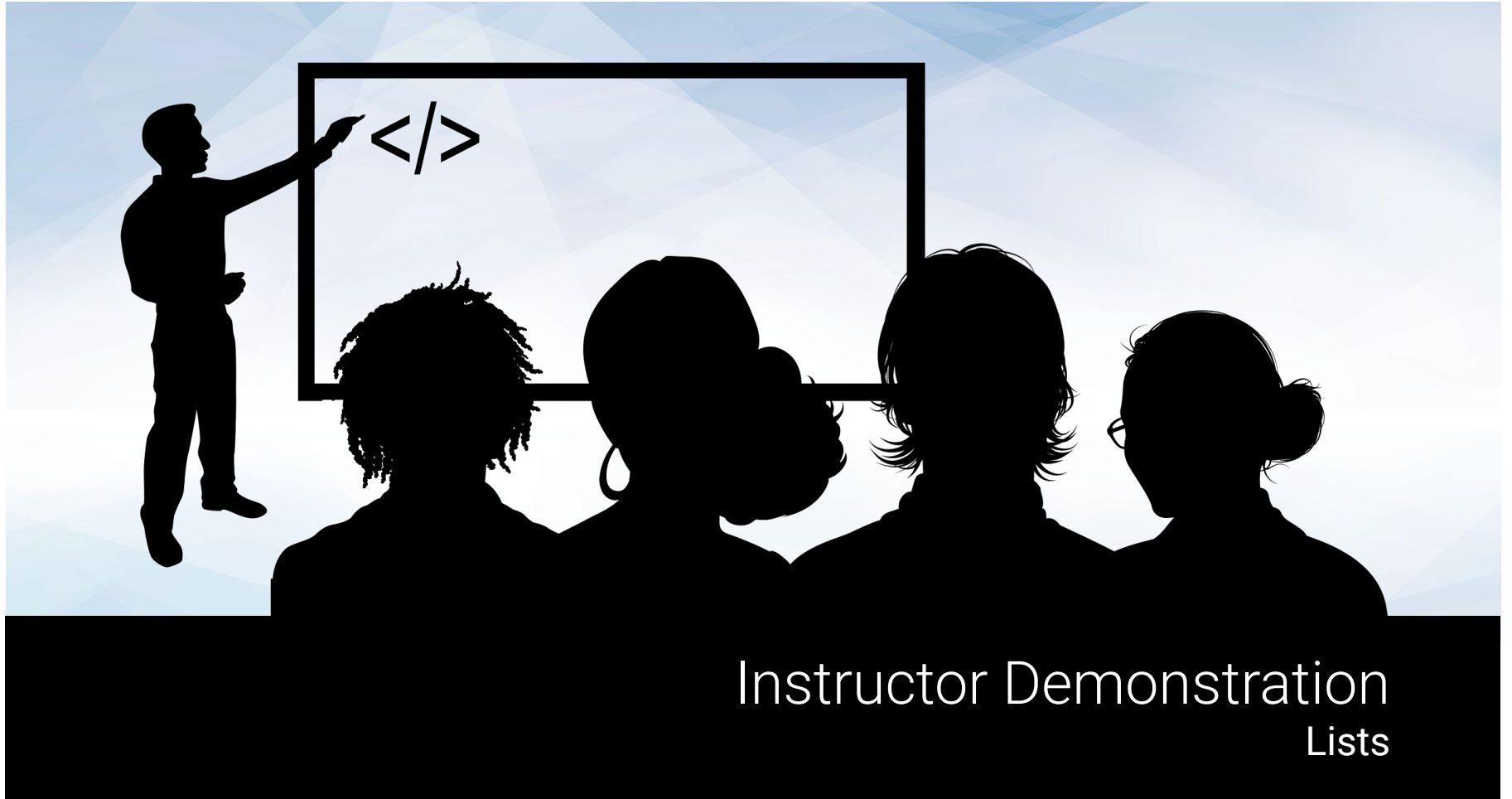
Instructor Demonstration

Lists

Python has a set of built-in methods that you can use on lists

Lists Methods in Python

The append method can add elements to the end of a list.

```python
# Creates a variable and set it as an List
myList = ["Jacob", 25, "Ahmed", 80]
print(myList)

# Adds an element onto the end of the List
myList.append("Matt")
print(myList)

# Changes a specified element within an List at the given index
myList[3] = 85
print(myList)

# Returns the index of first object with a matching value
print(myList.index("Matt"))

# Returns the length of the List
print(len(myList))

# Removes a specified object from an List
myList.remove("Matt")
print(myList)

# Removes the object at the index specified
myList.pop(0)
myList.pop(0)
print(myList)
```

Lists Methods in Python

The `index` method returns the numeric location of a given value within a list.

```python
# Creates a variable and set it as an List
myList = ["Jacob", 25, "Ahmed", 80]
print(myList)

# Adds an element onto the end of the List
myList.append("Matt")
print(myList)

# Changes a specified element within an List at the given index
myList[3] = 85
print(myList)

# Returns the index of first object with a matching value
print(myList.index("Matt"))

# Returns the length of the List
print(len(myList))

# Removes a specified object from an List
myList.remove("Matt")
print(myList)

# Removes the object at the index specified
myList.pop(0)
myList.pop(0)
print(myList)
```

Lists Methods in Python

The `len` function returns the length of a list.

```python
# Creates a variable and set it as an List
myList = ["Jacob", 25, "Ahmed", 80]
print(myList)

# Adds an element onto the end of the List
myList.append("Matt")
print(myList)

# Changes a specified element within an List at the given index
myList[3] = 85
print(myList)

# Returns the index of first object with a matching value
print(myList.index("Matt"))

# Returns the length of the List
print(len(myList))

# Removes a specified object from an List
myList.remove("Matt")
print(myList)

# Removes the object at the index specified
myList.pop(0)
myList.pop(0)
print(myList)
```

Lists Methods in Python

The remove method deletes a given value from a list.

```python
# Creates a variable and set it as an List
myList = ["Jacob", 25, "Ahmed", 80]
print(myList)

# Adds an element onto the end of the List
myList.append("Matt")
print(myList)

# Changes a specified element within an List at the given index
myList[3] = 85
print(myList)

# Returns the index of first object with a matching value
print(myList.index("Matt"))

# Returns the length of the List
print(len(myList))

# Removes a specified object from an List
myList.remove("Matt")
print(myList)

# Removes the object at the index specified
myList.pop(0)
myList.pop(0)
print(myList)
```

Lists Methods in Python

The pop method can be used to remove a value by index.

```python
# Creates a variable and set it as an List
myList = ["Jacob", 25, "Ahmed", 80]
print(myList)

# Adds an element onto the end of the List
myList.append("Matt")
print(myList)

# Changes a specified element within an List at the given index
myList[3] = 85
print(myList)

# Returns the index of first object with a matching value
print(myList.index("Matt"))

# Returns the length of the List
print(len(myList))

# Removes a specified object from an List
myList.remove("Matt")
print(myList)

# Removes the object at the index specified
myList.pop(0)
myList.pop(0)
print(myList)
```

Tuples

Tuples are functionally similar to lists in what they can store but are immutable

While lists in Python can be modified after their creation, tuples can never be modified after their declaration.

Tuples tend to be more efficient to navigate through than lists, and they also protect the data stored within from being changed.

```python
# Creates a tuple, a sequence of immutable Python objects that cannot be changed
myTuple = ('Python', 100, 'VBA', False)
print(myTuple)
```

Activity: Grocery List

In this activity, you will be creating a list of grocery store items as a list of strings that you will print out to the console. Once the list is created, it will need to be updated three times, and the updated list will be printed to the console.

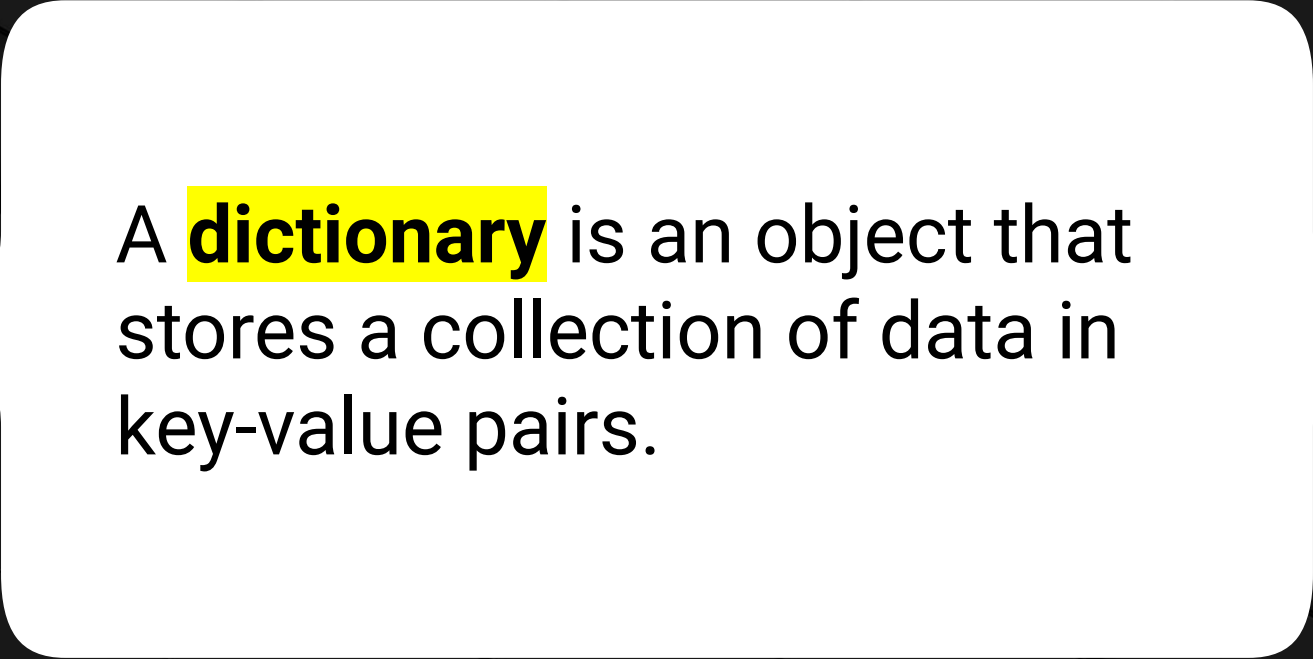# What is the difference between `remove()` and `pop()`?

The `remove()` method removes a specific object by its value from a list, whereas the `pop()` method removes an object by its index.
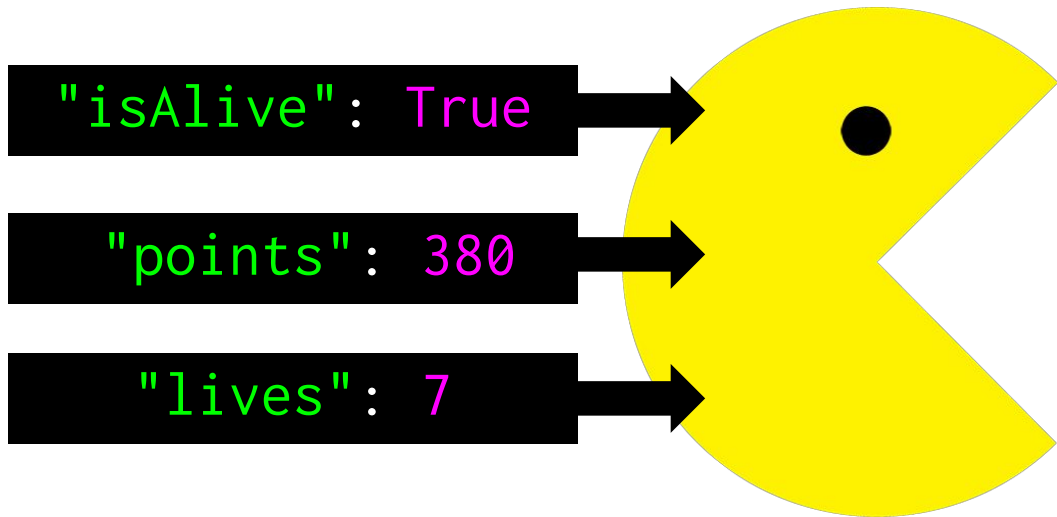
Questions?

Dictionaries

A **dictionary** is an object that stores a collection of data in key-value pairs.

Dictionaries

Dictionaries store data in key-value pairings. The key is a string that can be referenced in order to collect the value that is associated with it.

```
"isAlive": True

"points": 380

"lives": 7
```

```
pacman = {
    "isAlive": True,
    "points": 380,
    "lives": 7
}
```

Instructor Demonstration
Dictionaries

Dictionaries

To initialize or create an empty dictionary, we use the following syntax, `actors = {}`.

```
# Creates a dictionary to
hold the actors names.
actors = {}
```

Or, you can create a dictionary with the built-in Python `dict()` function, `actors = dict()`.

```
# Creates a dictionary
using the built in function
actors = dict()
```

Why do dictionaries have keys?

The key in a dictionary can be referenced in order to collect its associated value.

Questions?

Activity: Hobby-Book–Dictionaries

In this activity, you will create a dictionary to store your pet's name, age, and hobbies as a list, and what time your pet wakes up each day of the week in a dictionary format, where the day of the week is the `key` and the time is the `value`. Then, you'll use `f-strings` to print the results stored in the dictionary.

Suggested Time:
20 Minutes

How would you print two of the pet's hobbies?

We would use the
`{my_info["hobbies"][0]}`
and
`{my_info["hobbies"][1]}`.

Questions?