Introduction to Python

Nyle Siddiqui and Laura Pryor

Welcome!

- Today’s Objectives
  - Introduction to Python
  - Libraries used in machine learning
- Accompanying practice questions and info sheet

Why Python?

- Easy to use
- OOP language but not used heavily for development
- Access to large amount of libraries
  - Specific mathematical libraries
  - Machine learning libraries
  - Data access and manipulation
- Built on C, good for large computations

Example Python Program

```
def helloworld():
    print("Hello World")


if __name__ == '__main__':
    helloworld()
```

Example Python Program

- Functions are created using the keyword **def**
  - Functions are useful for chunks of code that repeat often in the program
  - Call function instead of re-typing code
- “Main” function is not necessary in Python, but creating one can help structure your code better

Introduction to Variables

```
print("There once was a man named George.")
print("He was 70 years old.")
print("He really liked the name George,")
print("But he didn't like being 70.")
```

```
There once was a man named George.
He was 70 years old.
He really liked the name George.
But didn't like being 70.
```

Introduction to Variables

```
name = "George"
age = '70'
print("There once was a man named " + name + ".")
print("He was " + age +  " years old.")
print("He really liked the name " + name + ".")
print("But didn't like being " + age + "."  )
```

```
There once was a man named George.
He was 70 years old.
He really liked the name George.
But didn't like being 70.
```

Introduction to Variables

```
name = "George"
age = '70'
print("There once was a man named " + name + ".")
print("He was " + age +  " years old.")
name = "Tom"
print("He really liked the name " + name + ".")
print("But didn't like being " + age + "."  )
```

```
There once was a man named George.
He was 70 years old.
He really liked the name Tom.
But didn't like being 70.
```

Introduction to Variables

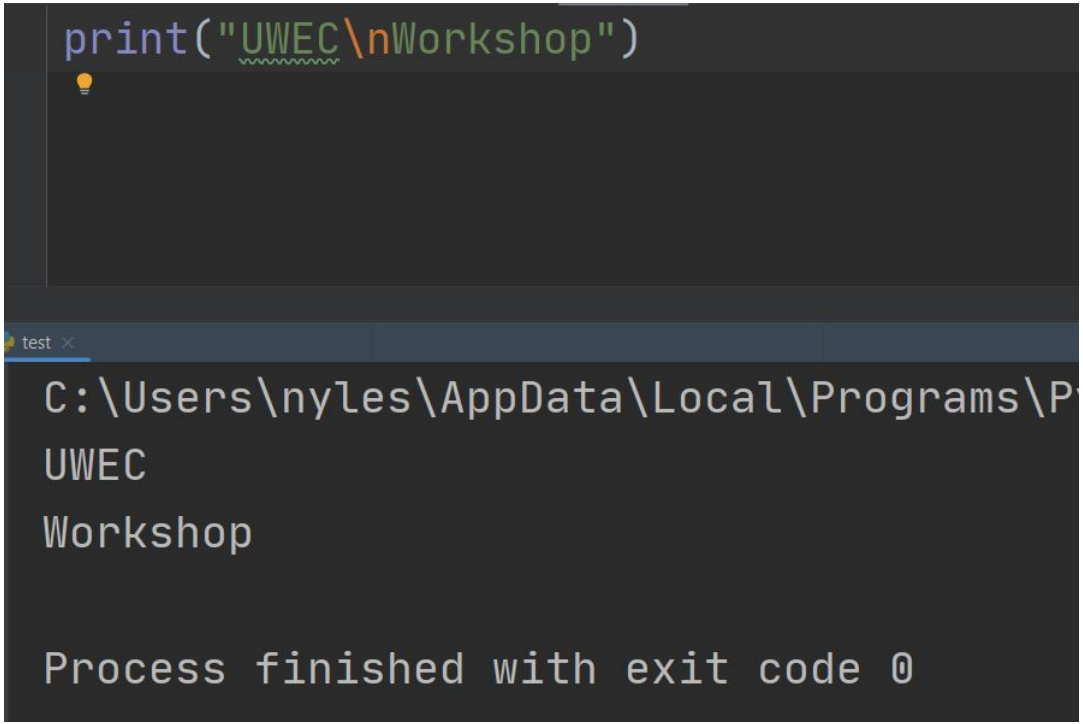
```
name = "George"
age = 70
print("There once was a man named " + name + ".")
print("He was " + age +  " years old.")
name = "Tom"
print("He really liked the name " + name + ".")
print("But didn't like being " + age + "."  )
```

```
Traceback (most recent call last):
  File "C:/Users/nyles/PycharmProjects/Research Code/test.py", line 4, in <module>
    print("He was " + age +  " years old.")
TypeError: can only concatenate str (not "int") to str
There once was a man named George.

Process finished with exit code 1
```

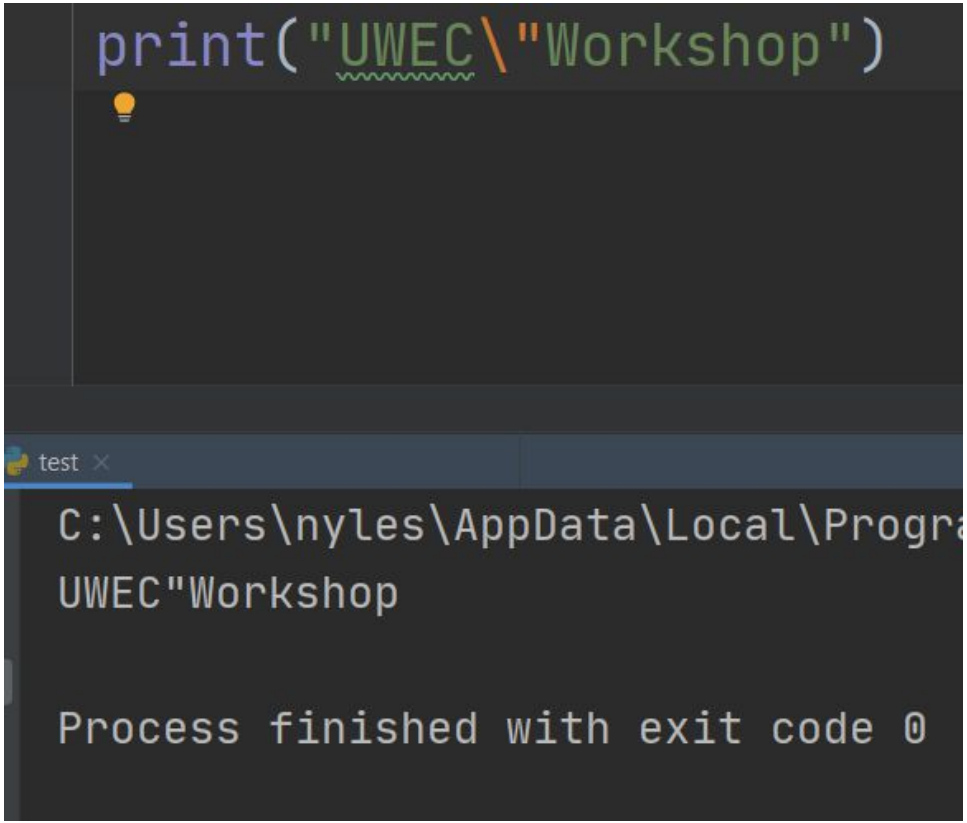
Introduction to Variables - Recap

- Just like functions, variables can be used to store commonly occurring values
- Variables are INTEGRAL to almost any programming language
- Can store integers, floats, strings, etc. (primitives)
    - Primitives are the most basic data structures of a programming language

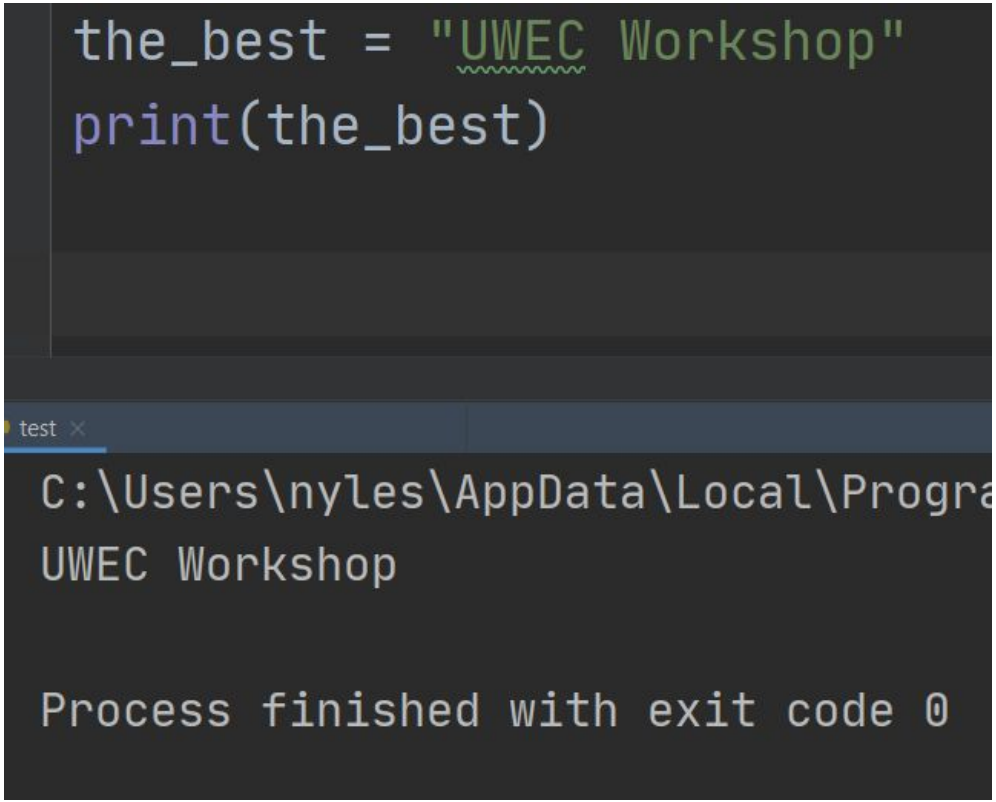
Working with Strings



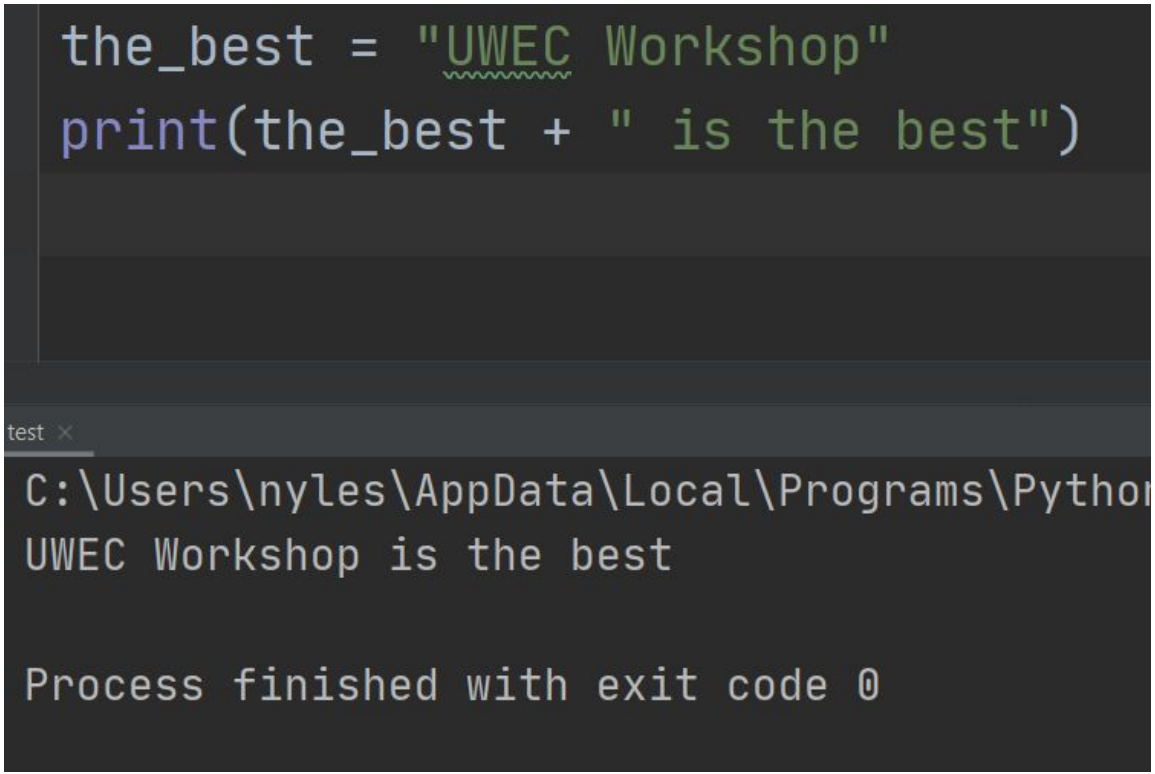
Working with Strings



Working with Strings



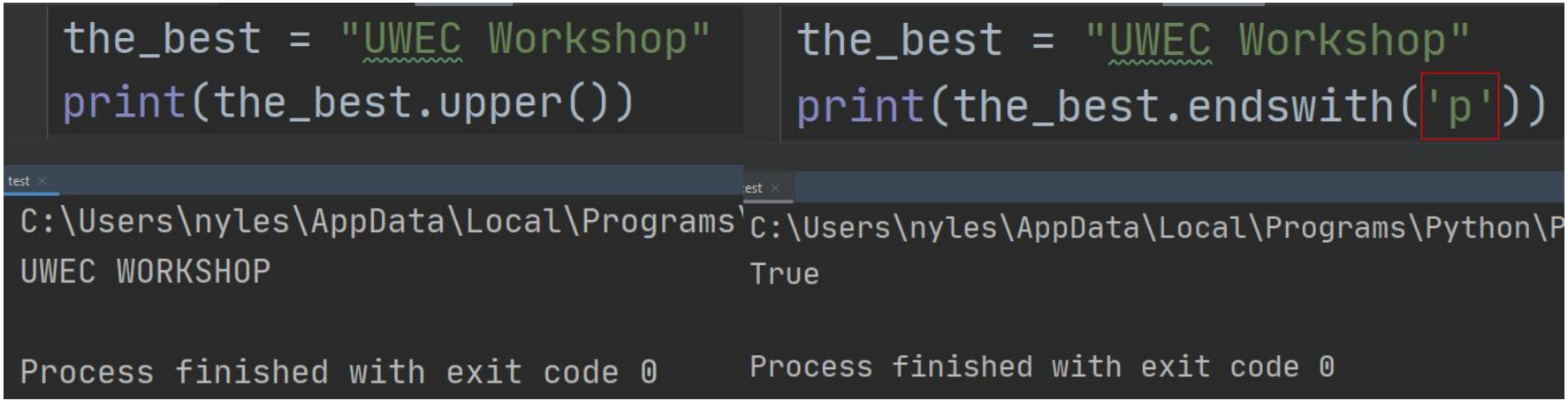
Working with Strings



Working with Strings

Method	Description
capitalize()	Converts the first character to upper case
casefold()	Converts string into lower case
center()	Returns a centered string
count()	Returns the number of times a specified value occurs in a string
encode()	Returns an encoded version of the string
endswith()	Returns true if the string ends with the specified value
expandtabs()	Sets the tab size of the string
find()	Searches the string for a specified value and returns the position of where it was found
format()	Formats specified values in a string
format_map()	Formats specified values in a string
index()	Searches the string for a specified value and returns the position of where it was found
isalnum()	Returns True if all characters in the string are alphanumeric
isalpha()	Returns True if all characters in the string are in the alphabet
isdecimal()	Returns True if all characters in the string are decimals
isdigit()	Returns True if all characters in the string are digits
isidentifier()	Returns True if the string is an identifier
islower()	Returns True if all characters in the string are lower case
isnumeric()	Returns True if all characters in the string are numeric
isprintable()	Returns True if all characters in the string are printable
isspace()	Returns True if all characters in the string are whitespaces
istitle()	Returns True if the string follows the rules of a title
isupper()	Returns True if all characters in the string are upper case
join()	Joins the elements of an iterable to the end of the string

Working with Strings

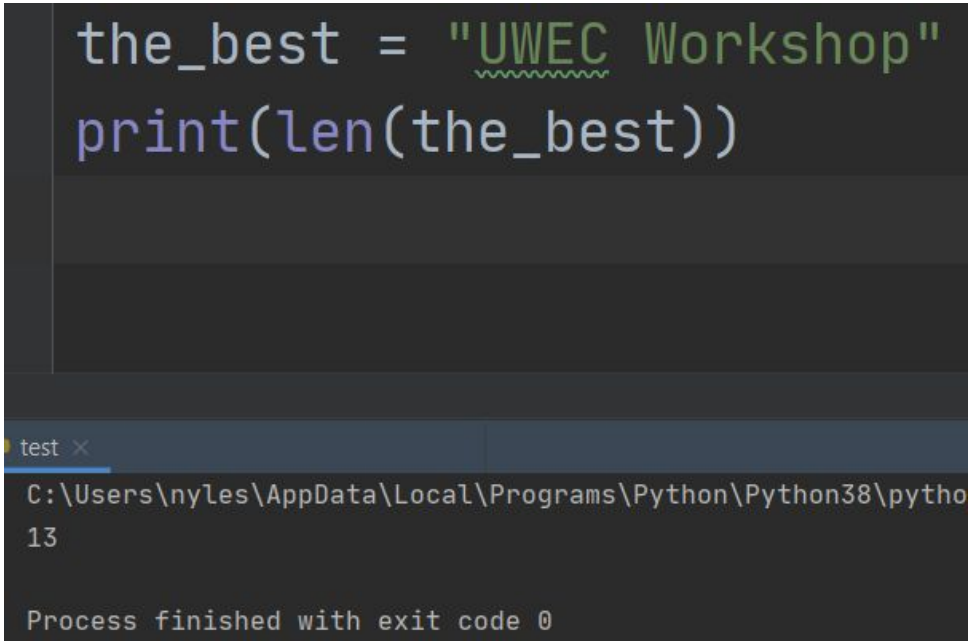


Working with Strings

```
the_best = "UWEC Workshop"
print(the_best.upper().endswith('P'))
```

```
test
C:\Users\nyles\AppData\Local\Programs\Python\Python38\python.ex
True

Process finished with exit code 0
```

Working with Strings



Working with Strings

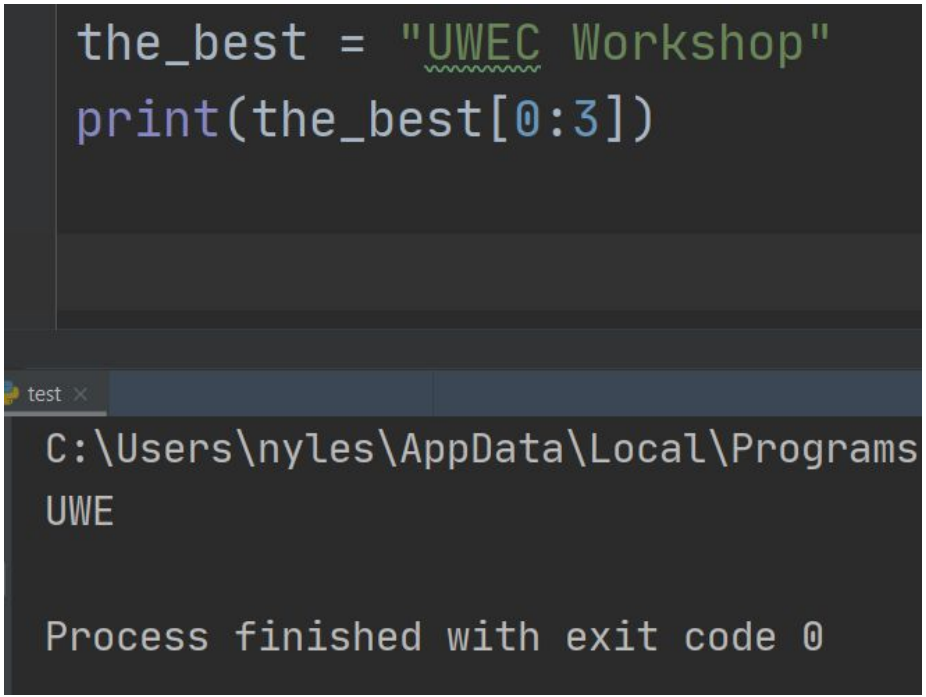
```
the_best = "UWEC Workshop"
print(the_best[0])
```

```
test
C:\Users\nyles\AppData\Local\Prograr
U

Process finished with exit code 0
```

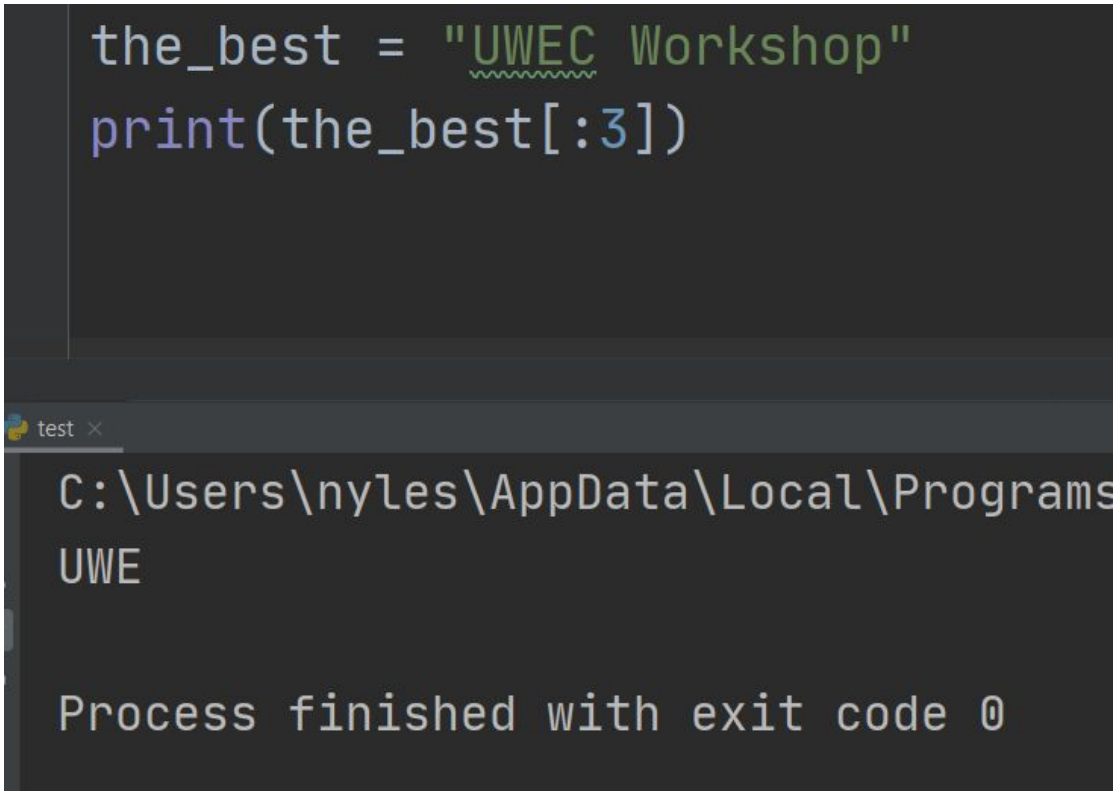
```
the_best = "UWEC Workshop"
print(the_best[13])
```

```
test
C:\Users\nyles\AppData\Local\Programs\Python\Pytho
Traceback (most recent call last):
  File "C:/Users/nyles/PycharmProjects/Research Co
    print(the_best[13])
IndexError: string index out of range

Process finished with exit code 1
```

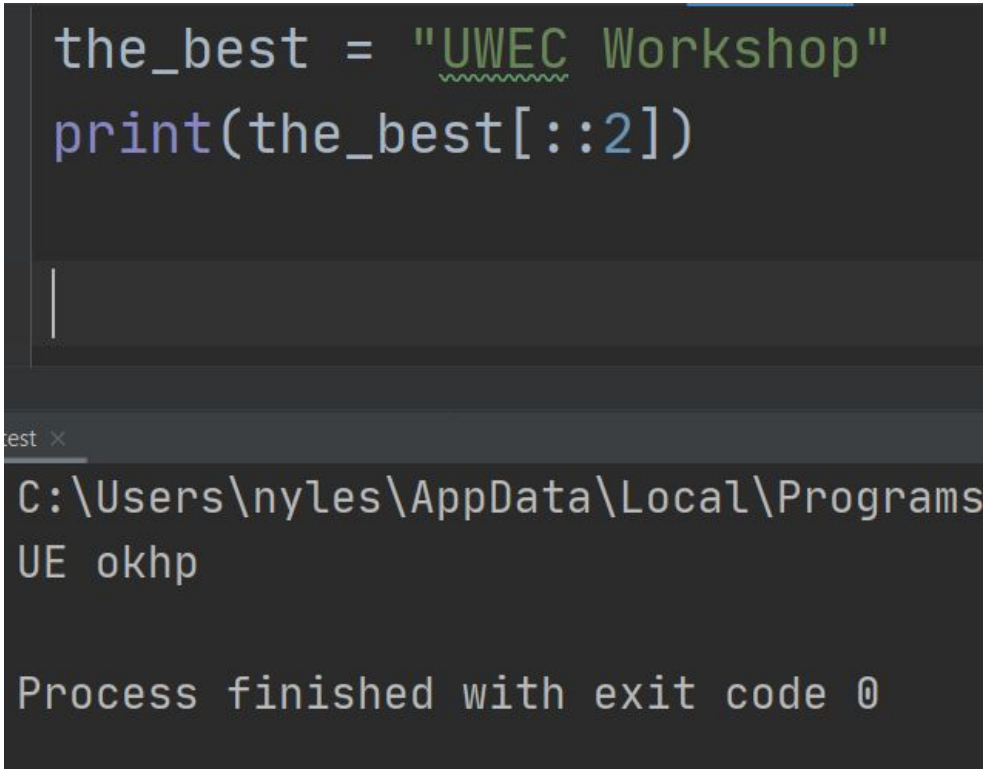
Working with Strings



Working with Strings



Working with Strings



Working with Strings

```
the_best = "UWEC Workshop"
print(the_best[::3])
```

```
C:\Users\nyles\AppData\Local\Programs
UCosp

Process finished with exit code 0
```

Working with Strings - Recap

- ‘\’, also known as the escape character, can be used in strings to print characters that may otherwise cause an error
  - For example, **“”** denotes the start and end of a string, so to print quotations you must use **\”**
- Strings have various methods built-in that can be called
  - One example was the **.upper()** method - it capitalizes all letters in the string
  - List of methods were given, more can be found online
- One of the most important facts about Python strings: string splicing
  - A string can be accessed using [*start_index:end_index:step_size*]
  - Default values are [0:len(string):1] - this essentially prints the entire string
- REMEMBER THAT STRING INDICES START AT 0

Working with Strings - Recap

- Python has built-in functions
  - **len()** - returns the length of a list or string
  - **int() -** changes an object into an int (if possible)
  - **input()** - used for user input
  - **abs()** - absolute value
  - And more!
- String concatenation
  - “Adding” strings together to create one unified string

Operators

```
print(3 * 4 + 5)
```

```
test
C:\Users\nyles\AppData\Local\Prograr
17

Process finished with exit code 0
```

```
print(3 * (4 + 5))
```

```
test
C:\Users\nyles\AppData\Local\Progra
27

Process finished with exit code 0
```

Operators

```
print(10 % 3)
```

```
test
C:\Users\nyles\AppData\Local\Prograr
1

Process finished with exit code 0
```

```
print(10 // 3)
```

```
test
C:\Users\nyles\AppData\Local\Progra
3

Process finished with exit code 0
```

```
print(True or False and False)
```

Operators - Recap

- Be cognizant of order of operations
  - Boolean order of operations are often forgotten: evaluate **and** statements before **or** statements
- Modulo and floor divide are operations not often talked about
  - Used quite often in computer science and programming in general

User Input

```
name = input("Enter a name: ")
age = input("Enter an age: ")
print("Name: " + name + "\nAge: " + age)
```

```
test
C:\Users\nyles\AppData\Local\Programs\Python\Python38\python
Enter a name: john
Enter an age: 25
Name: john
Age: 25

Process finished with exit code 0
```

Practice Questions Section 1

Booleans and Conditionals

```
is_male = True
if is_male:
    print("This person is a male")
else:
    print("This person is not a male")
```

```
C:\Users\nyles\AppData\Local\Programs\Python\Python38\pytho
This person is a male

Process finished with exit code 0
```

Booleans and Conditionals

```
is_male = True
is_tall = True
if is_male and is_tall:
    print("This person is a male and is tall")
else:
    print("This person is either not a male or not tall")
```

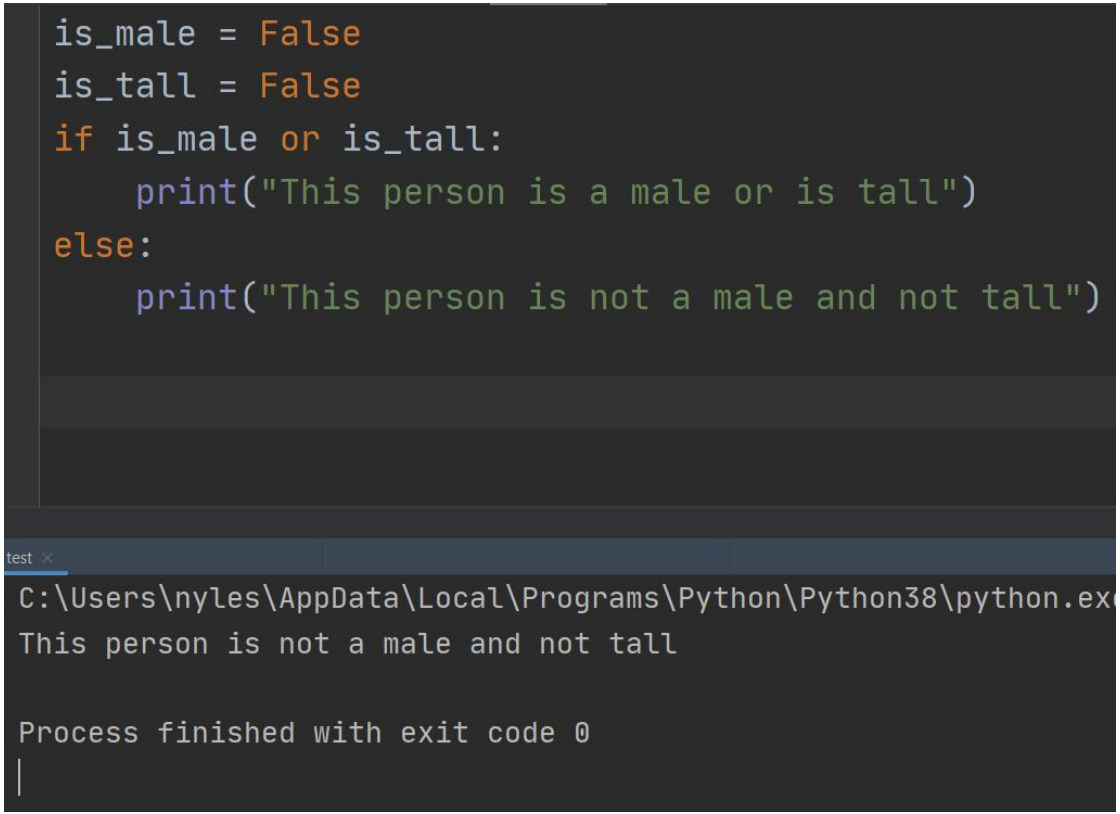
```
test
C:\Users\nyles\AppData\Local\Programs\Python\Python38\python.exe "C:/Users/nyles/
This person is a male and is tall

Process finished with exit code 0
```

Booleans and Conditionals

```
is_male = True
is_tall = False
if is_male and is_tall:
    print("This person is a male and is tall")
else:
    print("This person is either not a male or not tall")
```

```
test
C:\Users\nyles\AppData\Local\Programs\Python\Python38\python.exe "C:/Users/
This person is either not a male or not tall

Process finished with exit code 0
```

Booleans and Conditionals



Booleans and Conditionals

```
is_male = False
is_tall = True
if is_male and is_tall:
    print("This person is a male or is tall")
elif is_male and not is_tall:
    print("This person is a short male")
elif not is_male and is_tall:
    print("This person is not a male but is tall")
else:
    print("This person is not a male and not tall")
```

```
test ×
C:\Users\nyles\AppData\Local\Programs\Python\Python38\python.e
This person is not a male but is tall

Process finished with exit code 0
```

Booleans and Conditionals

```python
def max_num(num1, num2, num3):
    if num1 >= num2 and num1 >= num3:
        return num1
    elif num2 >= num1 and num2 >= num3:
        return num2
    else:
        return num3

print(max_num(3, 4, 5))
```

```
test
C:\Users\nyles\AppData\Local\Programs\Python\Python38\python.exe "C:/Users/n
5

Process finished with exit code 0
```

```python
def max_num(num1, num2, num3):
    if num1 >= num2 and num1 >= num3:
        return num1
    elif num2 >= num1 and num2 >= num3:
        return num2
    else:
        return num3

maxx = max_num(300, 400 ,5)
print(maxx)
```

```
test
C:\Users\nyles\AppData\Local\Programs\Python\Python38\python.exe "C:/User
400

Process finished with exit code 0
```

Booleans and Conditionals - Recap

- If - else statements
  - Used to separate code based on certain cases
- Return statements can be used in functions to literally “return” a value when the function has fully executed
  - Value can then be stored in a variable for later use

Checkpoint: Building a Calculator

- Ask the user for 2 numbers
- Ask the user for the operator
- Compute the operation
- Bonus: Keep asking for user input after each computation until the user quits

Checkpoint: Building a Calculator - Solution

```python
flag = True
while flag:
    x = int(input("Enter a number: "))
    y = int(input("Enter another number: "))
    operation = input("Enter your desired operation: ")
    if operation == '+':
        print(x + y)
    elif operation == '-':
        print(x - y)
    elif operation == '*' or operation == 'x':
        print(x * y)
    elif operation == '/':
        print(x / y)
    answer = input("quit? y/n: ")
    if answer == 'y':
        flag = False
```

```
test
C:\Users\nyles\AppData\Local\Programs\Python\Python3
Enter a number: 9
Enter another number: 6
Enter your desired operation: +
15
quit? y/nn
Enter a number: 9
Enter another number: 6
Enter your desired operation: -
3
quit? y/ny

Process finished with exit code 0
```

Practice Questions Section 2

Dictionaries, Lists, Tuples, and Sets

- Dictionaries
  - Unordered (3.6 and earlier), changeable, no duplicates
- Lists
  - Ordered, changeable, allows duplicates
- Tuple
  - Ordered, unchangeable, allows duplicates
- Sets
  - Unordered, unchangeable, no duplicates

Dictionaries

```
month_conversions = {
    'Jan' : 'January',
    'Feb' : 'February',
    'Mar' : 'March',
    'Apr' : 'April',
    'May' : 'March',
    'Jun' : 'June',
    'Jul' : 'July',
    'Aug' : 'August',
    'Sep' : 'September',
    'Oct' : 'October',
    'Nov' : 'November',
    'Dec' : 'December',
}
print(month_conversions.get("Jan"))
```

```
test
C:\Users\nyles\AppData\Local\Programs\Py
January

Process finished with exit code 0
```

```
month_conversions = {
    0 : 'January',
    'Feb' : 'February',
    'Mar' : 'March',
    'Apr' : 'April',
    'May' : 'March',
    'Jun' : 'June',
    'Jul' : 'July',
    'Aug' : 'August',
    'Sep' : 'September',
    'Oct' : 'October',
    'Nov' : 'November',
    'Dec' : 'December',
}
print(month_conversions.get(0))
```

```
test
C:\Users\nyles\AppData\Local\Programs\
January

Process finished with exit code 0
```

```
month_conversions = {
    0 : 'January',
    'Feb' : 'February',
    'Mar' : 'March',
    'Apr' : 'April',
    'May' : 'March',
    'Jun' : 'June',
    'Jul' : 'July',
    'Aug' : 'August',
    'Sep' : 'September',
    'Oct' : 'October',
    'Nov' : 'November',
    'Dec' : 'December',
}
print(month_conversions.get(1))
```

```
test
C:\Users\nyles\AppData\Local\Programs\
None

Process finished with exit code 0
```

Dictionaries

```
month_conversions = {
    0 : 'January',
    'Feb' : 'February',
    'Mar' : 'March',
    'Apr' : 'April',
    'May' : 'March',
    'Jun' : 'June',
    'Jul' : 'July',
    'Aug' : 'August',
    'Sep' : 'September',
    'Oct' : 'October',
    'Nov' : 'November',
    'Dec' : 'December',
}
print(month_conversions.get(1, "Not a valid key"))
```

```
test
C:\Users\nyles\AppData\Local\Programs\Python\Pytho
Not a valid key

Process finished with exit code 0
```

```
month_conversions = {
    0 : 'January',
    0 : 'February',
    'Feb' : 'February',
    'Mar' : 'March',
    'Apr' : 'April',
    'May' : 'March',
    'Jun' : 'June',
    'Jul' : 'July',
    'Aug' : 'August',
    'Sep' : 'September',
    'Oct' : 'October',
    'Nov' : 'November',
    'Dec' : 'December',
}
print(month_conversions.get(0, "Not a valid key"))
```

```
test
C:\Users\nyles\AppData\Local\Programs\Python\Python3
February

Process finished with exit code 0
```

Dictionaries

```
month_conversions = {
    "Jan" : 'January',
    'Feb' : 'February',
    'Mar' : 'March',
    'Apr' : 'April',
    'May' : 'March',
    'Jun' : 'June',
    'Jul' : 'July',
    'Aug' : 'August',
    'Sep' : 'September',
    'Oct' : 'October',
    'Nov' : 'November',
    'Dec' : 'December',
}
print(month_conversions[0, "not a valid key"])
```

```
test
C:\Users\nyles\AppData\Local\Programs\Python\Python38\python.exe "C:/Users/nyles/Pycha
Traceback (most recent call last):
  File "C:/Users/nyles/PycharmProjects/Research Code/test.py", line 15, in <module>
    print(month_conversions[0, "not a valid key"])
KeyError: (0, 'not a valid key')

Process finished with exit code 1
```

Method	Description
clear()	Removes all the elements from the dictionary
copy()	Returns a copy of the dictionary
fromkeys()	Returns a dictionary with the specified keys and value
get()	Returns the value of the specified key
items()	Returns a list containing a tuple for each key value pair
keys()	Returns a list containing the dictionary's keys
pop()	Removes the element with the specified key
popitem()	Removes the last inserted key-value pair
setdefault()	Returns the value of the specified key. If the key does not exist: insert the key, with the specified value
update()	Updates the dictionary with the specified key-value pairs
values()	Returns a list of all the values in the dictionary

Dictionaries - Recap

- Generally unordered, changeable, does not allow duplicates
- “Word/Definition” pair
- Not used often in ML

Lists

```
friends = ["Nyle", "Laura", "Zach", "Heather", "Alexis"]

print(friends)
```

```
Test
C:\Users\nyles\Anaconda3\python.exe C:/Users/nyles/Py
['Nyle', 'Laura', 'Zach', 'Heather', 'Alexis']

Process finished with exit code 0
```

```
friends = ["Nyle", "Laura", "Zach", "Heather", "Alexis"]

print(friends[1])
```

```
Test
C:\Users\nyles\Anaconda3\python.exe C:/Users/nyles/Pychar
Laura

Process finished with exit code 0
```

Lists

```
friends = ["Nyle", "Laura", "Zach", "Heather", "Alexis"]

print(friends[-1])
print(friends[-2])
```

```
Test
C:\Users\nyles\Anaconda3\python.exe C:/Users/nyles/PycharmPro
Alexis
Heather

Process finished with exit code 0
```

```
friends = ["Nyle", "Laura", "Zach", "Heather", "Alexis"]

print(friends[:-1])
```

```
est
C:\Users\nyles\Anaconda3\python.exe C:/Users/nyles/PycharmProjects/Sorting/Test.py
['Nyle', 'Laura', 'Zach', 'Heather']

Process finished with exit code 0
```

Lists

```
friends = ["Nyle", "Laura", "Zach", "Heather", "Alexis"]

print(friends[:-1])
```

```
Test
C:\Users\nyles\Anaconda3\python.exe C:/Users/nyles/PycharmProjects/Sorting/Test.p
['Nyle', 'Laura', 'Zach', 'Heather']

Process finished with exit code 0
```

Lists


friends = ["Nyle", "Laura", "Zach", 0 , True, 3.6]
print(friends[1:4])
test
C:\Users\nyles\AppData\Local\Programs\Python\Python38\python.exe "C:/Users/nyles/PycharmProjects
['Laura', 'Zach', 0]
Process finished with exit code 0

Lists

```
friends = ["Nyle", "Laura", "Zach", 0 , True, 3.6]
friends[1] = "Pryor"
print(friends[1:4])
```

```
test
C:\Users\nyles\AppData\Local\Programs\Python\Python38\python.exe "C:/Users/nyles/PycharmProje
['Pryor', 'Zach', 0]

Process finished with exit code 0
```

Lists

Method	Description
append()	Adds an element at the end of the list
clear()	Removes all the elements from the list
copy()	Returns a copy of the list
count()	Returns the number of elements with the specified value
extend()	Add the elements of a list (or any iterable), to the end of the current list
index()	Returns the index of the first element with the specified value
insert()	Adds an element at the specified position
pop()	Removes the element at the specified position
remove()	Removes the first item with the specified value
reverse()	Reverses the order of the list
sort()	Sorts the list

Lists - Recap

- Ordered, changeable, allows duplicates
- Python’s versions of arrays
- Very commonly used in ML
- Also can be accessed in splices like strings
    - Easily accessible in general
- Can contain any combination of primitive types

Tuples

```
coordinates = (4, 5)
print(coordinates)
```

```
test
C:\Users\nyles\AppData\Local\Program
(4, 5)

Process finished with exit code 0
```

```
coordinates = (4, 5)
print(coordinates[1])
```

```
test
C:\Users\nyles\AppData\Local\Programs
5

Process finished with exit code 0
```

Tuples

```
coordinates = (4, 5)
coordinates[1] = 10
print(coordinates[1])
```

```
test
C:\Users\nyles\AppData\Local\Programs\Python\Python38\python.exe "C:/Users/nyles/Pyc
Traceback (most recent call last):
  File "C:/Users/nyles/PycharmProjects/Research Code/test.py", line 2, in <module>
    coordinates[1] = 10
TypeError: 'tuple' object does not support item assignment

Process finished with exit code 1
```

```
coordinates = [4, 5]
coordinates[1] = 10
print(coordinates[1])
```

```
test
C:\Users\nyles\AppData\Local\Programs
10

Process finished with exit code 0
```

Tuples

Method	Description
count()	Returns the number of times a specified value occurs in a tuple
index()	Searches the tuple for a specified value and returns the position of where it was found

Tuples - Recap

- Ordered, **unchangeable**, allows duplicates
- Can be useful in ML to prevent data loss
- Very few methods available since tuples cannot change after initialization

For Loops

```
for i in range(10):
    print(i)
```

```
C:\Users\nyles\AppData\Local\Progra
0
1
2
3
4
5
6
7
8
9

Process finished with exit code 0
```

```
for i in range(3, 10):
    print(i)
```

```
C:\Users\nyles\AppData\Local\Progr
3
4
5
6
7
8
9

Process finished with exit code 0
```

```
for letter in "Wisconsin":
    print(letter)
```

```
C:\Users\nyles\AppData\Local\Programs
W
i
s
c
o
n
s
i
n

Process finished with exit code 0
```

For Loops

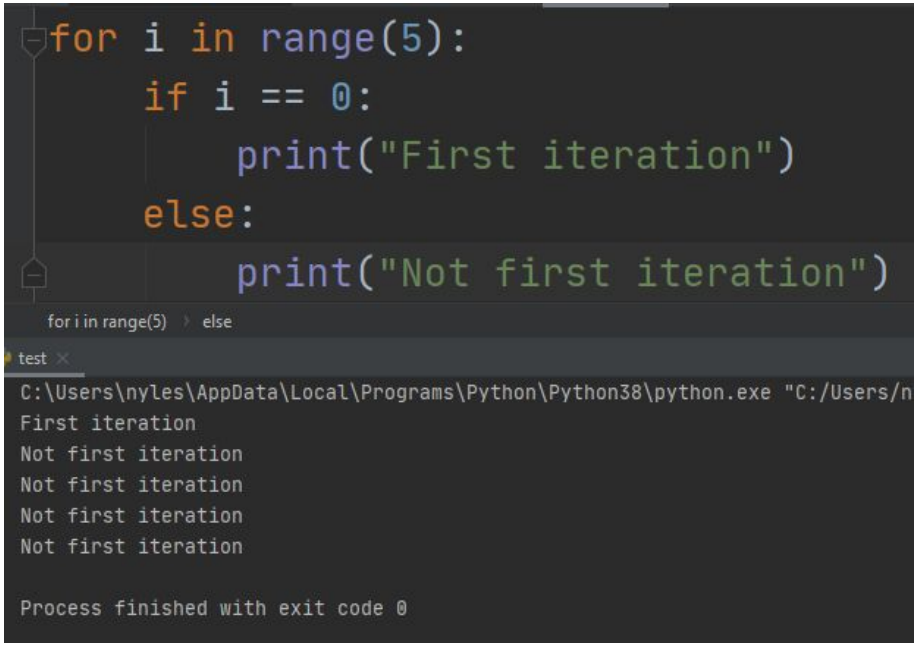
```
friends = ["Nyle", 'Laura', 'Zach']
for friend in friends:
    print(friend)
```

```
test
C:\Users\nyles\AppData\Local\Programs\Python\Pyth
Nyle
Laura
Zach

Process finished with exit code 0
```

```
friends = ["Nyle", 'Laura', 'Zach']
for index in range(len(friends)):
    print(friends[index])
```

```
test
C:\Users\nyles\AppData\Local\Programs\P
Nyle
Laura
Zach

Process finished with exit code 0
```

For Loops


for i in range(5):
if i == 0:
print("First iteration")
else:
print("Not first iteration")
for i in range(5) › else
test
C:\Users\nyles\AppData\Local\Programs\Python\Python38\python.exe "C:/Users/n
First iteration
Not first iteration
Not first iteration
Not first iteration
Not first iteration
Process finished with exit code 0

For Loops - Recap

- Syntactically different than in Java or C++ but functionally the same
  - An example to help you remember
    - Java: for(int i = 0; i < 5; i++)
    - Python: for i in range(5) (i++ is implicit)
- Useful for finitely looping through anything sequential

N-Dimensional Lists and Nested Loops

```
number_grid = [
    [1, 2, 3],
    [4, 5, 6],
    [7, 8, 9]
]
print(number_grid)
```

```
test
C:\Users\nyles\AppData\Local\Progra
[[1, 2, 3], [4, 5, 6], [7, 8, 9]]

Process finished with exit code 0
```

```
number_grid = [
    [1, 2, 3],
    [4, 5, 6],
    [7, 8, 9]
]
for row in number_grid:
    print(row)
```

```
test
C:\Users\nyles\AppData\Local\Prograr
[1, 2, 3]
[4, 5, 6]
[7, 8, 9]

Process finished with exit code 0
```

N-Dimensional Lists and Nested Loops

```
number_grid = [
    [1, 2, 3],
    [4, 5, 6],
    [7, 8, 9]
]
for row in number_grid:
    for number in row:
        print(number)
```

```
test
C:\Users\nyles\AppData\Local\Progra
1
2
3
4
5
6
7
8
9

Process finished with exit code 0
```

N-Dimensional Lists and Nested Loops

- Multi-dimensional lists need nested loops in order to access specific elements
- Think of each dimension being unravelled by each for loop
  - In our example, a 2-dimensional list needed 2 for loops to access a specific element
- Can use less for loops to get more general data - like rows and columns
- Nested loops are important and used for much more than just multidimensional list access
  - Don’t be scared when you see nested for loops: if you know how one loop works, you know how any nested loop works

Try and Except

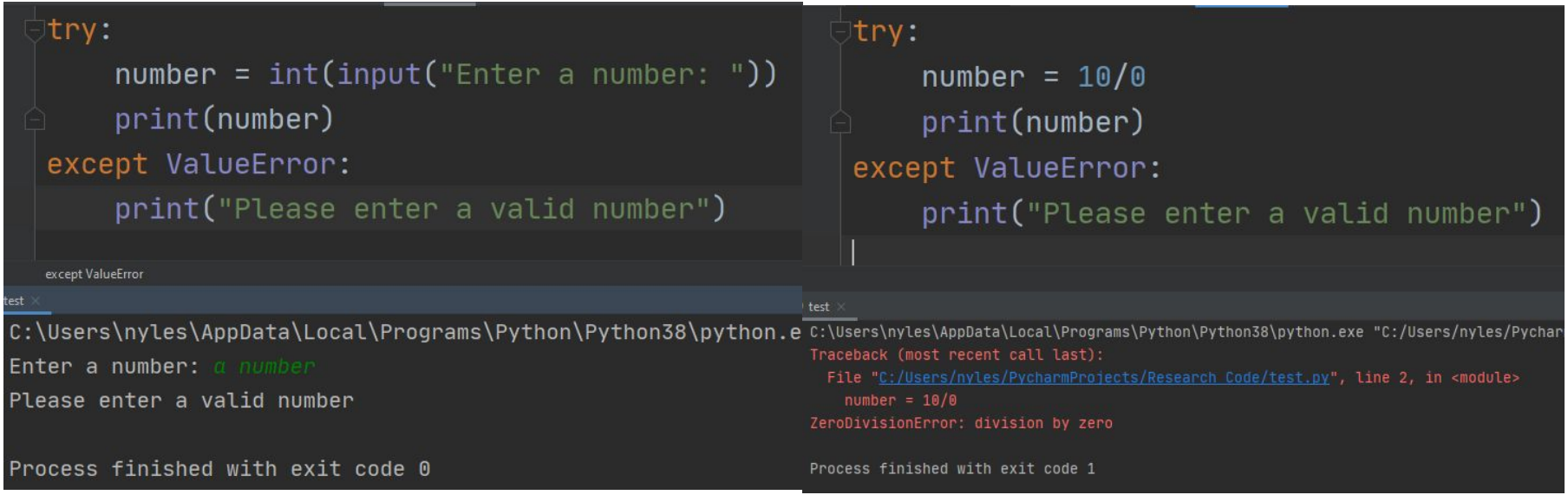
```
number = int(input("Enter a number: "))
print(number)
```

```
test
C:\Users\nyles\AppData\Local\Programs\Python\Pytho
Enter a number: 4
4

Process finished with exit code 0
```

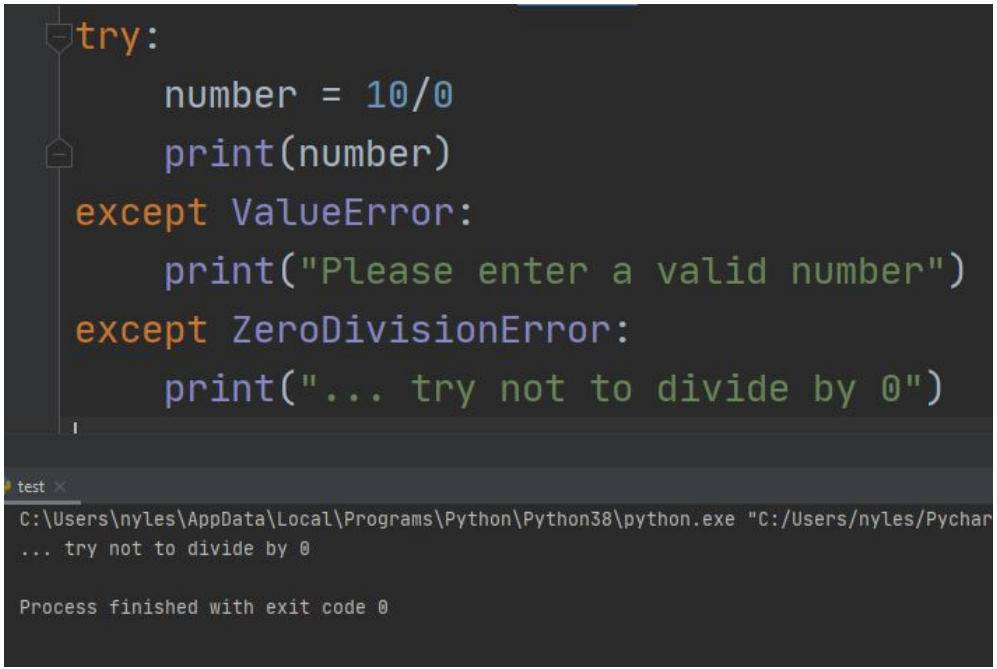
```
number = int(input("Enter a number: "))
print(number)
```

```
test
C:\Users\nyles\AppData\Local\Programs\Python\Python38\python.exe "C:/Users/nyles/Pycha
Enter a number: a number
Traceback (most recent call last):
  File "C:/Users/nyles/PycharmProjects/Research Code/test.py", line 1, in <module>
    number = int(input("Enter a number: "))
ValueError: invalid literal for int() with base 10: 'a number'

Process finished with exit code 1
```

Try and Except



Try and Except

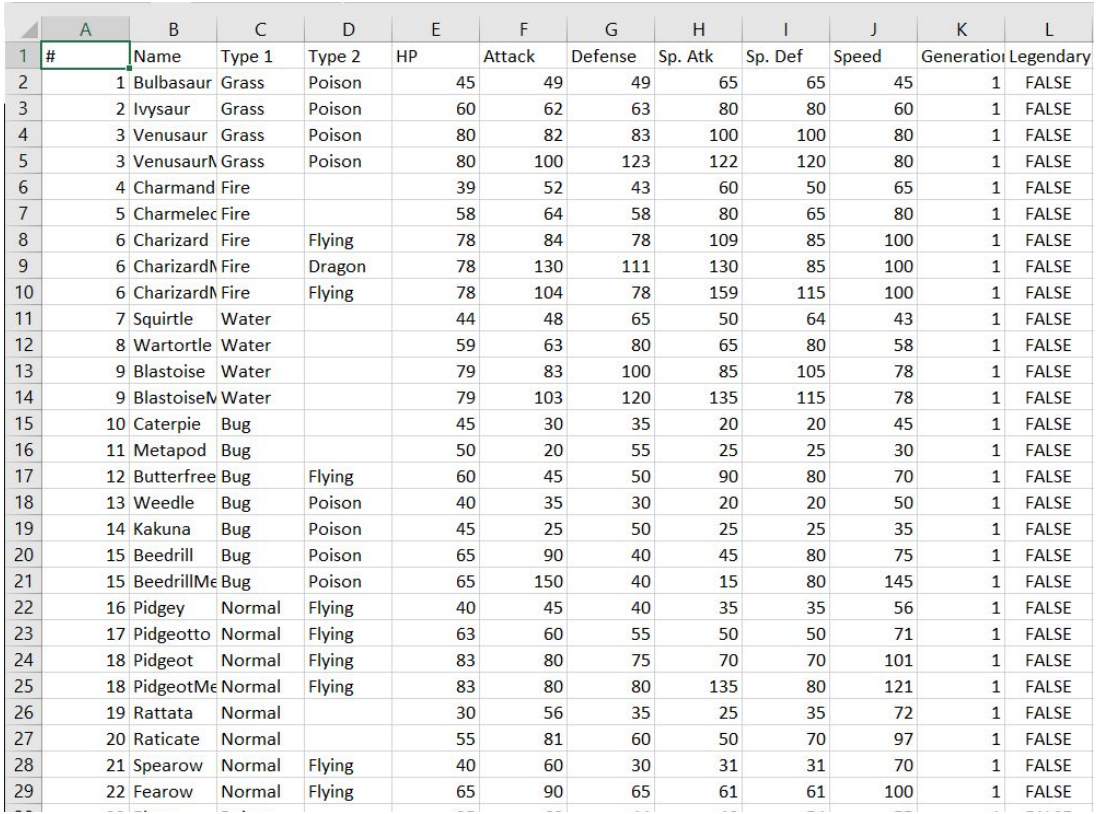


Try and Except

- Use **try** and **except** blocks to catch certain errors
  - Rather than the program stopping itself, you can make any necessary changes or print more helpful troubleshooting messages
  - Really useful when looking over data
    - If one row is bad, you can specify to skip over it and keep combing through data
  - Be sure to specify the error you want to catch
    - A regular **except** keyword will catch ALL errors

Practice Questions Section 3

Pandas



	A	B	C	D	E	F	G	H	I	J	K	L
1	#	Name	Type 1	Type 2	HP	Attack	Defense	Sp. Atk	Sp. Def	Speed	Generatio	Legendary
2	1	Bulbasaur	Grass	Poison	45	49	49	65	65	45	1	FALSE
3	2	Ivysaur	Grass	Poison	60	62	63	80	80	60	1	FALSE
4	3	Venusaur	Grass	Poison	80	82	83	100	100	80	1	FALSE
5	3	VenusaurM	Grass	Poison	80	100	123	122	120	80	1	FALSE
6	4	Charmand	Fire		39	52	43	60	50	65	1	FALSE
7	5	Charmelec	Fire		58	64	58	80	65	80	1	FALSE
8	6	Charizard	Fire	Flying	78	84	78	109	85	100	1	FALSE
9	6	CharizardM	Fire	Dragon	78	130	111	130	85	100	1	FALSE
10	6	CharizardM	Fire	Flying	78	104	78	159	115	100	1	FALSE
11	7	Squirtle	Water		44	48	65	50	64	43	1	FALSE
12	8	Wartortle	Water		59	63	80	65	80	58	1	FALSE
13	9	Blastoise	Water		79	83	100	85	105	78	1	FALSE
14	9	BlastoiseM	Water		79	103	120	135	115	78	1	FALSE
15	10	Caterpie	Bug		45	30	35	20	20	45	1	FALSE
16	11	Metapod	Bug		50	20	55	25	25	30	1	FALSE
17	12	Butterfree	Bug	Flying	60	45	50	90	80	70	1	FALSE
18	13	Weedle	Bug	Poison	40	35	30	20	20	50	1	FALSE
19	14	Kakuna	Bug	Poison	45	25	50	25	25	35	1	FALSE
20	15	Beedrill	Bug	Poison	65	90	40	45	80	75	1	FALSE
21	15	BeedrillMe	Bug	Poison	65	150	40	15	80	145	1	FALSE
22	16	Pidgey	Normal	Flying	40	45	40	35	35	56	1	FALSE
23	17	Pidgeotto	Normal	Flying	63	60	55	50	50	71	1	FALSE
24	18	Pidgeot	Normal	Flying	83	80	75	70	70	101	1	FALSE
25	18	PidgeotMe	Normal	Flying	83	80	80	135	80	121	1	FALSE
26	19	Rattata	Normal		30	56	35	25	35	72	1	FALSE
27	20	Raticate	Normal		55	81	60	50	70	97	1	FALSE
28	21	Spearow	Normal	Flying	40	60	30	31	31	70	1	FALSE
29	22	Fearow	Normal	Flying	65	90	65	61	61	100	1	FALSE

Pandas

```python
import pandas as pd

df = pd.read_csv('pokemon_data.csv')

print(df)
```

```
test
C:\Users\nyles\AppData\Local\Programs\Python\Python38\python.exe "C:/Users/
       #                   Name   Type 1  ... Speed  Generation  Legendary
0      1              Bulbasaur    Grass  ...    45           1      False
1      2                Ivysaur    Grass  ...    60           1      False
2      3               Venusaur    Grass  ...    80           1      False
3      3  VenusaurMega Venusaur    Grass  ...    80           1      False
4      4             Charmander     Fire  ...    65           1      False
..   ...                    ...      ...  ...   ...         ...        ...
795  719                Diancie     Rock  ...    50           6       True
796  719    DiancieMega Diancie     Rock  ...   110           6       True
797  720    HoopaHoopa Confined  Psychic  ...    70           6       True
798  720     HoopaHoopa Unbound  Psychic  ...    80           6       True
799  721              Volcanion     Fire  ...    70           6       True

[800 rows x 12 columns]
```

Pandas

```python
import pandas as pd

df = pd.read_csv('pokemon_data.csv')

print(df.head())
```

```
test
C:\Users\nyles\AppData\Local\Programs\Python\Python38\python.exe "C:/Us
   #                  Name Type 1  ... Speed  Generation  Legendary
0  1             Bulbasaur  Grass  ...    45           1      False
1  2               Ivysaur  Grass  ...    60           1      False
2  3               Venusaur  Grass  ...    80           1      False
3  3  VenusaurMega Venusaur  Grass  ...    80           1      False
4  4            Charmander   Fire  ...    65           1      False

[5 rows x 12 columns]

Process finished with exit code 0
```

```python
import pandas as pd

df = pd.read_csv('pokemon_data.csv')

print(df.tail())
```

```
test
C:\Users\nyles\AppData\Local\Programs\Python\Python38\python.exe "C:/Users/
      #                  Name   Type 1  ... Speed  Generation  Legendary
795  719              Diancie     Rock  ...    50           6       True
796  719  DiancieMega Diancie     Rock  ...   110           6       True
797  720  HoopaHoopa Confined  Psychic  ...    70           6       True
798  720   HoopaHoopa Unbound  Psychic  ...    80           6       True
799  721            Volcanion     Fire  ...    70           6       True

[5 rows x 12 columns]

Process finished with exit code 0
```

Pandas

#	Name	Type 1	Type 2	HP	Attack	Defense	Sp. Atk	Sp. Def	Speed	Generation	Legendary		
1	Bulbasaur	Grass	Poison	45	49	49	65	65	45	1	FALSE		
2	Ivysaur	Grass	Poison	60	62	63	80	80	60	1	FALSE		
3	Venusaur	Grass	Poison	80	82	83	100	100	80	1	FALSE		
3	VenusaurMega Venusaur	Grass	Poison	80	100	123	122	120	80	1	FALSE		
4	Charmander	Fire		39	52	43	60	50	65	1	FALSE		
5	Charmeleon	Fire		58	64	58	80	65	80	1	FALSE		
6	Charizard	Fire	Flying	78	84	78	109	85	100	1	FALSE		
6	CharizardMega Charizard X	Fire	Dragon	78	130	111	130	85	100	1	FALSE		
6	CharizardMega Charizard Y	Fire	Flying	78	104	78	159	115	100	1	FALSE		
7	Squirtle	Water		44	48	65	50	64	43	1	FALSE		
8	Wartortle	Water		59	63	80	65	80	58	1	FALSE		
9	Blastoise	Water		79	83	100	85	105	78	1	FALSE		
9	BlastoiseMega Blastoise	Water		79	103	120	135	115	78	1	FALSE		
10	Caterpie	Bug		45	30	35	20	20	45	1	FALSE		
11	Metapod	Bug		50	20	55	25	25	30	1	FALSE		
12	Butterfree	Bug	Flying	60	45	50	90	80	70	1	FALSE		
13	Weedle	Bug	Poison	40	35	30	20	20	50	1	FALSE		
14	Kakuna	Bug	Poison	45	25	50	25	25	35	1	FALSE		

Pandas

```python
import pandas as pd

df = pd.read_csv('pokemon_data.txt')

print(df.head())
```

```
test
C:\Users\nyles\AppData\Local\Programs\Python\Python38\python.exe "C:/
  #\tName\tType 1\tType 2\tHP\tAttack\tDefense\tSp. Atk\tSp. Def\tSpe
0  1\tBulbasaur\tGrass\tPoison\t45\t49\t49\t65\t6...
1  2\tIvysaur\tGrass\tPoison\t60\t62\t63\t80\t80\...
2  3\tVenusaur\tGrass\tPoison\t80\t82\t83\t100\t1...
3  3\tVenusaurMega Venusaur\tGrass\tPoison\t80\t1...
4  4\tCharmander\tFire\t\t39\t52\t43\t60\t50\t65\...

Process finished with exit code 0
```

```python
import pandas as pd

df = pd.read_csv('pokemon_data.txt', delimiter='\t')

print(df.head())
```

```
test
C:\Users\nyles\AppData\Local\Programs\Python\Python38\python.exe "C:/U
   #                   Name Type 1  ... Speed  Generation  Legendary
0  1              Bulbasaur  Grass  ...    45           1      False
1  2                Ivysaur  Grass  ...    60           1      False
2  3                Venusaur  Grass  ...    80           1      False
3  3  VenusaurMega Venusaur  Grass  ...    80           1      False
4  4              Charmander   Fire  ...    65           1      False

[5 rows x 12 columns]

Process finished with exit code 0
```

Pandas

```python
import pandas as pd

df = pd.read_csv('pokemon_data.csv')

print(df.columns)
```

```
test
C:\Users\nyles\AppData\Local\Programs\Python\Python38\python.exe "C:/Users/nyl
Index(['#', 'Name', 'Type 1', 'Type 2', 'HP', 'Attack', 'Defense', 'Sp. Atk',
       'Sp. Def', 'Speed', 'Generation', 'Legendary'],
      dtype='object')

Process finished with exit code 0
```

```python
import pandas as pd

df = pd.read_csv('pokemon_data.csv')

print(df["Name"])
```

```
test
C:\Users\nyles\AppData\Local\Programs\Python\Python38\p
0                  Bulbasaur
1                    Ivysaur
2                   Venusaur
3      VenusaurMega Venusaur
4                 Charmander
              ...
795                  Diancie
796      DiancieMega Diancie
797      HoopaHoopa Confined
798       HoopaHoopa Unbound
799                Volcanion
Name: Name, Length: 800, dtype: object

Process finished with exit code 0
```

Pandas

```python
import pandas as pd

df = pd.read_csv('pokemon_data.csv')

print(df["Name"][0:5])
```

```
test
C:\Users\nyles\AppData\Local\Programs\Python\Pyth
0                Bulbasaur
1                  Ivysaur
2                 Venusaur
3    VenusaurMega Venusaur
4               Charmander
Name: Name, dtype: object

Process finished with exit code 0
```

```python
import pandas as pd

df = pd.read_csv('pokemon_data.csv')

print(df[["Name", 'Type 1', 'HP']])
```

```
test
C:\Users\nyles\AppData\Local\Programs\Python\Python38
                      Name   Type 1  HP
0                Bulbasaur    Grass  45
1                  Ivysaur    Grass  60
2                 Venusaur    Grass  80
3    VenusaurMega Venusaur    Grass  80
4               Charmander     Fire  39
..                     ...      ...  ..
795               Diancie     Rock  50
796    DiancieMega Diancie     Rock  50
797    HoopaHoopa Confined  Psychic  80
798     HoopaHoopa Unbound  Psychic  80
799              Volcanion     Fire  80

[800 rows x 3 columns]

Process finished with exit code 0
```

Pandas

```
import pandas as pd

df = pd.read_csv('pokemon_data.csv')

print(df.iloc[0:4])
```

```
test
C:\Users\nyles\AppData\Local\Programs\Python\Python38\python.exe "C:/U
   #                   Name Type 1  ... Speed  Generation  Legendary
0  1              Bulbasaur  Grass  ...    45           1      False
1  2                Ivysaur  Grass  ...    60           1      False
2  3               Venusaur  Grass  ...    80           1      False
3  3  VenusaurMega Venusaur  Grass  ...    80           1      False

[4 rows x 12 columns]

Process finished with exit code 0
```

```
import pandas as pd

df = pd.read_csv('pokemon_data.csv')

print(df.iloc[2, 1])
```

```
test
C:\Users\nyles\AppData\Local\Programs\Pyt
Venusaur

Process finished with exit code 0
```

Pandas

```python
import pandas as pd

df = pd.read_csv('pokemon_data.csv')

print(df.loc[df['Type 1'] == 'Fire'])
```

```
test
C:\Users\nyles\AppData\Local\Programs\Python\Python38\python.exe "C:/Users/ny
      #                    Name Type 1  ... Speed  Generation  Legendary
4     4              Charmander   Fire  ...    65           1      False
5     5              Charmeleon   Fire  ...    80           1      False
6     6               Charizard   Fire  ...   100           1      False
7     6  CharizardMega Charizard X   Fire  ...   100           1      False
8     6  CharizardMega Charizard Y   Fire  ...   100           1      False
42   37                  Vulpix   Fire  ...    65           1      False
43   38               Ninetales   Fire  ...   100           1      False
63   58              Growlithe   Fire  ...    60           1      False
64   59               Arcanine   Fire  ...    95           1      False
83   77                 Ponyta   Fire  ...    90           1      False
84   78               Rapidash   Fire  ...   105           1      False
```

Pandas - Quick Recap

- Pandas is a library in Python that is great for data loading and manipulation
  - Data can be loaded from a csv or txt file
    - Others are available, but these are the most efficient types of files to use
  - Load your data into what is called a “DataFrame”
    - Basically just a fancy multi-dimensional array
  - Common to name your dataframe variable **df**
  - **.head()** and **.tail()** will return first and last 5 values, respectively
- Access certain columns and rows of your data using array access syntax
  - Notice that in our previous examples, accessing a certain column looks very similar to how we would access arrays (square brackets, splicing, etc.)
- Use **iloc** to access rows that have integer indices
  - Use **loc** for rows that use anything else; most commonly strings

Pandas

```python
import pandas as pd

df = pd.read_csv('pokemon_data.csv')

print(df.describe())
```

```
test
C:\Users\nyles\AppData\Local\Programs\Python\Python38\python.exe "C:/Users/nyles/Pych
                #          HP      Attack  ...     Sp. Def       Speed  Generation
count  800.000000  800.000000  800.000000  ...  800.000000  800.000000   800.00000
mean   362.813750   69.258750   79.001250  ...   71.902500   68.277500     3.32375
std    208.343798   25.534669   32.457366  ...   27.828916   29.060474     1.66129
min      1.000000    1.000000    5.000000  ...   20.000000    5.000000     1.00000
25%    184.750000   50.000000   55.000000  ...   50.000000   45.000000     2.00000
50%    364.500000   65.000000   75.000000  ...   70.000000   65.000000     3.00000
75%    539.250000   80.000000  100.000000  ...   90.000000   90.000000     5.00000
max    721.000000  255.000000  190.000000  ...  230.000000  180.000000     6.00000

[8 rows x 8 columns]
```

Pandas

```python
import pandas as pd

df = pd.read_csv('pokemon_data.csv')

print(df.sort_values('Name'))
```

```
test
C:\Users\nyles\AppData\Local\Programs\Python\Python38\python.exe "C:/Users/nyle
       #                    Name   Type 1  ... Speed  Generation  Legendary
510  460               Abomasnow    Grass  ...    60           4      False
511  460  AbomasnowMega Abomasnow    Grass  ...    30           4      False
68    63                    Abra  Psychic  ...    90           1      False
392  359                  Absol     Dark  ...    75           3      False
393  359         AbsolMega Absol     Dark  ...   115           3      False
..   ...                     ...      ...  ...   ...         ...        ...
632  571                Zoroark     Dark  ...   105           5      False
631  570                  Zorua     Dark  ...    65           5      False
46    41                  Zubat   Poison  ...    55           1      False
695  634               Zweilous     Dark  ...    58           5      False
794  718        Zygarde50% Forme   Dragon  ...    95           6       True

[800 rows x 12 columns]

Process finished with exit code 0
```

```python
import pandas as pd

df = pd.read_csv('pokemon_data.csv')

print(df.sort_values('Name', ascending=False))
```

```
test
C:\Users\nyles\AppData\Local\Programs\Python\Python38\python.exe "C:/Users/nyl
       #                    Name   Type 1  ... Speed  Generation  Legendary
794  718        Zygarde50% Forme   Dragon  ...    95           6       True
695  634               Zweilous     Dark  ...    58           5      False
46    41                  Zubat   Poison  ...    55           1      False
631  570                  Zorua     Dark  ...    65           5      False
632  571                Zoroark     Dark  ...   105           5      False
..   ...                     ...      ...  ...   ...         ...        ...
393  359         AbsolMega Absol     Dark  ...   115           3      False
392  359                  Absol     Dark  ...    75           3      False
68    63                    Abra  Psychic  ...    90           1      False
511  460  AbomasnowMega Abomasnow    Grass  ...    30           4      False
510  460               Abomasnow    Grass  ...    60           4      False

[800 rows x 12 columns]

Process finished with exit code 0
```

Pandas

```python
import pandas as pd
pd.set_option('display.max_columns', None)
pd.set_option('display.width', None)
df = pd.read_csv('pokemon_data.csv')
print(df.sort_values(['Type 1', 'HP']))
```

```
test
C:\Users\nyles\AppData\Local\Programs\Python\Python38\python.exe "C:/Users/nyles/PycharmProjects/Research C
       #       Name Type 1  Type 2   HP  Attack  Defense  Sp. Atk  Sp. Def  Speed  Generation  Legendary
316  292   Shedinja    Bug   Ghost    1      90       45       30       30     40           3      False
230  213    Shuckle    Bug    Rock   20      10      230       10      230      5           2      False
462  415     Combee    Bug  Flying   30      30       42       30       42     70           4      False
603  543   Venipede    Bug  Poison   30      45       59       30       39     57           5      False
314  290    Nincada    Bug  Ground   31      45       90       30       30     40           3      False
..   ...        ...    ...     ...  ...     ...      ...      ...      ...    ...         ...        ...
142  131     Lapras  Water     Ice  130      85       80       85       95     60           1      False
145  134   Vaporeon  Water     NaN  130      65       60      110       95     65           1      False
350  320    Wailmer  Water     NaN  130      70       35       70       35     60           3      False
655  594  Alomomola  Water     NaN  165      75       80       40       45     65           5      False
351  321    Wailord  Water     NaN  170      90       45       90       45     60           3      False

[800 rows x 12 columns]

Process finished with exit code 0
```

Pandas

```python
import pandas as pd
pd.set_option('display.max_columns', None)
pd.set_option('display.width', None)
df = pd.read_csv('pokemon_data.csv')
print(df.sort_values(['Type 1', 'HP'], ascending=[1, 0]))
```

```
test
C:\Users\nyles\AppData\Local\Programs\Python\Python38\python.exe "C:/Users/nyles/PycharmProjects/Research Code/test.py"
       #                   Name Type 1    Type 2  HP  Attack  Defense  Sp. Atk  Sp. Def  Speed  Generation  Legendary
520  469                Yanmega    Bug    Flying  86      76       86      116       56     95           4      False
698  637              Volcarona    Bug      Fire  85      60       65      135      105    100           5      False
231  214              Heracross    Bug  Fighting  80     125       75       40       95     85           2      False
232  214  HeracrossMega Heracross    Bug  Fighting  80     185      115       40      105     75           2      False
678  617               Accelgor    Bug       NaN  80      70       40      100       60    145           5      False
..   ...                    ...    ...       ...  ..     ...      ...      ...      ...    ...         ...        ...
106   98                Krabby  Water       NaN  30     105       90       25       25     50           1      False
125  116                Horsea  Water       NaN  30      40       70       70       25     60           1      False
129  120                Staryu  Water       NaN  30      45       55       70       55     85           1      False
139  129              Magikarp  Water       NaN  20      10       55       15       20     80           1      False
381  349                Feebas  Water       NaN  20      15       20       10       55     80           3      False

[800 rows x 12 columns]

Process finished with exit code 0
```

Pandas

```python
import pandas as pd
pd.set_option('display.max_columns', None)
pd.set_option('display.width', None)
df = pd.read_csv('pokemon_data.csv')
df['Total'] = df['HP'] + df['Attack'] + df['Defense'] + df['Sp. Atk'] + df['Sp. Def'] + df['Speed']
print(df.head())
```

```
test
C:\Users\nyles\AppData\Local\Programs\Python\Python38\python.exe "C:/Users/nyles/PycharmProjects/Research Code/test.py"
   #                  Name Type 1  Type 2  HP  Attack  Defense  Sp. Atk  Sp. Def  Speed  Generation  Legendary  Total
0  1             Bulbasaur  Grass  Poison  45      49       49       65       65     45           1      False    318
1  2               Ivysaur  Grass  Poison  60      62       63       80       80     60           1      False    405
2  3              Venusaur  Grass  Poison  80      82       83      100      100     80           1      False    525
3  3  VenusaurMega Venusaur  Grass  Poison  80     100      123      122      120     80           1      False    625
4  4            Charmander   Fire     NaN  39      52       43       60       50     65           1      False    309

Process finished with exit code 0
```

Pandas

```
import pandas as pd
pd.set_option('display.max_columns', None)
pd.set_option('display.width', None)
df = pd.read_csv('pokemon_data.csv')
df['Total'] = df.iloc[:, 4:10].sum(axis=1)
print(df.head())
```

test

```
C:\Users\nyles\AppData\Local\Programs\Python\Python38\python.exe "C:/Users/nyles/PycharmProjects/Research Code/test.py"
   #                   Name Type 1  Type 2  HP  Attack  Defense  Sp. Atk  Sp. Def  Speed  Generation  Legendary  Total
0  1              Bulbasaur  Grass  Poison  45      49       49       65       65     45           1      False    318
1  2                Ivysaur  Grass  Poison  60      62       63       80       80     60           1      False    405
2  3               Venusaur  Grass  Poison  80      82       83      100      100     80           1      False    525
3  3  VenusaurMega Venusaur  Grass  Poison  80     100      123      122      120     80           1      False    625
4  4             Charmander   Fire     NaN  39      52       43       60       50     65           1      False    309

Process finished with exit code 0
```

Pandas

```python
import pandas as pd
pd.set_option('display.max_columns', None)
pd.set_option('display.width', None)
df = pd.read_csv('pokemon_data.csv')
df['Total'] = df.iloc[:, 4:10].sum(axis=1)
print(df.head())
df.to_csv('pokemon_data.csv')
```

	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1		#	Name	Type 1	Type 2	HP	Attack	Defense	Sp. Atk	Sp. Def	Speed	Generatio	Legendary	Total
2	0	1	Bulbasaur	Grass	Poison	45	49	49	65	65	45	1	FALSE	318
3	1	2	Ivysaur	Grass	Poison	60	62	63	80	80	60	1	FALSE	405
4	2	3	Venusaur	Grass	Poison	80	82	83	100	100	80	1	FALSE	525
5	3	3	VenusaurM	Grass	Poison	80	100	123	122	120	80	1	FALSE	625
6	4	4	Charmand	Fire		39	52	43	60	50	65	1	FALSE	309
7	5	5	Charmelec	Fire		58	64	58	80	65	80	1	FALSE	405
8	6	6	Charizard	Fire	Flying	78	84	78	109	85	100	1	FALSE	534
9	7	6	CharizardM	Fire	Dragon	78	130	111	130	85	100	1	FALSE	634
10	8	6	CharizardM	Fire	Flying	78	104	78	159	115	100	1	FALSE	634

Pandas

```python
import pandas as pd
pd.set_option('display.max_columns', None)
pd.set_option('display.width', None)
df = pd.read_csv('pokemon_data.csv')
df['Total'] = df.iloc[:, 4:10].sum(axis=1)
print(df.head())
df.to_csv('pokemon_data.csv', index=False)
```

1	#	Name	Type 1	Type 2	HP	Attack	Defense	Sp. Atk	Sp. Def	Speed	Generatio	Legendary	Total
2	1	Bulbasaur	Grass	Poison	45	49	49	65	65	45	1	FALSE	273
3	2	Ivysaur	Grass	Poison	60	62	63	80	80	60	1	FALSE	345
4	3	Venusaur	Grass	Poison	80	82	83	100	100	80	1	FALSE	445
5	3	VenusaurM	Grass	Poison	80	100	123	122	120	80	1	FALSE	545
6	4	Charmand	Fire		39	52	43	60	50	65	1	FALSE	244
7	5	Charmelec	Fire		58	64	58	80	65	80	1	FALSE	325
8	6	Charizard	Fire	Flying	78	84	78	109	85	100	1	FALSE	434
9	6	CharizardM	Fire	Dragon	78	130	111	130	85	100	1	FALSE	534
10	6	CharizardM	Fire	Flying	78	104	78	159	115	100	1	FALSE	534

Pandas - Quick Recap 2

- DataFrame has methods (just like strings) for various situations as seen in the examples
- You can easily add your own rows or columns of data as well
- As we learn more about DataFrames, does it make sense how it may be easier to change or load data using code rather than a non-programming method? (i.e Excel)
- It’s okay if you know nothing about your data (to an extent)

Pandas

```python
import pandas as pd
pd.set_option('display.max_columns', None)
pd.set_option('display.width', None)
df = pd.read_csv('pokemon_data.csv')
print(df.loc[(df['Type 1'] == 'Grass') & (df['Type 2'] == 'Poison')])
```

test

```
C:\Users\nyles\AppData\Local\Programs\Python\Python38\python.exe "C:/Users/nyles/PycharmProjects/Research Code/test.py"
     Unnamed: 0    #                   Name Type 1  Type 2   HP  Attack  Defense  Sp. Atk  Sp. Def  Speed  Generation  Legendary  Total
0             0    1              Bulbasaur  Grass  Poison   45      49       49       65       65     45           1      False    273
1             1    2                Ivysaur  Grass  Poison   60      62       63       80       80     60           1      False    345
2             2    3               Venusaur  Grass  Poison   80      82       83      100      100     80           1      False    445
3             3    3  VenusaurMega Venusaur  Grass  Poison   80     100      123      122      120     80           1      False    545
48           48   43                 Oddish  Grass  Poison   45      50       55       75       65     30           1      False    290
49           49   44                  Gloom  Grass  Poison   60      65       70       85       75     40           1      False    355
50           50   45              Vileplume  Grass  Poison   75      80       85      110       90     50           1      False    440
75           75   69             Bellsprout  Grass  Poison   50      75       35       70       30     40           1      False    260
76           76   70             Weepinbell  Grass  Poison   65      90       50       85       45     55           1      False    335
77           77   71             Victreebel  Grass  Poison   80     105       65      100       70     70           1      False    420
344         344  315                Roselia  Grass  Poison   50      60       45      100       80     65           3      False    335
451         451  406                  Budew  Grass  Poison   40      30       35       50       70     55           4      False    225
452         452  407                Roserade  Grass  Poison   60      70       65      125      105     90           4      False    425
651         651  590                Foongus  Grass  Poison   69      55       45       55       55     15           5      False    279
652         652  591              Amoonguss  Grass  Poison  114      85       70       85       80     30           5      False    434

Process finished with exit code 0
```

Pandas

```python
import pandas as pd
pd.set_option('display.max_columns', None)
pd.set_option('display.width', None)
df = pd.read_csv('pokemon_data.csv')
print(df.loc[(df['Type 1'] == 'Grass') | (df['Type 2'] == 'Poison')])
```

```
test
C:\Users\nyles\AppData\Local\Programs\Python\Python38\python.exe "C:/Users/nyles/PycharmProjects/Research Code/test.py"
     Unnamed: 0    #                  Name Type 1    Type 2   HP  Attack  Defense  Sp. Atk  Sp. Def  Speed  Generation  Legendary  Total
0             0    1             Bulbasaur  Grass    Poison   45      49       49       65       65     45           1      False    273
1             1    2               Ivysaur  Grass    Poison   60      62       63       80       80     60           1      False    345
2             2    3              Venusaur  Grass    Poison   80      82       83      100      100     80           1      False    445
3             3    3  VenusaurMega Venusaur  Grass    Poison   80     100      123      122      120     80           1      False    545
16           16   13                Weedle    Bug    Poison   40      35       30       20       20     50           1      False    145
..          ...  ...                   ...    ...       ...  ...     ...      ...      ...      ...    ...         ...        ...    ...
718         718  650               Chespin  Grass       NaN   56      61       65       48       45     38           6      False    275
719         719  651              Quilladin  Grass       NaN   61      78       95       56       58     57           6      False    348
720         720  652            Chesnaught  Grass  Fighting   88     107      122       74       75     64           6      False    466
740         740  672                Skiddo  Grass       NaN   66      65       48       62       57     52           6      False    298
741         741  673                Gogoat  Grass       NaN  123     100       62       97       81     68           6      False    463

[89 rows x 14 columns]

Process finished with exit code 0
```

Pandas

```python
import pandas as pd
pd.set_option('display.max_columns', None)
pd.set_option('display.width', None)
df = pd.read_csv('pokemon_data.csv')
print(df.loc[(df['Type 1'] == 'Grass') & (df['Type 2'] == 'Poison') & (df['HP'] > 70)])
```

test

```
C:\Users\nyles\AppData\Local\Programs\Python\Python38\python.exe "C:/Users/nyles/PycharmProjects/Research Code/test.py"
     Unnamed: 0    #                  Name Type 1  Type 2   HP  Attack  Defense  Sp. Atk  Sp. Def  Speed  Generation  Legendary  Total
2             2    3              Venusaur  Grass  Poison   80      82       83      100      100     80           1      False    445
3             3    3  VenusaurMega Venusaur  Grass  Poison   80     100      123      122      120     80           1      False    545
50           50   45             Vileplume  Grass  Poison   75      80       85      110       90     50           1      False    440
77           77   71            Victreebel  Grass  Poison   80     105       65      100       70     70           1      False    420
652         652  591             Amoonguss  Grass  Poison  114      85       70       85       80     30           5      False    434

Process finished with exit code 0
```

Pandas

```python
import pandas as pd
pd.set_option('display.max_columns', None)
pd.set_option('display.width', None)
df = pd.read_csv('pokemon_data.csv')
print(df.loc[df['Name'].str.contains('Mega')])
```

test

```
C:\Users\nyles\AppData\Local\Programs\Python\Python38\python.exe "C:/Users/nyles/PycharmProjects/Research Code/test.py"
     Unnamed: 0    #                    Name    Type 1    Type 2   HP  Attack  Defense  Sp. Atk  Sp. Def  Speed  Generation  Legendary  Total
3             3    3     VenusaurMega Venusaur     Grass    Poison   80     100      123      122      120     80           1      False    545
7             7    6  CharizardMega Charizard X      Fire    Dragon   78     130      111      130       85    100           1      False    534
8             8    6  CharizardMega Charizard Y      Fire    Flying   78     104       78      159      115    100           1      False    534
12           12    9   BlastoiseMega Blastoise     Water       NaN   79     103      120      135      115     78           1      False    552
19           19   15     BeedrillMega Beedrill       Bug    Poison   65     150       40       15       80    145           1      False    350
23           23   18       PidgeotMega Pidgeot    Normal    Flying   83      80       80      135       80    121           1      False    458
71           71   65     AlakazamMega Alakazam   Psychic       NaN   55      50       65      175       95    150           1      False    440
87           87   80       SlowbroMega Slowbro     Water   Psychic   95      75      180      130       80     30           1      False    560
102         102   94         GengarMega Gengar     Ghost    Poison   60      65       80      170       95    130           1      False    470
124         124  115  KangaskhanMega Kangaskhan    Normal       NaN  105     125      100       60      100    100           1      False    490
```

Pandas

```python
import pandas as pd
pd.set_option('display.max_columns', None)
pd.set_option('display.width', None)
df = pd.read_csv('pokemon_data.csv')
print(df.loc[~df['Name'].str.contains('Mega')])
```

```
test
C:\Users\nyles\AppData\Local\Programs\Python\Python38\python.exe "C:/Users/nyles/PycharmProjects/Research Code/test.py"
     Unnamed: 0    #                  Name   Type 1  Type 2   HP  Attack  Defense  Sp. Atk  Sp. Def  Speed  Generation  Legendary  Total
0             0    1             Bulbasaur    Grass  Poison   45      49       49       65       65     45           1      False    273
1             1    2               Ivysaur    Grass  Poison   60      62       63       80       80     60           1      False    345
2             2    3              Venusaur    Grass  Poison   80      82       83      100      100     80           1      False    445
4             4    4            Charmander     Fire     NaN   39      52       43       60       50     65           1      False    244
5             5    5            Charmeleon     Fire     NaN   58      64       58       80       65     80           1      False    325
..          ...  ...                   ...      ...     ...  ...     ...      ...      ...      ...    ...         ...        ...    ...
794         794  718     Zygarde50% Forme   Dragon  Ground  108     100      121       81       95     95           6       True    505
795         795  719               Diancie     Rock   Fairy   50     100      150      100      150     50           6       True    550
797         797  720  HoopaHoopa Confined  Psychic   Ghost   80     110       60      150      130     70           6       True    530
798         798  720   HoopaHoopa Unbound  Psychic    Dark   80     160       60      170      130     80           6       True    600
799         799  721             Volcanion     Fire   Water   80     110      120      130       90     70           6       True    530

[751 rows x 14 columns]

Process finished with exit code 0
```

Pandas

```python
import pandas as pd
pd.set_option('display.max_columns', None)
pd.set_option('display.width', None)
df = pd.read_csv('pokemon_data.csv')
df.loc[df['Type 1'] == 'Fire', 'Type 1'] = 'Flames'
print(df)
```

```
test
C:\Users\nyles\AppData\Local\Programs\Python\Python38\python.exe "C:/Users/nyles/PycharmProjects/Research Code/test.py"
     Unnamed: 0    #                   Name   Type 1  Type 2  HP  Attack  Defense  Sp. Atk  Sp. Def  Speed  Generation  Legendary  Total
0             0    1              Bulbasaur    Grass  Poison  45      49       49       65       65     45           1      False    273
1             1    2                Ivysaur    Grass  Poison  60      62       63       80       80     60           1      False    345
2             2    3               Venusaur    Grass  Poison  80      82       83      100      100     80           1      False    445
3             3    3  VenusaurMega Venusaur    Grass  Poison  80     100      123      122      120     80           1      False    545
4             4    4             Charmander   Flames     NaN  39      52       43       60       50     65           1      False    244
..          ...  ...                    ...      ...     ...  ..     ...      ...      ...      ...    ...         ...        ...    ...
795         795  719                Diancie     Rock   Fairy  50     100      150      100      150     50           6       True    550
796         796  719    DiancieMega Diancie     Rock   Fairy  50     160      110      160      110    110           6       True    590
797         797  720    HoopaHoopa Confined  Psychic   Ghost  80     110       60      150      130     70           6       True    530
798         798  720     HoopaHoopa Unbound  Psychic    Dark  80     160       60      170      130     80           6       True    600
799         799  721              Volcanion   Flames   Water  80     110      120      130       90     70           6       True    530

[800 rows x 14 columns]

Process finished with exit code 0
```

Pandas

```python
import pandas as pd
pd.set_option('display.max_columns', None)
pd.set_option('display.width', None)
df = pd.read_csv('pokemon_data.csv')
df.loc[df['Type 1'] == 'Fire', 'Legendary'] = True
print(df)
```

```
test
C:\Users\nyles\AppData\Local\Programs\Python\Python38\python.exe "C:/Users/nyles/PycharmProjects/Research Code/test.py"
     Unnamed: 0    #                   Name   Type 1  Type 2  HP  Attack  Defense  Sp. Atk  Sp. Def  Speed  Generation  Legendary  Total
0             0    1              Bulbasaur    Grass  Poison  45      49       49       65       65     45           1      False    273
1             1    2                Ivysaur    Grass  Poison  60      62       63       80       80     60           1      False    345
2             2    3               Venusaur    Grass  Poison  80      82       83      100      100     80           1      False    445
3             3    3  VenusaurMega Venusaur    Grass  Poison  80     100      123      122      120     80           1      False    545
4             4    4             Charmander     Fire     NaN  39      52       43       60       50     65           1       True    244
..          ...  ...                    ...      ...     ...  ..     ...      ...      ...      ...    ...         ...        ...    ...
795         795  719                Diancie     Rock   Fairy  50     100      150      100      150     50           6       True    550
796         796  719    DiancieMega Diancie     Rock   Fairy  50     160      110      160      110    110           6       True    590
797         797  720    HoopaHoopa Confined  Psychic   Ghost  80     110       60      150      130     70           6       True    530
798         798  720     HoopaHoopa Unbound  Psychic    Dark  80     160       60      170      130     80           6       True    600
799         799  721              Volcanion     Fire   Water  80     110      120      130       90     70           6       True    530

[800 rows x 14 columns]

Process finished with exit code 0
```

Pandas

```python
import pandas as pd
pd.set_option('display.max_columns', None)
pd.set_option('display.width', None)
df = pd.read_csv('pokemon_data.csv')
df.loc[df['Type 1'] == 'Fire', ['Attack' , 'Defense']] = 99
print(df)
```

test

```
C:\Users\nyles\AppData\Local\Programs\Python\Python38\python.exe "C:/Users/nyles/PycharmProjects/Research Code/test.py"
     Unnamed: 0    #                   Name   Type 1  Type 2  HP  Attack  Defense  Sp. Atk  Sp. Def  Speed  Generation  Legendary  Total
0             0    1              Bulbasaur    Grass  Poison  45      49       49       65       65     45           1      False    273
1             1    2                Ivysaur    Grass  Poison  60      62       63       80       80     60           1      False    345
2             2    3               Venusaur    Grass  Poison  80      82       83      100      100     80           1      False    445
3             3    3  VenusaurMega Venusaur    Grass  Poison  80     100      123      122      120     80           1      False    545
4             4    4             Charmander     Fire     NaN  39      99       99       60       50     65           1      False    244
..          ...  ...                    ...      ...     ...  ..     ...      ...      ...      ...    ...         ...        ...    ...
795         795  719                Diancie     Rock   Fairy  50     100      150      100      150     50           6       True    550
796         796  719    DiancieMega Diancie     Rock   Fairy  50     160      110      160      110    110           6       True    590
797         797  720    HoopaHoopa Confined  Psychic   Ghost  80     110       60      150      130     70           6       True    530
798         798  720     HoopaHoopa Unbound  Psychic    Dark  80     160       60      170      130     80           6       True    600
799         799  721              Volcanion     Fire   Water  80      99       99      130       90     70           6       True    530

[800 rows x 14 columns]

Process finished with exit code 0
```

Pandas

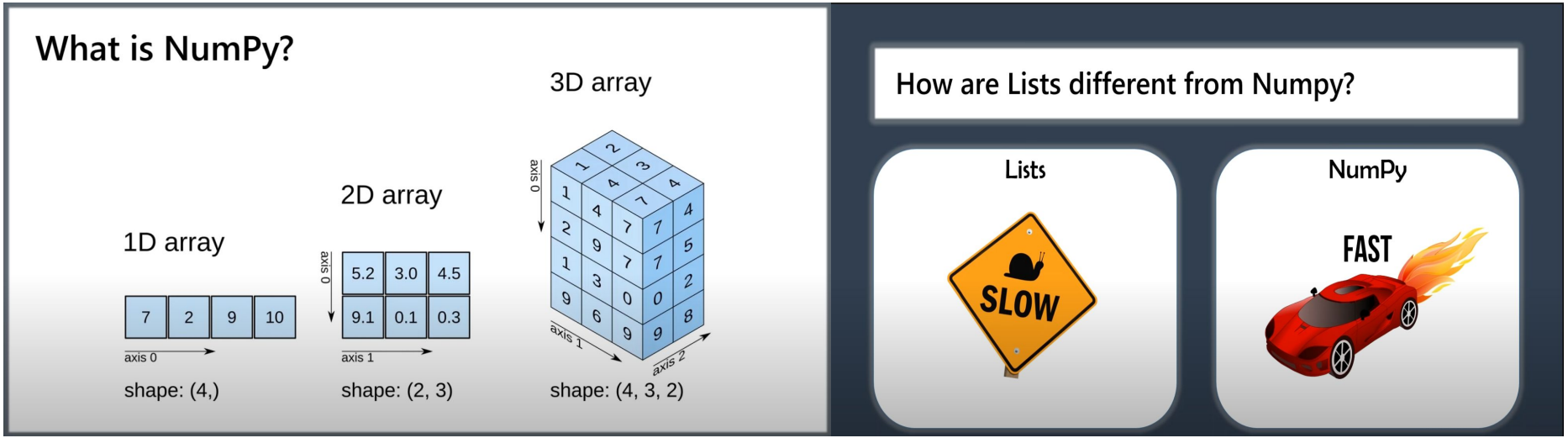
```python
import pandas as pd
pd.set_option('display.max_columns', None)
pd.set_option('display.width', None)
df = pd.read_csv('pokemon_data.csv')
df.loc[df['Type 1'] == 'Fire', ['Attack' , 'Legendary']] = [99, 'True']
print(df)
```

```
test
C:\Users\nyles\AppData\Local\Programs\Python\Python38\python.exe "C:/Users/nyles/PycharmProjects/Research Code/test.py"
     Unnamed: 0    #                   Name   Type 1  Type 2  HP  Attack  Defense  Sp. Atk  Sp. Def  Speed  Generation Legendary  Total
0             0    1              Bulbasaur    Grass  Poison  45      49       49       65       65     45           1     False    273
1             1    2                Ivysaur    Grass  Poison  60      62       63       80       80     60           1     False    345
2             2    3               Venusaur    Grass  Poison  80      82       83      100      100     80           1     False    445
3             3    3  VenusaurMega Venusaur    Grass  Poison  80     100      123      122      120     80           1     False    545
4             4    4             Charmander     Fire     NaN  39      99       43       60       50     65           1      True    244
..          ...  ...                    ...      ...     ...  ..     ...      ...      ...      ...    ...         ...       ...    ...
795         795  719               Diancie     Rock   Fairy  50     100      150      100      150     50           6      True    550
796         796  719    DiancieMega Diancie     Rock   Fairy  50     160      110      160      110    110           6      True    590
797         797  720    HoopaHoopa Confined  Psychic   Ghost  80     110       60      150      130     70           6      True    530
798         798  720     HoopaHoopa Unbound  Psychic    Dark  80     160       60      170      130     80           6      True    600
799         799  721              Volcanion     Fire   Water  80      99      120      130       90     70           6      True    530

[800 rows x 14 columns]

Process finished with exit code 0
```

Pandas - Final Recap

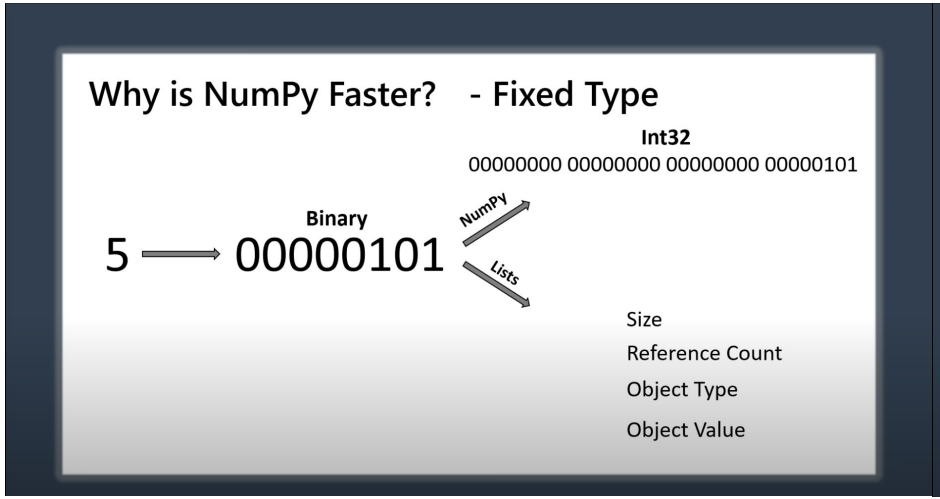
- We can use conditionals to access specific data
  - Only want pokemon with HP > 70, Grass as their Type 1, etc.
- We can also use conditionals to mutate or add data
- It can be tough to keep track of all the syntax!
  - If you are struggling, go back and practice accessing lists and multi-dimensional lists

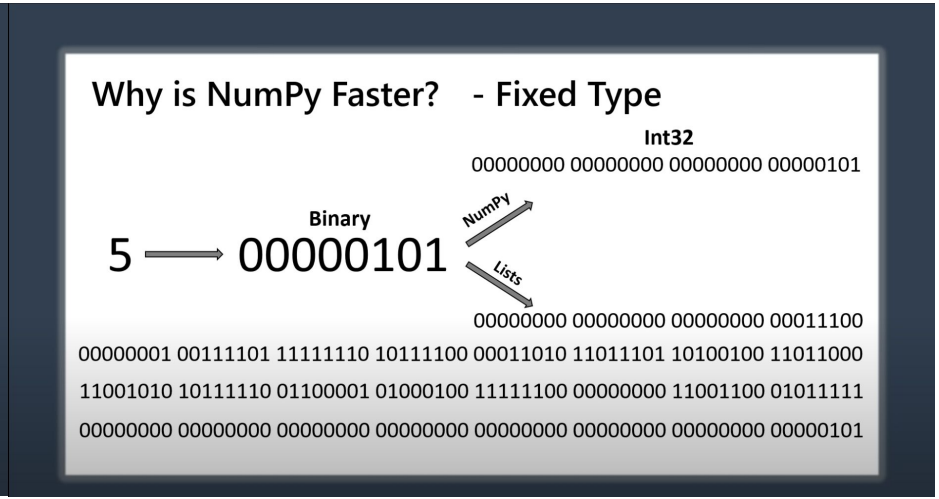
NumPy



(Credit to Keith Galli)

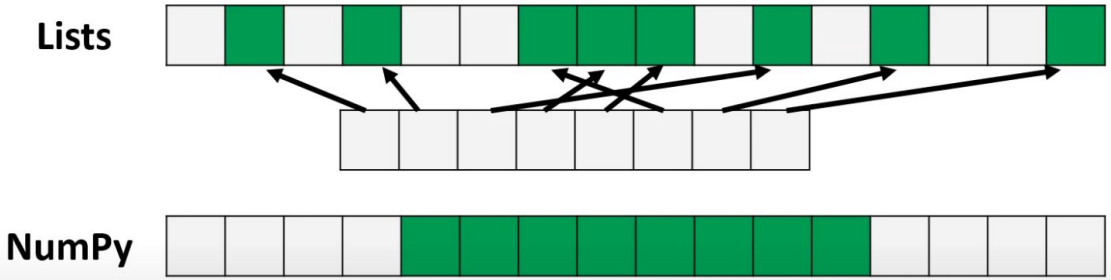
NumPy


Why is NumPy Faster? - Fixed Type
Int32
00000000 00000000 00000000 00000101
Binary
5
00000101
NumPy
Lists
Size
Reference Count
Object Type
Object Value


Why is NumPy Faster? - Fixed Type
Int32
00000000 00000000 00000000 00000101
Binary
5
00000101
NumPy
Lists
00000000 00000000 00000000 00011100
00000001 00111101 11111110 10111100 00011010 11011101 10100100 11011000
11001010 10111110 01100001 01000100 11111100 00000000 11001100 01011111
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000101

NumPy

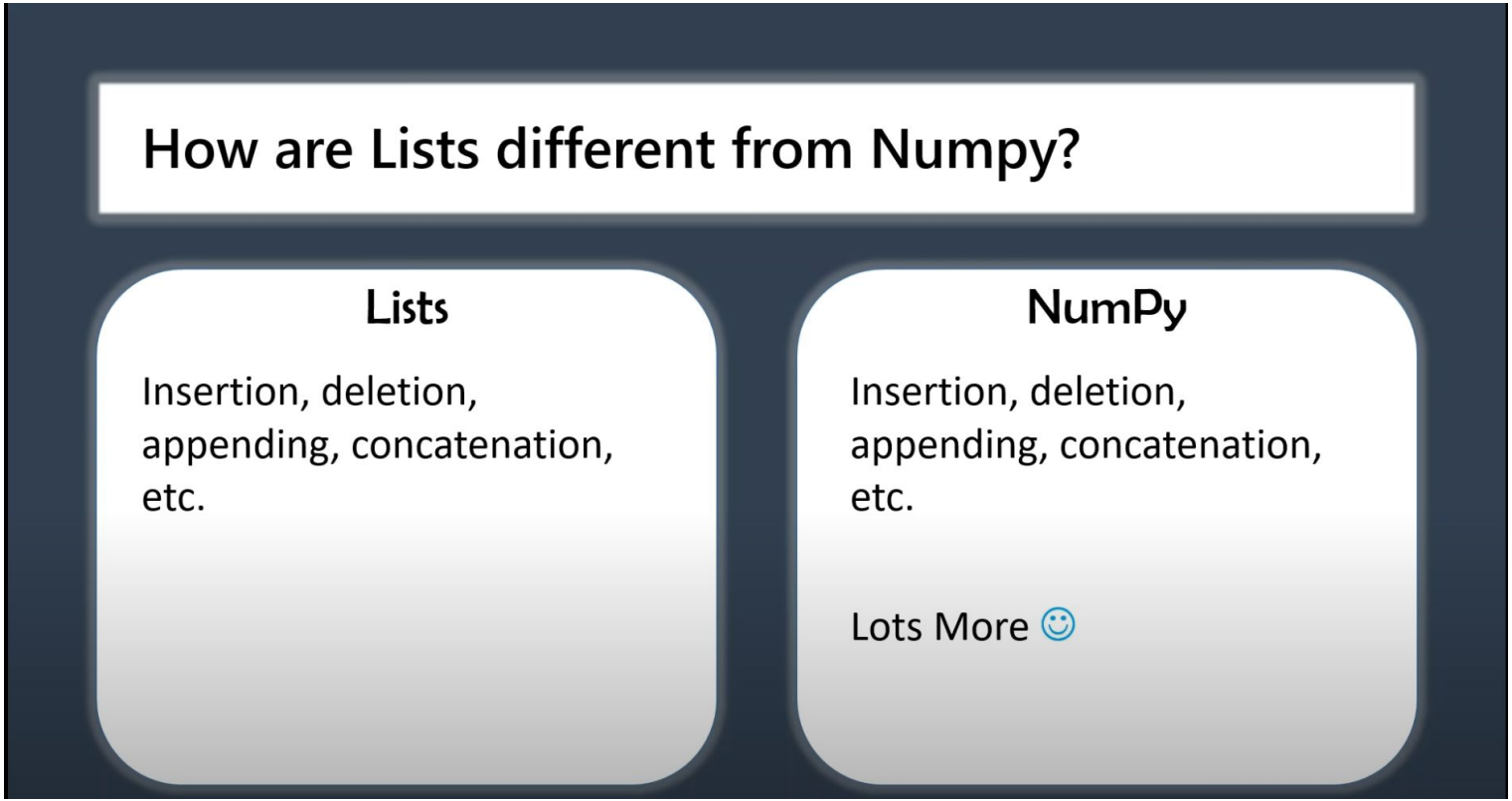
Why is NumPy Faster? - Contiguous Memory



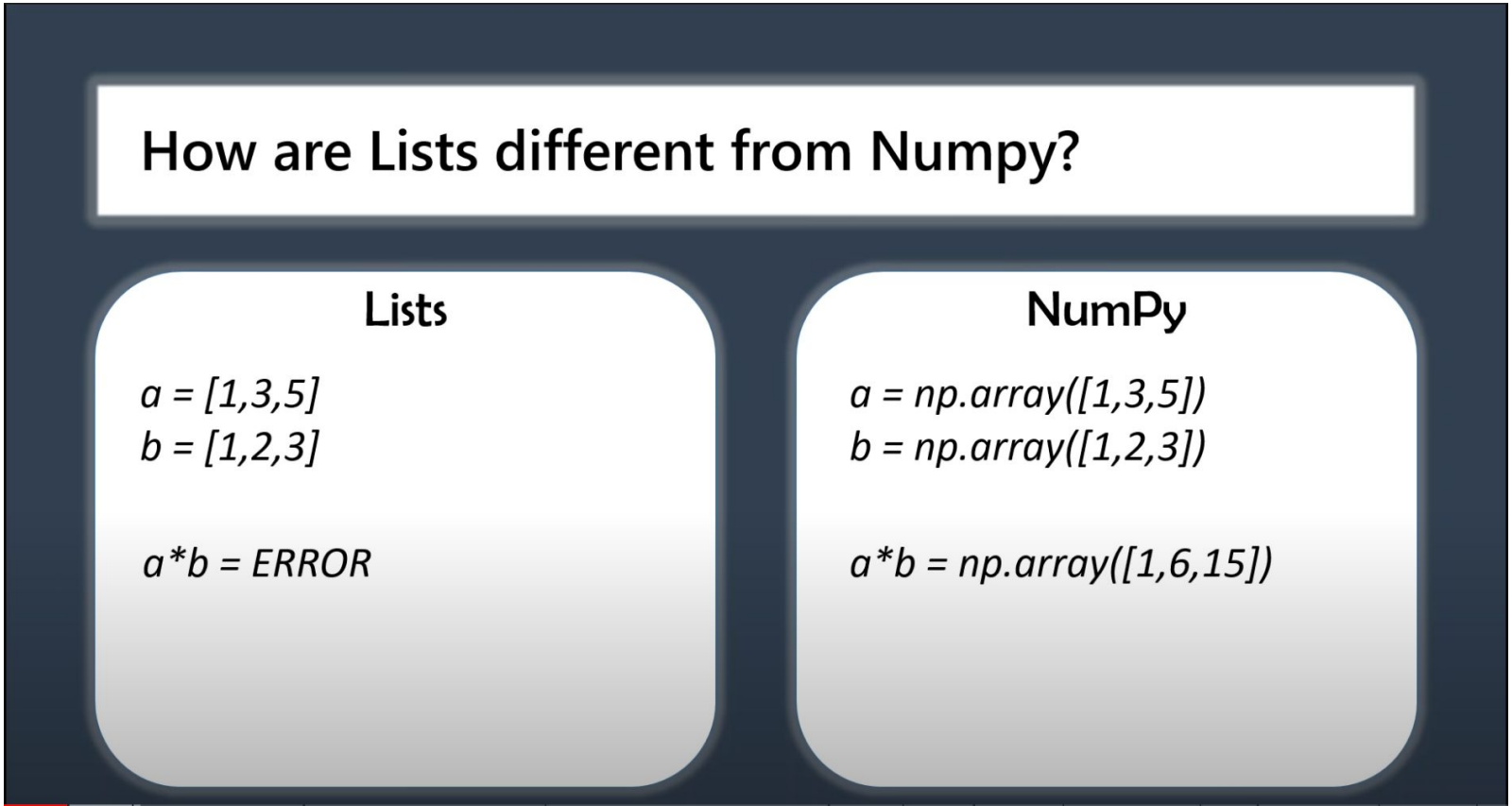
Benefits:

- SIMD Vector Processing
- Effective Cache Utilization

NumPy


How are Lists different from Numpy?
Lists
Insertion, deletion, appending, concatenation, etc.
NumPy
Insertion, deletion, appending, concatenation, etc.
Lots More ☺

NumPy


How are Lists different from Numpy?
Lists
a = [1,3,5]
b = [1,2,3]
a*b = ERROR
NumPy
a = np.array([1,3,5])
b = np.array([1,2,3])
a*b = np.array([1,6,15])

NumPy

Applications of NumPy?

Mathematics (MATLAB Replacement)

Plotting (Matplotlib)

Backend (Pandas, Connect 4, Digital Photography)

Machine Learning

NumPy - Recap

- NumPy is a library that optimizes and improves multi-dimensional list operations in Python
  - As mentioned before, lists are extremely common in ML, hence why NumPy is so prevalent in ML
- NumPy is used over regular lists because of its increased speed and efficient memory use
- NumPy also support element-wise operations, something that regular lists do not possess
  - Shown in example

NumPy

```python
import numpy as np

arr = np.array([1, 2, 3])
print("Dimension: ", arr.ndim)
print("Shape: ", arr.shape)
print("Element type: ", arr.dtype)
```

```
test
C:\Users\nyles\AppData\Local\Programs\Pytho
Dimension:  1
Shape:  (3,)
Element type:  int32

Process finished with exit code 0
```

```python
import numpy as np

arr = np.array([[1, 2, 3], [4, 5, 6]], dtype='int16')
print("Dimension: ", arr.ndim)
print("Shape: ", arr.shape)
print("Element type: ", arr.dtype)
```

```
test
C:\Users\nyles\AppData\Local\Programs\Python\Python38\python.exe "C:/U
Dimension:  2
Shape:  (2, 3)
Element type:  int16

Process finished with exit code 0
```

NumPy

```python
import numpy as np

arr = np.array([[1, 2, 3, 4, 5, 6, 7],
                [8, 9, 10, 11, 12, 13, 14]])
print("Dimension: ", arr.ndim)
print("Shape: ", arr.shape)
print("Element type: ", arr.dtype)
print(arr)
print(arr[1, 5])
```

```
test
C:\Users\nyles\AppData\Local\Programs\Python\Pyth
Dimension:  2
Shape:  (2, 7)
Element type:  int32
[[ 1  2  3  4  5  6  7]
 [ 8  9 10 11 12 13 14]]
13

Process finished with exit code 0
```

```python
import numpy as np

arr = np.array([[1, 2, 3, 4, 5, 6, 7],
                [8, 9, 10, 11, 12, 13, 14]])
print(arr)
print()
print(arr[0, :])
```

```
test
C:\Users\nyles\AppData\Local\Programs\Python\Python38\
[[ 1  2  3  4  5  6  7]
 [ 8  9 10 11 12 13 14]]

[1 2 3 4 5 6 7]

Process finished with exit code 0
```

NumPy

```python
import numpy as np

arr = np.array([[1, 2, 3, 4, 5, 6, 7],
                [8, 9, 10, 11, 12, 13, 14]])
print(arr)
print()
print(arr[:, 1])
```

```
test
C:\Users\nyles\AppData\Local\Programs\Python\Python38
[[ 1  2  3  4  5  6  7]
 [ 8  9 10 11 12 13 14]]

[2 9]

Process finished with exit code 0
```

```python
import numpy as np

arr = np.array([[1, 2, 3, 4, 5, 6, 7],
                [8, 9, 10, 11, 12, 13, 14]])
print(arr)
print()
print(arr[0, 1:7:2])
```

```
test
C:\Users\nyles\AppData\Local\Programs\Python\Python38\pytho
[[ 1  2  3  4  5  6  7]
 [ 8  9 10 11 12 13 14]]

[2 4 6]

Process finished with exit code 0
```

NumPy

```python
import numpy as np

arr = np.array([[1, 2, 3, 4, 5, 6, 7],
                [8, 9, 10, 11, 12, 13, 14]])
print(arr)
print()
arr[1, 5] = 20
print(arr)
```

```
test
C:\Users\nyles\AppData\Local\Programs\Python\Pyth
[[ 1  2  3  4  5  6  7]
 [ 8  9 10 11 12 13 14]]

[[ 1  2  3  4  5  6  7]
 [ 8  9 10 11 12 20 14]]

Process finished with exit code 0
```

```python
import numpy as np

arr = np.array([[1, 2, 3, 4, 5, 6, 7],
                [8, 9, 10, 11, 12, 13, 14]])
print(arr)
print()
arr[1, :] = 5
print(arr)
```

```
test
C:\Users\nyles\AppData\Local\Programs\Python\Python
[[ 1  2  3  4  5  6  7]
 [ 8  9 10 11 12 13 14]]

[[1 2 3 4 5 6 7]
 [5 5 5 5 5 5 5]]

Process finished with exit code 0
```

NumPy

```python
import numpy as np

arr = np.array([[[1, 2], [3,4]], [[5, 6], [7, 8]]])
print(arr)
```

```
test
C:\Users\nyles\AppData\Local\Programs\Python\Python38\python.exe "C:
[[[1 2]
  [3 4]]

 [[5 6]
  [7 8]]]


Process finished with exit code 0
```

```python
import numpy as np

arr = np.array([[[1, 2], [3,4]], [[5, 6], [7, 8]]])
print(arr)
print()
print(arr[0, 1, 1])
```

```
test
C:\Users\nyles\AppData\Local\Programs\Python\Python38\pyt
[[[1 2]
  [3 4]]

 [[5 6]
  [7 8]]]


4

Process finished with exit code 0
```

NumPy

```python
import numpy as np

arr = np.array([[[1, 2], [3,4]], [[5, 6], [7, 8]]])
print(arr[:, 1, :])
print()
arr[:, 1, :] = [[9, 9], [8, 8]]
print(arr)
```

```
test
C:\Users\nyles\AppData\Local\Programs\Python\Python38\pyt
[[3 4]
 [7 8]]

[[[1 2]
  [9 9]]

 [[5 6]
  [8 8]]]

Process finished with exit code 0
```

```python
import numpy as np

arr = np.array([[[1, 2], [3,4]], [[5, 6], [7, 8]]])
print(arr[:, 1, :])
print()
arr[:, 1, :] = [[9, 9, 9], [8, 8]]
print(arr)
```

```
test
C:\Users\nyles\AppData\Local\Programs\Python\Python38\python.exe "C:/Users/nyles/Pycharm
[[3 4]
 [7 8]]

TypeError: int() argument must be a string, a bytes-like object or a number, not 'list'

The above exception was the direct cause of the following exception:

ValueError: setting an array element with a sequence.

The above exception was the direct cause of the following exception:

Traceback (most recent call last):
  File "C:/Users/nyles/PycharmProjects/Research Code/test.py", line 6, in <module>
    arr[:, 1, :] = [[9, 9, 9], [8, 8]]
ValueError: setting an array element with a sequence.

Process finished with exit code 1
```

NumPy

```python
import numpy as np

arr = np.zeros((5, 2))
print(arr)
```

```
test
C:\Users\nyles\AppData\Local\Program
[[0. 0.]
 [0. 0.]
 [0. 0.]
 [0. 0.]
 [0. 0.]]

Process finished with exit code 0
```

```python
import numpy as np

arr = np.ones((5, 2, 3))
print(arr)
```

```
test
C:\Users\nyles\AppData\Local\Programs\Pyt
[[[1. 1. 1.]
  [1. 1. 1.]]

 [[1. 1. 1.]
  [1. 1. 1.]]

 [[1. 1. 1.]
  [1. 1. 1.]]

 [[1. 1. 1.]
  [1. 1. 1.]]

 [[1. 1. 1.]
  [1. 1. 1.]]]

Process finished with exit code 0
```

```python
import numpy as np

arr = np.full((5, 2, 3), 100)
print(arr)
```

```
test
C:\Users\nyles\AppData\Local\Programs\Python
[[[100 100 100]
  [100 100 100]]

 [[100 100 100]
  [100 100 100]]

 [[100 100 100]
  [100 100 100]]

 [[100 100 100]
  [100 100 100]]

 [[100 100 100]
  [100 100 100]]]

Process finished with exit code 0
```

NumPy

```python
import numpy as np

arr = np.random.rand(4, 2, 3)
print(arr)
```

```
test
C:\Users\nyles\AppData\Local\Programs\Python
[[[0.95563073 0.98350015 0.18840445]
  [0.69477148 0.38309721 0.55460786]]

 [[0.32534254 0.09514864 0.32851864]
  [0.48583419 0.16042843 0.16100699]]

 [[0.7519215  0.32832585 0.3510535 ]
  [0.11500454 0.43908896 0.24945232]]

 [[0.56203616 0.9149402  0.58440399]
  [0.52138457 0.95230466 0.267391  ]]]

Process finished with exit code 0
```

```python
import numpy as np

arr = np.random.randint(7, size=(4, 2, 3))
print(arr)
```

```
test
C:\Users\nyles\AppData\Local\Programs\Python\Python38\python.ex
[[[5 6 3]
  [3 6 1]]

 [[1 4 1]
  [6 6 0]]

 [[6 5 2]
  [6 1 6]]

 [[3 1 5]
  [6 4 3]]]

Process finished with exit code 0
```

NumPy

```python
import numpy as np

arr = np.random.randint(2, 7, size=(4, 2, 3))
print(arr)
```

```
test
C:\Users\nyles\AppData\Local\Programs\Python\Python38\pyt
[[[5 6 2]
  [6 5 2]]

 [[2 3 3]
  [5 2 5]]

 [[5 2 2]
  [4 5 2]]

 [[6 2 6]
  [2 3 5]]]

Process finished with exit code 0
```

NumPy - Recap 2

- After seeing these examples, we can see how similar the syntax between NumPy and Python lists is
  - They can be randomly generated the same, accessed the same, mutated the same, etc.
  - Subtle differences like having to be aware of dimensions and shape
- Notice that we can also use some NumPy methods to create certain desirable lists
  - Lists will all 0's, lists with all 1's, lists whose elements are all the same (all of these are shown in the examples)
- Again, if dimensions and accessing gets confusing because of all the brackets, try to practice simpler examples with lists and come back when you are feeling more confident

NumPy - Checkpoint

Can you create this list? (There are multiple ways to do this)

```
[[1. 1. 1. 1. 1.]
 [1. 0. 0. 0. 1.]
 [1. 0. 9. 0. 1.]
 [1. 0. 0. 0. 1.]
 [1. 1. 1. 1. 1.]]
```

1	1	1	1	1
1	0	0	0	1
1	0	9	0	1
1	0	0	0	1
1	1	1	1	1

NumPy - Checkpoint Solution

1	1	1	1	1
1	0	0	0	1
1	0	9	0	1
1	0	0	0	1
1	1	1	1	1

```python
import numpy as np

output = np.ones((5, 5))

z = np.zeros((3, 3))
z[1, 1] = 9

output[1:4, 1:4] = z
print(output)
```

```
test
C:\Users\nyles\AppData\Local\Progra
[[1. 1. 1. 1. 1.]
 [1. 0. 0. 0. 1.]
 [1. 0. 9. 0. 1.]
 [1. 0. 0. 0. 1.]
 [1. 1. 1. 1. 1.]]

Process finished with exit code 0
```

Caution!

```
import numpy as np

a = np.array([1, 2, 3])
b = a
b[0] = 2
print(b)
print(a)
```

```
test
C:\Users\nyles\AppData\Local\Prograr
[2 2 3]
[2 2 3]

Process finished with exit code 0
```

```
import numpy as np

a = np.array([1, 2, 3])
b = a.copy()
b[0] = 2
print(b)
print(a)
```

```
test
C:\Users\nyles\AppData\Local\Program
[2 2 3]
[1 2 3]

Process finished with exit code 0
```

Element-wise Manipulation

```python
import numpy as np

a = np.array([1, 2, 3, 4])

print("a + 2 = ", a+2)
print("a - 2 = ", a-2)
print("a * 2 = ", a*2)
print("a / 2 = ", a/2)
b = np.array([1, 2, 3, 4])
print(f"b = {b}, a+b = {a+b}")
print("a^2 = ", a**2)
print("sin(a) = ", np.sin(a))
```

```
test
C:\Users\nyles\AppData\Local\Programs\Python\Python38\python.
a + 2 =  [3 4 5 6]
a - 2 =  [-1  0  1  2]
a * 2 =  [2 4 6 8]
a / 2 =  [0.5 1.  1.5 2. ]
b = [1 2 3 4], a+b = [2 4 6 8]
a^2 =  [ 1  4  9 16]
sin(a) =  [ 0.84147098  0.90929743  0.14112001 -0.7568025 ]

Process finished with exit code 0
```

```python
import numpy as np

a = [1, 2, 3, 4]

print("a + 2 = ", a+2)
print("a - 2 = ", a-2)
print("a * 2 = ", a*2)
print("a / 2 = ", a/2)
b = [1, 2, 3, 4]
print(f"b = {b}, a+b = {a+b}")
print("a^2 = ", a**2)
print("sin(a) = ", np.sin(a))
```

```
test
C:\Users\nyles\AppData\Local\Programs\Python\Python38\python.exe "C:/Users/nyles/Pych
Traceback (most recent call last):
  File "C:/Users/nyles/PycharmProjects/Research Code/test.py", line 5, in <module>
    print("a + 2 = ", a+2)
TypeError: can only concatenate list (not "int") to list

Process finished with exit code 1
```

NumPy

```python
import numpy as np

a = np.array([[1, 2, 3, 4], [5, 6, 7, 8]])
print(a)
print()
b = a.reshape(1, 8)
print(b)
```

```
test
C:\Users\nyles\AppData\Local\Programs\Python
[[1 2 3 4]
 [5 6 7 8]]

[[1 2 3 4 5 6 7 8]]

Process finished with exit code 0
```

```python
import numpy as np

a = np.array([[1, 2, 3, 4], [5, 6, 7, 8]])
print(a)
print()
b = a.reshape((2, 2, 2))
print(b)
```

```
test
C:\Users\nyles\AppData\Local\Programs\Python\Python38\python.e
[[1 2 3 4]
 [5 6 7 8]]

[[[1 2]
  [3 4]]

 [[5 6]
  [7 8]]]

Process finished with exit code 0
```

Checkpoint - Accessing Matrices

1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20
21	22	23	24	25
26	27	28	29	30

Checkpoint - Accessing Matrices

1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20
21	22	23	24	25
26	27	28	29	30

```python
import numpy as np

a = np.arange(1, 31).reshape(6, 5)
print(a)
print()
print(a[2:4, :2])
```

```
test
C:\Users\nyles\AppData\Local\Programs\Python
[[ 1  2  3  4  5]
 [ 6  7  8  9 10]
 [11 12 13 14 15]
 [16 17 18 19 20]
 [21 22 23 24 25]
 [26 27 28 29 30]]

[[11 12]
 [16 17]]

Process finished with exit code 0
```

Checkpoint - Accessing Matrices

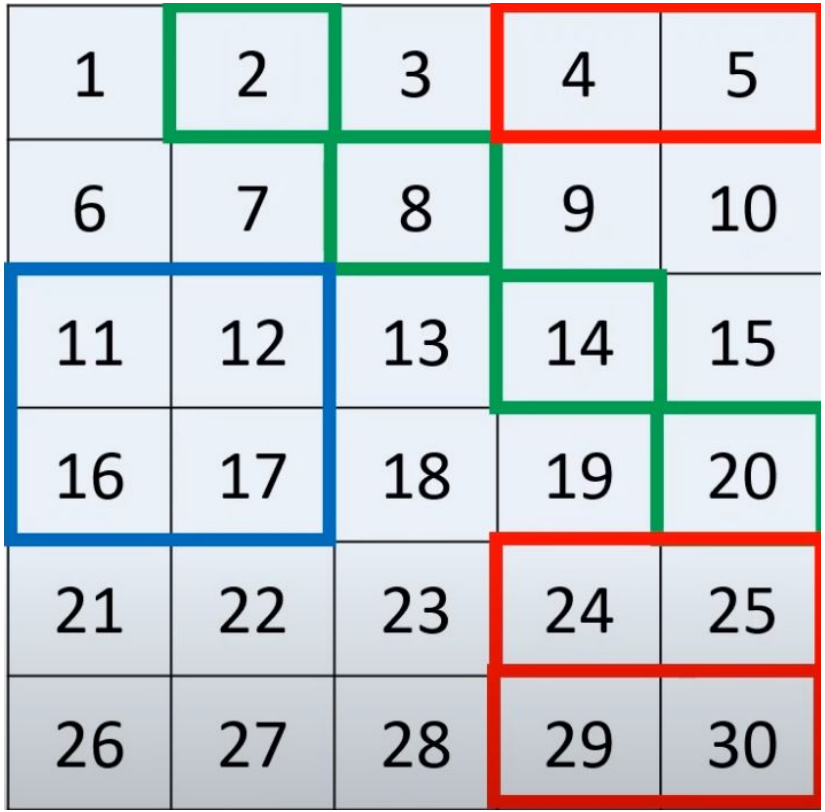
Checkpoint - Accessing Matrices

1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20
21	22	23	24	25
26	27	28	29	30

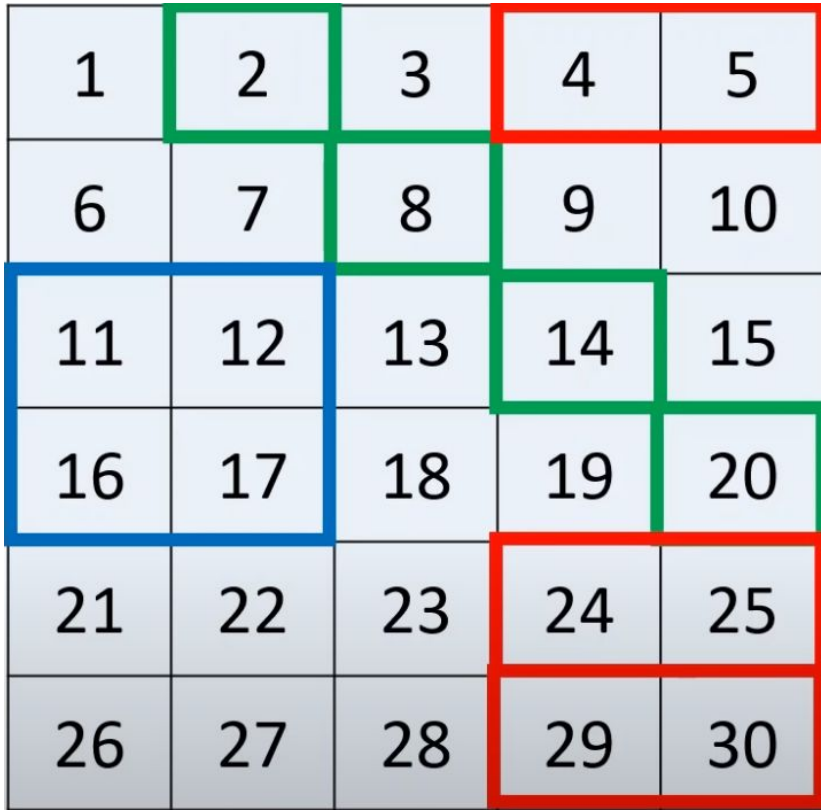
```python
import numpy as np

a = np.arange(1, 31).reshape(6, 5)
print(a)
print()
print(a[[0, 1, 2, 3], [1, 2, 3, 4]])
```

```
test
C:\Users\nyles\AppData\Local\Programs\Pytho
[[ 1  2  3  4  5]
 [ 6  7  8  9 10]
 [11 12 13 14 15]
 [16 17 18 19 20]
 [21 22 23 24 25]
 [26 27 28 29 30]]

[ 2  8 14 20]

Process finished with exit code 0
```

Checkpoint - Accessing Matrices



Checkpoint - Accessing Matrices




```python
import numpy as np

a = np.arange(1, 31).reshape(6, 5)
print(a)
print()
print(a[[0, 4, 5], 3:])
```

```
test
C:\Users\nyles\AppData\Local\Programs\Python\
[[ 1  2  3  4  5]
 [ 6  7  8  9 10]
 [11 12 13 14 15]
 [16 17 18 19 20]
 [21 22 23 24 25]
 [26 27 28 29 30]]

[[ 4  5]
 [24 25]
 [29 30]]

Process finished with exit code 0
```

Putting It All Together

```python
import numpy as np
import pandas as pd

df = pd.read_csv('pokemon_data.csv')
data = df.to_numpy()
print(f'{data} \n {type(data)}')
```

```
test
C:\Users\nyles\AppData\Local\Programs\Python\Pyt
[[0 1 'Bulbasaur' ... 1 False 273]
 [1 2 'Ivysaur' ... 1 False 345]
 [2 3 'Venusaur' ... 1 False 445]
 ...
 [797 720 'HoopaHoopa Confined' ... 6 True 530]
 [798 720 'HoopaHoopa Unbound' ... 6 True 600]
 [799 721 'Volcanion' ... 6 True 530]]
 <class 'numpy.ndarray'>

Process finished with exit code 0
```

- Loading data into a dataframe and formatting it into a NumPy list only takes 2(!) lines of code
  - Now we are ready to feed this list into some ML algorithms!
  - First, we will need to learn about how ML works in the first place, but that will come tomorrow...

End of Day 1!

Practice Questions Section 4 (optional)