Week 1:

Binary: represented by 1 and 0

1. Computer programming: the process of giving instructions to a computer to perform an action or set of actions
2. Programming Languages: the words and symbols we use to write instructions for computers to follow

Python was named after a British comedian troupe

Library: a reusable collections of code

1. SYNTAX: the structure of *code words*, *symbols*, *placement*, and *punctuation*.

* VARIABLES: a named container which stores values in a reserved location in the computer’s memory. a container’s contents
* OBJECTS:

“+” is addition

“/” is division

“\*\*” is exponent

“==” is to check equivalence

“=” : single equal sign are reserved for assigning variables.

* These are called “operator”

# Discover more about Python

A screenshot of a computer

Description automatically generated

* “**Print to the console”:** print function will output whatever we enter in its parentheses.
* **“Computations”**

A screenshot of a computer code

Description automatically generated

* **“Assign variables” :** a variable like a container that you can name. A container’s contents are known as its value.
* For ex: variable is called “country”, and its value is “Vietnam”

A screenshot of a computer program

Description automatically generated

* “**evaluate statement”:** check equivalence (True or False)

A computer code with text

Description automatically generated

* **“Conditional statements”:** the value stored in the “age” variable was 30, so the computer returns “adult”

A screenshot of a computer

Description automatically generated

* **“Looping”:** performs the **same action** to each element of something.
* For ex: create a list containing the numbers 1,2,3,10,12 and assign it to a variable named “my list”.

Loop over “my list” and for each number in the list, the computer will print that number divided by 3.

It outputs 0.25, 0.5,….

A screenshot of a computer code

Description automatically generated

* **“Function”:** a chunk of code that can be reused to perform the same task.
* **“An argument”:** info that you give a function in its parentheses.
* For ex: define this function as “is adult”, and it will accept an argument called “age”. In the body of function, we’ll use the same code as we used for the conditional statement.

As we call the function and give it an argument of 10, the computer returns the word “minor”. Now, we can perform this same evaluation as many times as we want.

A white box with black text

Description automatically generated

* **“Built-in Python function”**
* For ex: “sorted” as function, and enter “new list” as its argument.

It returns the list with its values sorted from least to greatest.

A screenshot of a computer program

Description automatically generated

<https://docs.python.org/3/library/functions.html#len>

1. A function: a chunk of code that can be reused to perform the same task.

CONDITIONAL LOGIC: *if…else*

LOOP:

FUNCTION: a chunk of code that can be reused to perform the same task.

ARGUMENT: info that you give a function in its parentheses

* print function: will output whatever we enter in its parentheses.
* Exponent: \*\*

A white grid with yellow text

Description automatically generated

# Jupyter Notebooks

1. Jupyter notebook: an open-source web application for creating and sharing doc containing live code, math formulas, viz, and text

Terminal-based text editor:

Cells: the modular code input/output fields into which Jupyter Notebooks are partitioned.

* Can run a cell w/o necessarily having to run the whole notebook🡪 helpful for data exploration and experimentation.
* Cells are helpful with debugging, bc they provide user-friendly way to make a mistake, and notice it, and iterate back to correct it, w/o having to re-execute a whole script.

Markdowns: let you write formatted text in a coding environment or plain-text editor.

A screenshot of a computer

Description automatically generated

A screenshot of a cell phone

Description automatically generatedA screenshot of a computer program

Description automatically generated

# Object-oriented programming

1. Object-oriented Programming: a programming system that is based around objects, which can contain both data and code that manipulates that data.

Object: an instance of a class; a fundamental building block of Python.

* *Lists, functions, strings…*

Class: an object’s data type that bundles data and functionality together

<!!!> An object needs to have a type- or belong to a class- is because it allows us to build a bunch of useful tools that can be packed directly into the object itself.

A computer code with text

Description automatically generated

* “**hocus pocus**” inside quotations and assign that to a variable called “**magic**”, this variable becomes an instance of the **STRING class.**

Because it belongs to the string class, it behaves in a certain way and has lots of built-in functionality reserved for strings.

A screenshot of a computer program

Description automatically generated

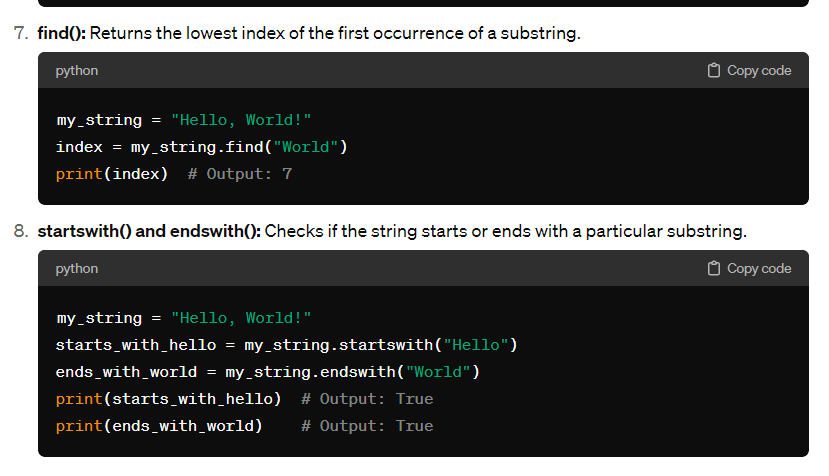
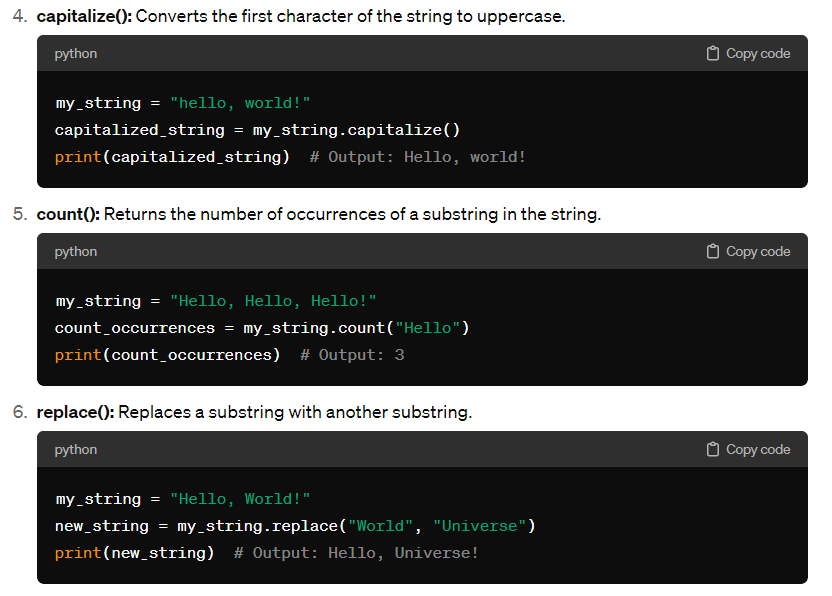
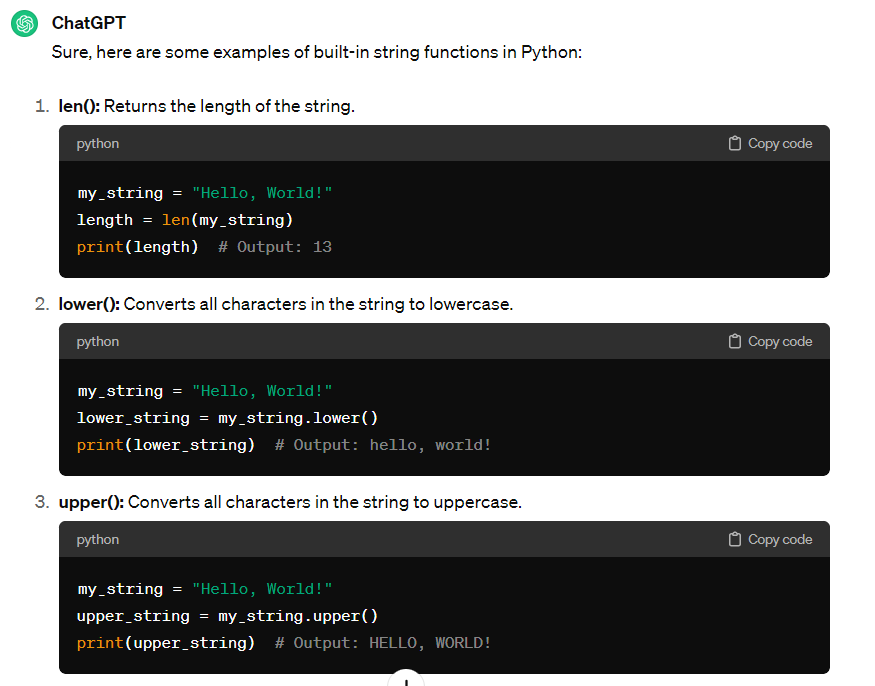
* “**swapcase ( )”:** swap the case of the characters by typing “magic.swapcase()” with empty parentheses after it.
* “**magic.replace ( )”:** We can replace some characters with the new characters by typing “magic.replace” and entering the characters we want to replace and what we want to replace them with.

A white rectangular object with black text

Description automatically generated

🡪 **“magic.split ( )”:** split the string into a list of 2 strings

<!!!> these are called METHOD.



1. Method: a function that belongs to a class and typically performs an action or operation. They use parenthesis ( … )

Dot Notation: how to access the methods and attributes that belong to an instance of a class.

A screen shot of a computer

Description automatically generated

1. Attribute: a value associated with an object or class which is referenced by name using dot notation. They don’t use parentheses ( … )

Attributes 🡪 important for custom-built classes and more complex data structures, like dataframes

A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated

* Have a dataframe called “planets” that contains a row for each planet and columns that represent planet name, its radius, and the number of moons it has.

One attribute of this dataframe would be its shape. This dataframe is 8 rows by 3 columns.

Another attribute of the dataframe class is columns. Calling this attribute on the dataframe object returns an index object containing the column names of the dataframe.

Attributes allow you to access characteristics of a class, but they don’t do anything to it or change it.

<!!!> Attributes vs. Methods

ATTRIBUTES are *characteristics* of the object, while METHODS are *actions* or *operations.*



For example: “Spaceship”

* Attributes might be:
* A white background with black text

  Description automatically generated A close-up of a computer code

  Description automatically generated
* Methods could be:

A close-up of a word

Description automatically generated A close-up of a computer code

Description automatically generated

<!!!> Methods are followed by parentheses, and it’s possible for them to take arguments. For ex: **Spaceship.warp(7)** could change the speed of the ship to warp 7.

A screenshot of a computer program

Description automatically generatedA screenshot of a computer code

Description automatically generated

# Variables and data types

1. Variables in Python are like noun in English. Variables point to values.

For ex: if you have the expression x = 3, then x is the variable, and it’s stored value is 3.

3 exists in specific location in computer’s memory.

X points to that location

/OR another way to think variables is think it as a container with a label on it.

A container is a separate thing from whatever it contains, but if I ask you to pass me the ketchup, you’ll know which bottle to pass me even if you can’t see the ketchup inside it, because the bottle is labeled.

Data type: an attribute that describes a piece of data based on its value, its programming language, or the operations it can perform.

* In Python, it includes: strings, floats, integers, lists, dictionary, etc.

Variable algorithm questions:

* What’s the variable’s name?
* What’s the variable’s type?
* What’s the variable’s starting value?

Assignment: the process of storing a value in a variable

Expression: a combination of numbers, symbols, or other variables that produce a result when evaluated.

Dynamic typing: variables can point to objects of any data type.

* No default types for most new variables🡪 need to assign, initialize, before calling them.

STRING: output of Str has quotation mark ‘….’

**<!!!>**

* when you want to modify the contents of a variable, you have to reassign it.
* The order you run your cells is matters.

The value contained in the “max age” variable changed every time we reassigned it. That’s why it’s dynamic.

A screenshot of a computer

Description automatically generated

* Assign the list to a variable called “age list”.

Notice that we didn’t call it “x”, bc “x” doesn’t tell us anything about the value it contains, and if we encounter an “x” later, we might not remember that it’s a list of ages.

* Use Python’s built-in “max” function and passing age list to it as an argument.
* Assign the result to a new variable call “max age”. When we call the variable, the computer returns a value of 34- an integer.

The “max age” variable contains a value whose data type is “integer”.

A screenshot of a computer

Description automatically generated

* Suppose we want the “max age” variable to contain a string value.

We can convert it to string by using string function, S-T-R, which is indicated by the quotation marks in the output.

A white background with black text

Description automatically generated

* Store variable in text string “ninety-nine”. Now, when we call the variable, the computer returns our new string.

A screenshot of a computer program

Description automatically generated

**<!!!>** a couple things to note here:

* First, notice when we converted “max age” from INTEGER to STRING, we reassigned it back to itself.

If we hadn’t done this, and we had simply used the string function on the “max age” variable, the computer would have returned a string, but the contents of the variable would not have changed.

Generally, when you want to modify the contents of a variable, you have to reassign it.

* Second, the order that you run your cells matters.

For ex, if I rerun the cell where I first assigned the “max age” variable, and then call this variable in a new cell, you’ll notice that its value has reverted back to the Integer 34. It’s no longer the string, “ninety-nine”.

The value contained in the “max age” variable changed every time we reassigned it. That’s why it’s DYNAMIC.

# Create precise variable names

1. Naming conventions: consistent guidelines that describes the content, creation date, and version of a file in its name.

Keyword: a special word that is reversed for a specific purpose and that can only be used for that purpose.

Ex: “for”, “in”, “if”, and “else” -🡪 Should never be used when naming variables.

* “print”, “str”, … 🡪 don’t use

**<!!!>** Naming restrictions for variables:

* Only include letters, numbers, and underscores
* No spaces, tabs, or special characters ($ or &)
* Must start with a letter or underscore
* Case-sensitive
* Capitalized matters
* Cannot include parentheses

Little vocab here 😊

A black text on a white background

Description automatically generated

A close-up of a text

Description automatically generated

* Invalid because: Must start with a letter or underscore
* Invalid because: use special character &

In calculation, Python follows the order of operations. For ex:

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer error

Description automatically generatedA screenshot of a computer error

Description automatically generated

# Data types and conversions

10)

String: a sequence of characters and punctuation that contains textual info.

* Strings are instantiated with single or double quotation marks or the string function.



Immutable data type: a data type in which the values can never be altered or updated. *(bat’ bien’)*

Integer: a data type used to represent whole numbers without fractions

Float: a data type that represent numbers that contain decimals

A screenshot of a computer

Description automatically generated

* **7** being read as INTEGERS while **‘8’** is read as a STRING 🡪 cannot add 7 to a word

1. Python operators: **<!!!>**

<https://python-reference.readthedocs.io/en/latest/docs/operators/index.html>

Python reference library:

<https://docs.python.org/3/library/>

Built-in data types:

<https://docs.python.org/3/library/stdtypes.html>

Built-in functions:

<https://docs.python.org/3/library/functions.html#built-in-functions>

1. You can use the **type()** function to have the computer tell you the data type.

A class is an object’s data type. The class bundles data and functionality together

A screenshot of a computer

Description automatically generated

1. Implicit conversion: python automatically converts one data type to another without user involvement. *(co’ the chuyen doi)*

A white rectangular object with green text

Description automatically generated

* An arithmetic operations involving both integers and floats, the interpreter works in the background and converts integers to floats.
* Don’t have to specify any code to do this.

1. Explicit conversion: users convert the data type of an object to a required data type. *(chuyen doi ro rang)*

* Use to convert numerical values to strings.

Typecasting: the user casts or changes the data type.

A white rectangular object with red numbers and black text

Description automatically generated

* The S-T-R function inside the string we want to have interpreted as Output.

Now, the output of this calc is stored as a STRing.

GLOSSARY:

A screenshot of a computer

Description automatically generatedA white text on a black background

Description automatically generated

[end of Week 1]

Week 2:

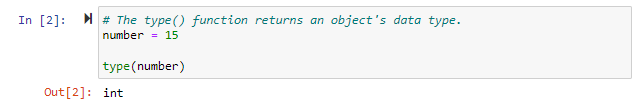
**Define functions and returning values**

1. Function: a body of reusable code for performing specific processes or tasks

* “print” function: writes text on the screen

A white rectangular box with blue and red text

Description automatically generated

* “type” function: tells us the data type contained within a variable
* S-T-R function: converts an object into a string.A white rectangular object with black text

  Description automatically generated

<!!!> the print syntax is a function and requires parentheses, even when there are no argument used, and the parentheses are empty.

1. def: a keyword that defines a function at the start of the function block.

A screenshot of a computer

Description automatically generated

Def 🡪 function’s name (which is greeting???)

🡪 put its parameters/arguments – in parentheses (which is name)

Argument: things you give to the function to modify in some way.

Call them anything you want. Whatever we call them here when we define the fx is how we’ll have to refer to them below in the function’s body.

🡪 ends with colon

1. return: a reserved keyword in Python that makes a function produce new results, which are saved for later use.

A screenshot of a computer

Description automatically generated

Step 1:

Def 🡪 function’s name: area\_triangle

* parameters/arguments- in parentheses: is base and height of a triangle
* return the area of the triangle \*\*\*\*\*

the area is calculated as base times height divided by 2

use “return” to tell Python that this is the value that we want to come out of the fx. In stead of “print”, return lets us store this value in a variable.

* ends with colon

Step 2:

the area is calculated as “base \* height / 2”

* Use return to tell Python, this is the result we want to come out of a function.
* Return helps us to store this value in a variable.

Step 3:

Suppose we have 2 triangles and want to add the sum of both areas.

* First, calc 2 areas separately, store each value in its own named variable.
* Second, add 2 areas together, assign the results to a variable called “total-area”
* Third, call the variable to get the result which is “total\_area”

1. reusability: defining code once and using it many times without having to rewrite it.

A screenshot of a phone

Description automatically generated

* have a fx called as “get\_seconds”
* this fx takes hours, minutes, and seconds as Inputs,

and returns the total number of seconds those inputs represent.

* 1st line: Begin with the keyword “def” and name the fx “get\_seconds”

In parentheses, give 3 parameters: hours, minutes, seconds.

* 2nd line: Computation: calc total number of seconds and assigns that value to a variable of “total\_seconds”
* 3rd line and Final line: return statement that returns the value of “total\_seconds”

1. write clean code: to save time & effort, reduce errors, and enhance teamwork

A screenshot of a computer code

Description automatically generated

* 1st line: use “len” function which returns the length of an object – the number of characters in the string.
* 2nd line: use that length to calc. the “lucky number”
* 3rd line: “print” a message with the name & the number

**“CLEAN” Version:**

**A screenshot of a computer

Description automatically generated**

Modularity: the ability to write code in separate components that work together and that can be reused for other programs ~~~ Reusability – as it allows you to reuse blocks and sections of code.

Refactoring: the process of restructuring code while maintaining its original functionality.

Self-documenting code: code written in a way that is readable and makes its purpose clear.

Comment:

A screenshot of a computer code

Description automatically generated

# Use comments to scaffold your code

# algorithm: a set of instructions for solving a problem or accomplishing a task.

# Best way to approach writing a new function is to break it into small, simple pieces, beginning with the comments.

A screenshot of a computer code

Description automatically generated

Thesis:

* have a square fountain, want to plant grass and a border around that square.
* Question: calculate the amount of grass seeds needed?

Know: the length of the side of fountain (fountain\_side) and the width of grass border (grass\_width)

* Need to calc:

1. Area of fountain

A yellow square with black text

Description automatically generated

Total area is the length of one side of the larger outer square, and squaring it.

1. Total area of the square and the grass border combined

A screenshot of a computer

Description automatically generated

The length of the large square is equal to the width of the border times 2, plus the length of the side of the fountain

1. From (a) and (b)🡪 find: area of grass border

A blue square with green squares

Description automatically generated

1. Amount of seed needed (35 gr/sq. m)
2. Convert gram to kg
3. Docstring: a string at the beginning of a function’s body that summarizes the function’s behavior and explains its arguments and return values.

* Begins and ends with quotation marks
* Can be single quotes or double quotes
* First, write what the function does. It takes the form of a command, and ends in a period.

A screenshot of a computer

Description automatically generatedA screenshot of a computer program

Description automatically generatedA white paper with black text

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated

# Make comparisons using operators

1. Boolean: a data type that has only 2 possible values, usually true or false

Comparators: operators that compare two values and produce Boolean values (True/False)

A screen shot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

1. The plus operator doesn’t work between integers and strings.

For ex: compare int and str

A screenshot of a computer screen

Description automatically generated

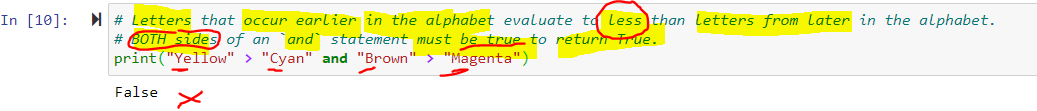
Logical operators: operators that connect multiple statements together and perform more complex comparisons.

A close-up of a white background

Description automatically generated

1. The **and operator** needs both expressions to be true to return a True result.

* When used on strings of text, comparators evaluate the first letter of each string, with A being least and Z being greatest.
* If 2 strings have the same first letter, the second letter will be compared.

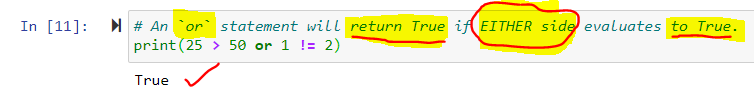


**Y** in “yellow” is greater than **C** in “cyan”. But, **B** in “brown” comes before **M** in “magenta” 🡪 1st statement is True, but 2nd statement is False

* *If only part of an expression is true, the result of the whole and statement is False.*

1. The **or** **operator** the expression will be True if:

* either of the expressions is true,
* and False only when both expressions are false.



25 > 50 is False

1 != 2 is True



1. The **not** **operator** inverts the value of the expression that follows it.

* If it’s True, it becomes False
* If it’s False, it becomes True

A white rectangular sign with yellow text

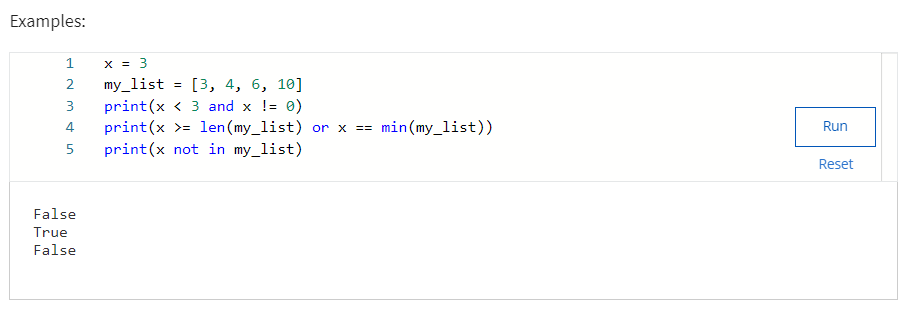
Description automatically generated

* **not** int equals to str

Because there is a **not** statement in front of 42 equals to the string “Answer”, the result is True.

A screenshot of a computer

Description automatically generated



# Use if, elif, else statements to make decisions

1. branching: the ability of a program to alter its execution sequence

**if**

a reserved keyword that sets up a condition in Python.

**A close up of a box

Description automatically generated**

**Goal:** to define a function so that it generates a username.

* Len() function can be paired with the less-than comparator to identify usernames that don’t meet the criteria.
* If , then followed by a condition we wanna check for, followed by a colon
* Then, we have the body of the IF block, which is indented further to the right.

**<!!!>** the body of the IF block will only execute when the condition evaluates to TRUE. Otherwise, it won’t execute.

**Means:** if you run an IF block and the argument conditions are not met, the idented code beneath it gets ignored.

**else**

A reversed keyword that executes when preceding conditions evaluate as False.

The **else** statement let us set a piece of code to run only when the condition of the IF statement is False.

A white box with text

Description automatically generated

A yellow marker on a white background

Description automatically generated

1. Modulo: an operator that returns the remainder when one number is divided by another.

Represented by the percentage sign ( **%** )

* Quotient
* Remainder

A screenshot of a computer

Description automatically generated

Even numbers are all multiples of two, which means the remainder of the integer division between an even number and two is always going to be zero. ????????????

A screenshot of a computer

Description automatically generatedA close up of a text

Description automatically generated

<!!!>

an IF statement branches the execution based on a specific condition being True, and the ELSE statement sets a piece of code to run only when the condition of the IF statement is False.

<!!!>

1. **elif**

A reversed keyword that executes subsequent conditions when the previous conditions are not true.

* Means: “if the previous conditions were not True, then try this condition.”

A screenshot of a computer code

Description automatically generatedA screenshot of a computer code

Description automatically generated



1. Uses of branching:

* Bin data based on its value
* Backup files
* Restrict login access

A screenshot of a computer code

Description automatically generated

[end of Week 2]

Week 3:

# Introduction to while loops

1. Loop: a block of code used to carry out iterations (lap lai.)

Iterations: the repeated execution of a set of statements, where one iteration is the single execution of a block of code.

Iterable: an object that’s looped, or iterated, over.

Data professionals use for loops and While loops to work with iterables.

while loop: a loop that instructs your computer to continuously execute your code based on the value of a condition.

* while loop work in a similar way to branching IF statements.

The difference is that, in while loops, the body of the block can be executed multiple times instead of just once. 🡪 avoid redundancy in codes

**<!!!>**

The condition used by the while loop needs to evaluate to True or False.

It doesn’t matter if this is done by using comparison operators or calling additional functions.

**<!!!>**

A screenshot of a computer program

Description automatically generated

* assign the value of 0 to the variable x
* start “While” loop, set a condition for this loop that x less than 5 (the prior line just initialized it, so the condition is currently True)
* end While loop with a colon.

While loop’s body:

* 1st line: print a message followed by the current value of x
* 2nd line: increment the value of x by adding 1 to its current value and assigning it back to x

We told the computer to print the results of each iteration.

Each iteration changed the value of x.

* Our While loop started at x = 0, then printed the message and increments the value of x to one.
* Because this is a loop, the computer doesn’t just continue executing with the next line in the script.

Instead, it loops back around to re-evaluate the condition for the While loop.--> this is the 2nd iteration.

* Because one here (x=1) is still less than 5, it executes the body of loop again.

The 3rd iteration increments x by 1, now the value of x is 2.

The computer will keep doing this until the condition isn’t True anymore.

* Add a “print” statement outside the body of While loop to let us know the final value of x. When the loop finishes, the next line of code is executed.
* Now, x has reached the value 5, loop statement ends.

The computer prints the last line of output as x = 5.

A screenshot of a computer game

Description automatically generated

1. break

A keyword that lets you escape a loop without triggering and ELSE statement that follows it in the loop.

Continue

To skip to the next iteration of the loop without executing the rest of the code in the current iteration.

A screenshot of a computer program

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer program

Description automatically generated

# Introduction to for loops

1. for loop

a piece of code that iterates over a sequence of values

A screenshot of a computer code

Description automatically generated

* for **x** in range 5, print **x**.
* for loop syntax’s structure is similar to a typical Python statement.
* First line indicates the distinguishing keyword, for
* Like functions and other expressions that starts a distinct code block, it ends with a colon.
* The body of the for loop is indented to the right.
* What’s different is we have the word **in**.
* Also, between for and **in**, we have the name of a variable, **x**.
* This variable, **x,** will take each of the values in the sequence the loop iterates thro.

**x** takes the values **0,1,2,3,4.**

1. range ()

a Python function that returns a sequence of numbers starting from zero, increments by 1 by default, and stops before the given number.

* It can be used in while or for loops

Range function:

1. A range of numbers will start with the value 0 by default
2. The list of numbers generated will be one less than the given value.

Use a for **loop** to read in a file, and iterate over the file line by line.

A screenshot of a computer program

Description automatically generated

* “**with** open” statement uses the file path to read in the file.
* For easier notation, assign it the value of “**f**”. Otherwise, we’d have to write the file path again.
* Next line, start for **loop** for each line. Inside, you indent, and on the next line, tell the computer to print each line.
* After the loop is complete, tell computer to print, “I’m done.”

# Loops with multiple range() parameters

# A white box with black text and green text Description automatically generated

# for loop that calc the factorial of nine.

# First, assign a variable called “product” with a number 1

# Second, have a range function that starts at 1 and stops at 10

# This means the code multiplies the variable by every whole number in the sequence beginning with 1, and ending with 9.

# So, for n in range 1 to 10, we multiply our “product” variable by 1 and reassign the result back to itself.

# Then, we multiply the “product” variable by 2, and reassign the result back to itself, then by 3,…. all the way up to 9.

# The factorial of 9 equals 362,880.

# <!!!> note that we started with 1 and not with 0, because if we multiplied by 0, the product would be 0.

# Range() function includes the following parameters:

# Start value

# Stop value

# Step value

# A screenshot of a computer Description automatically generated

# First, define a function that converts a temperature value from Fahrenheit to Celsius.

# Temperature in F is identified by x, 32 is subtracted from x, times 5, divided by 9.

# Next, have a for loop that will print a table of Temp. conversions every 10 degrees from 0 to 100 degrees Fahrenheit.

# Notice that for loop starts at 0, and goes up to 100 in steps of 10.

# Also, the range excludes the final value in a sequence. So, to include 100 in our sequence, we put the end value as 101.

# The body of for loop prints the value in F and value in C to create a conversion table.

# For every 10 degrees F, the code prints the corresponding value in C on each line.

<!!!>

Use for loops when there’s a sequence of elements that you want to iterate over.

Use while loops when you want to repeat an action until a Boolean condition changes.

A screenshot of a computer code

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated

# Work with strings

1. String: a sequence of characters and punctuation that contains textual info.

* This is an immutable data type, which means the values can never be altered or updated.

1. Concatenate: to link or join together

* Use addition operator “+”
* Blank spaces (in computer programming) counts as their own characters
* If you want a whitespace between your concatenated strings:
* One of the strings must contain a white space
* /or, you must add a third string between them that contains just a whitespace

A screenshot of a computer

Description automatically generated

* /or, adding 2 variables

A screenshot of a computer

Description automatically generated

* Can also multiple strings, using the multiplication operator

A white rectangular sign with black text

Description automatically generated

**<!!!>** we cannot divide or subtract strings.

A screenshot of a computer code

Description automatically generated

1. Quotation marks:

* 1st solution: include double quotation marks **“ “** in your string, use single quotation marks to begin and end your string, and vice versa.

A white rectangular box with black text

Description automatically generated

* 2nd solution: use backslash \ \

A screenshot of a computer

Description automatically generated

Escape character: a character that changes the typical behavior of the characters that follow it.

* In this case, the typical behavior of quotation marks is to begin or end the string, but if we proceed each with a backslash, they’ll behave as regular punctuation marks in the string.

A screenshot of a message

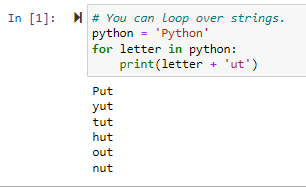
Description automatically generated

* \n is a special character combination that used to indicate a new line when printing a string.
* But, if you want to include \n characters in your strings when you print it, you must precede the combination with an initial backslash.

A white box with text

Description automatically generated

1. Iterate over STRINGS with LOOPS:



* Use a **for loop** to iterate over each letter of the word PYTHON, and print the letter, plus the letters **u-t.**

# String slicing

1. Indexing: a way to refer to the individual items within an iterable by their relative position.

* Allows us to select, filter, edit, and manipulate data
* Can be used on: Strings, Lists, Tuples, and most other iterable data types

1. Index ()

* A string method that outputs the index number of a character in a string.
* Python uses zero-based indexing.

This means the first element of a sequence is indexed as zero. With strings, indexing works by interpreting a string as a sequence of characters, where each character has a numbered slot.

A close-up of a line

Description automatically generated

* If you’re reading from LEFT to RIGHT, the 1st character is located at slot 0.

The 2nd character is located at slot 1.

And the 3rd character is at slot 2,…………. and so on.

* Indexing lets us slice strings to create smaller strings or substrings. For ex:

A table with many names

Description automatically generated with medium confidence

* Has both strings and integers: the currency symbol and salary amount.
* In this case, Python would automatically interpret the mix of data types as a string data type.

This is a problem bc we want values that represent money to behave like numbers so that we can perform mathematical operations on them. If it’s a strings, we can’t do that.

* Fix this problem: slicing helps us remove non-numeric characters, like dollar sign, from the string.

We can drop the character at the zero index of each value in the salary column🡪 end up with salary info without currency prefix.

A screenshot of a computer program

Description automatically generated

A screenshot of a text box

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A white background with black text

Description automatically generated

1. String slice: a portion of a string, also known as substring, that can contain more than one character.

A white rectangular box with black text

Description automatically generated

* Putting index numbers inside square brackets [1:4], and separating the numbers by a colon
* This defines a range of characters in the new slice. Go from index one up to index four.
* The closing index is not included in the range that’s returned to us (which is 4), so this would capture indices 1, 2, 3 🡪 which is R-A-N

Slice notation with just 1 of 2 indices.

Omitting the first number in the range implies that the range begins at zero.

A screenshot of a computer

Description automatically generated

* The string is “pineapple”, and we indicate our slice using “ :4 ”🡪 we’ll capture the first 4 letters: “pine”
* Similarly, if we slice using **“ 4: ”,** we’ll capture everything beginning with index 4 all the way to the end: “apple”

1. To check whether or not a substring is contained in a string, use the keyword in

A screenshot of a phone

Description automatically generated

* No “banana” in string pineapple, but “apple” is a substring of “pineapple”.

# String indexing and slicing

A screenshot of a computer code

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer

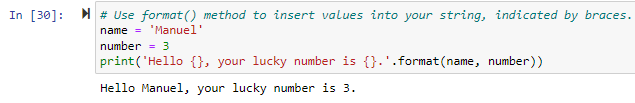
Description automatically generatedA screenshot of a computer program

Description automatically generated

# Format strings

**format ( )**

formats and inserts specific substrings into designated places within a larger string.



* 2 variables: name & number.
* Curly braces **{ }🡪** variables should be inserted into the string.

In format () **method**, it doesn’t matter that the name is a string and the number is an integer.

* format () **method** will insert strings of the values represented by these variables.

The order matters when inserting. “name” to the 1st set of braces, and “number” is the 2nd set.

A white box with black text

Description automatically generated

* we can name keywords and insert them into the braces.

For ex, “name” and “num”

* we assign our variables to those keywords in the method parameters.

When we run the cell, the value represented by the variables get inserted into the printed string according to their keywords.

The order doesn’t matter anymore. “name” will be inserted into “name” field, “number” will be inserted into “num” field of the string.

* This method makes rearranging strings fast and easy.

A white box with black text

Description automatically generated

* Use integer values in the braces to indicate the order in which to insert the arguments.
* For ex, we can enter the variables number and name in the argument field in a different order than they get inserted into the printed string.

A screenshot of a computer code

Description automatically generated



Inserting substrings into a larger string and format them.

* Printing the price of an item with and without tax.
* Use string formatting to set a limit on the number of decimal places in the output. In this case, our item costs $7 and 75 cents w/o tax, the tax rate is 7%. So, the price with tax would be $8.2925.
* Use a Special syntax: {: .2f} 🡪 limit the output to 2 places beyond the decimal point.

“.2” is 2 places beyond the decimal, and “f” stands is float.

If you put 0, only a whole number will print.

A screenshot of a computer

Description automatically generated

Conversion of Temp. from Fahrenheit to Celsius.

* Begin with a colon,

then use “greater than” operator to align the text the right so that the output is neatly formatted.

“greater than 3” will align the output 3 spaces to the right.

* For conversion to Celsius, use “greater than 6” will align the Celsius Temp. 6 spaces to the right.

A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer program

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer program

Description automatically generatedA screenshot of a computer program

Description automatically generatedA screenshot of a computer program

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer program

Description automatically generatedA screenshot of a computer

Description automatically generated

[end of Week 3]

Week 4:

1. Data structure: collections of data values or objects that contain different data types.

* Used to store, access, organize, categorize data with speed and efficiency.
* Lists, Tuples, Dictionaries, Sets, Arrays
* NumPy: high-performance computational power.

Used to rapidly process large quantities of data

* Pandas: Python Data Analysis Library.

Makes analyzing data in the form of a table with rows and columns easier and more efficient, because it has tools specifically designed for the job.

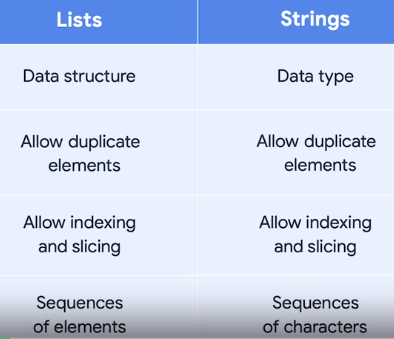
# Introduction to lists

1. Data structure: collections of data values or objects that contain different data types.

* Used to store, access, organize, categorize data with speed and efficiency.
* Contain data type elements, such as a float or a string.
* Enables more efficient storage, access, and modifications.

Allow you to organize and categorize your data, relate collections of data to each other, and perform operations accordingly.

1. List: a data structure that helps store and manipulate an ordered collection of items. For ex: a list of email addresses associated with a user account.

* Quite similar to a String.
* 
* Both Lists & Strings allow duplicate elements, as well as, indexing and slicing.
* Both are sequences.

1. Sequence: a positionally ordered collection of items.

* However, S-T-R are sequences of characters, while LISTS store sequences of elements of any data type.

1. Mutability: the ability to change the internal state of a data structure.

* Lists and their contents are mutable, so their elements can be modified, added, or removed.

1. Immutability: a data structure or element’s values can never be altered or updated.

* Strings are immutable.

A blue rectangular object with numbers

Description automatically generated

* List is like as long box, with a space inside, divided into different slots.
* Each slot contains a value, and each value can store data.
* This could be another data structure, such as another *list*, or an *integer*, *string*, *floa*t, or *output from another function*.
* When working w lists, use *an index* to access each of the elements.

An index provides the numbered position of each element in an ordered sequence.

In this case, our sequence is a list.

A white rectangular object with text

Description automatically generated with medium confidence

* First, assign the following list of words to a variable X: “Now”, “we”, “are”, “cooking”, ‘with”, “7”, “ingredients”.

In Python, square brackets [ ] indicate where the list starts and ends.

Commas to separate each element contained in it.

* To print an element of a list, use its index number.

So, to print the word “cooking”, print the 3rd element of the list variable X.

* This is just like focusing on a specific character or substring in a string. The 1st element in a list, as with strings, is 0.
* So, if we print the element with slot number 3, we get the item, or word “cooking”, from our list of 7 words.

**<!!!>** Indexing always starts at 0.

A screenshot of a computer code

Description automatically generated

* No “7”,

“6” is the last word in the list.

A screenshot of a computer program

Description automatically generated

**<!!!>** use *index* to create *a slice of the list.*

For ex:

* Use ranges of 2 numbers, separated by a colon
* Get 2nd & 3rd words of our list: “we” and “are”
* Use a colon to get all words until the index slot 2: “now” and “we” (which have index slots 0 and 1)
* And to leave one of the range indexes empty, use 2 colon 🡪 this will give us the other part of the list.

A screenshot of a computer

Description automatically generated

* To check if a list of words contains a certain element, “This”, use to keyword “in” to generate Boolean statement.

This verifies whether the word exists.

# Modify the contents of a list

A green squares with white text

Description automatically generated

A screenshot of a cell phone

Description automatically generated

Modifying it means we keep the box, but we add, remove or change what’s inside.

append( )

* method that adds an element to the end of the list.
* This requires 1 argument because this function adds the incoming element to the end of the list as a single new entry.
* You can even start with an empty list and all new elements will be added at the end.

A screenshot of a computer

Description automatically generated

* For ex: in the list of fruits, forgot to add *Kiwi* to the list. So, we can use the append ( ) **method** to add it. This uses one parameter; in this case, the string *Kiwi*.

insert ( )

* Requires 2 arguments:
* The index number of the elements to be modified,
* The contents being put in that slot, such as a *string or integer*.
* Insert (): function that takes an index as the 1st parameter, and an element as the 2nd parameter, then inserts the element into a list at the given index.

A screenshot of a computer

Description automatically generated

* “orange” is inserted at the 2nd spot at index one in our fruit list.

remove ( ): A method that removes an element from a list

* Similar to *aspend(),* *remove()* only requires 1 parameter.

A white background with black text

Description automatically generated

* Now, our fruit list has no “banana”

A screenshot of a computer code

Description automatically generated

* Get a ValueError if we try to remove an element that is not in the list.

pop (): A method that extracts an element from a list by removing it at a given index.

A screenshot of a computer

Description automatically generated

* To remove “orange”, pop the 3rd element in the list with index number 2.

A screenshot of a computer

Description automatically generated

* Suppose that, after removing “orange”, you decide to also remove “pineapple” and replace it with “mango”🡪 simply reassign its value.
* Reference the pineapple item’s index number, one, and replace it with mango.

A screenshot of a computer program

Description automatically generated

* Strings are immutable

Lists are mutable.

* Whenever we modify a string, we always have to reassign the change back to the variable that contained the string. This is overwriting the existing variable with a brand new one.
* We can’t overwrite the character at index 0. We get an error. But, we can do this with a list-🡪 that’s why lists are mutable.

A screenshot of a computer

Description automatically generatedA screenshot of a computer program

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer program

Description automatically generatedA screenshot of a computer program

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated

# Introduction to tuples

1. Tuple: an immutable sequence that can contain elements of any data type.

* Tuples are kind of like Lists, but they’re more secure because they cannot be changed easily.
* They’re helpful bc they keep data that needs to be processed together in the same structure.

Tuples are expressed with parentheses ( ) or the tuple ( ) function.

* Tuples are also used to return values and functions.

When a function returns more than 1 value, it’s actually returning a tuple.

A screenshot of a computer program

Description automatically generated

A screenshot of a text

Description automatically generated

1. tuple ( ): function that transforms input into tuples.

A screenshot of a computer code

Description automatically generatedA white background with black text

Description automatically generated

A screenshot of a computer code

Description automatically generatedA text on a white background

Description automatically generated

A screenshot of a computer

Description automatically generatedA screenshot of a text box

Description automatically generated

1. Advantages of working w Tuples:

* Let you store data of different types inside other data structures
* It saves memory and can really optimize your programs
* during collaboration, tuples make it clear to your teammates that your sequences of values are not intended to be modified. 🡪 save time and effort.

A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer program

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer program

Description automatically generatedA screenshot of a computer program

Description automatically generatedA screenshot of a computer

Description automatically generated

A screenshot of a computer program

Description automatically generated

A screenshot of a computer

Description automatically generated

A white text on a white background

Description automatically generated

A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer program

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated

# More with loops, lists, and tuples

A screenshot of a computer program

Description automatically generatedA text on a white background

Description automatically generated

A diagram of a chart

Description automatically generated with medium confidenceA close-up of a white background

Description automatically generated

A diagram of a game

Description automatically generated with medium confidenceA white background with black text

Description automatically generated

A screenshot of a computer program

Description automatically generatedA screenshot of a phone

Description automatically generated

1. Nested loop: is when you have 1 loop inside another loop.

* These loops create all the different domino tiles in a set of dominoes, which is a tile-based game played with numbered gaming pieces.

A screenshot of a computer program

Description automatically generatedA text on a page

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generatedA white text with black text

Description automatically generated

A screenshot of a computer code

Description automatically generated

A screenshot of a computer

Description automatically generated

1. list comprehension: formulaic creation of a new list based on the values in an existing list.

A computer code with black text

Description automatically generatedA screenshot of a computer

Description automatically generated

A screenshot of a text

Description automatically generated

1. A screenshot of a computer

   Description automatically generatedA screenshot of a computer program

   Description automatically generatedA screenshot of a computer

   Description automatically generatedA screenshot of a computer

   Description automatically generatedA screenshot of a computer

   Description automatically generated

# Introduction to dictionaries

1. Dictionary: a data structure that consists of a collection of key-value pairs.

* They are instantiated with braces { } or the dict ( ) function.
* Use by both advanced and entry level professionals to analyze large data sets with fast processing power.
* Helps them gather and transform user info.
* Provides a straightforward way to store data, make it easier for users to find specific info.

A screenshot of a computer code

Description automatically generatedA white text with black text

Description automatically generated

A screenshot of a computer code

Description automatically generatedA close up of a text

Description automatically generated

Dictionaries are instantiated mainly in 2 ways:

* 1st way is with braces { }

Each key is separated from its value by a colon, and each key-value pair is separated from the next by a comma.

* 2nd way is with dict ( ) function.

A screenshot of a text

Description automatically generated

A screenshot of a computer code

Description automatically generatedA screenshot of a white text

Description automatically generated

1. Immutable keys: must be immutable

* *Integers*
* *Floats*
* *Tuples*
* *Strings*

Mutable data types cannot be used as keys:

* *Lists*
* *Sets*
* *Other dictionaries*

A screenshot of a computer code

Description automatically generatedA screenshot of a computer

Description automatically generated

A white rectangular object with black text

Description automatically generatedA screenshot of a computer

Description automatically generated

# Dictionary methods

A screenshot of a computer code

Description automatically generatedA screenshot of a dictionary

Description automatically generated

A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generatedA white rectangular box with black text

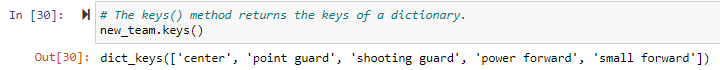
Description automatically generated

A screenshot of a computer

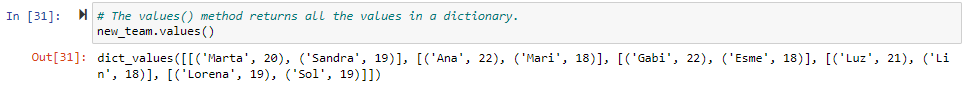
Description automatically generatedA text on a white background

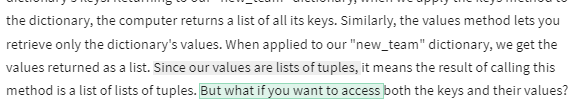
Description automatically generated

1. keys (): a dictionary method to retrieve only the dictionary’s keys.



1. values (): a dictionary method to retrieve only the dictionary’s values.





1. items (): a dictionary method to retrieve both the dictionary’s keys and values.

A screenshot of a computer code

Description automatically generated

A screenshot of a dictionary

Description automatically generatedA screenshot of a computer code

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer program

Description automatically generatedA screenshot of a computer program

Description automatically generatedA screenshot of a computer program

Description automatically generatedA screenshot of a computer program

Description automatically generated

# Introduction to sets

1. Set: a data structure in Python that contains only unordered, non-interchangeable elements.

* Instantiated with set () function or non-empty braces.
* Each set element is unique and immutable. However, the set () itself is mutable.
* Sets are valuable when storing mixed data in a single row, or a record, in a data table.
* Also frequently used when storing a lot of elements, and you want to be certain that each one is only present once.
* Because sets are mutable, they cannot be used as keys in a dictionary
* 2 ways to create Set:
* 1st way: set () function- takes an iterable as an argument and returns a new set object.
* 2nd way: put inside braces. Cannot be empty braces because your computer will interpret as a dictionary.

To define an empty set, you have to use set ()

A screenshot of a computer code

Description automatically generated

A white text on a white background

Description automatically generated

A computer screen shot of a computer

Description automatically generated

A close up of text

Description automatically generated

A screenshot of a computer code

Description automatically generated

A screenshot of a computer

Description automatically generated

A white rectangular object with black text

Description automatically generatedA close up of text

Description automatically generated

**<!!!>** to define an empty set, use **set ( )**

* Can only use curly braces { } when the set is not empty, and you are assigning the set to a variable.
* Because the elements inside a set are immutable, a set cannot be indexed or sliced.

1. intersection () : a function that finds the elements that 2 sets have in common.

A screen shot of a computer code

Description automatically generatedA black text on a white background

Description automatically generated

1. union (): a function that finds all the elements from both sets.

A screenshot of a computer

Description automatically generated  
A white text with black text

Description automatically generated

1. difference (): a function that finds the elements present in 1 set, but not the other.

A white screen with black text

Description automatically generated

A text on a white background

Description automatically generated

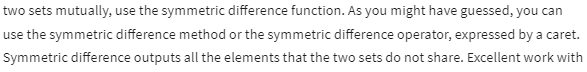
A white rectangular object with black text

Description automatically generated

1. symmetric\_difference (): a function that finds elements from both sets that are mutually not present in the other.

A screenshot of a computer program

Description automatically generated



A screenshot of a reference guide

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer program

Description automatically generatedA screenshot of a computer program

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer program

Description automatically generated

# The power of packages

1. library (or package): broadly refers to a reusable collection of code

* it contains related modules and documentation.
* Python libraries:
* *matplotlib, seaborn,*
* *NumPy, pandas.*

1. Matplotlib: a library for creating static, animated, and interactive viz in Python.
2. Seaborn: a viz library based on matplotlib that provides a simpler interface for working with common plots and graphs.
3. NumPy (Numerical Python): an essential library that contains multidimensional array and matrix data structures and functions to manipulate them.

* Is used for scientific computation.

1. Pandas: a powerful library built on top of NumPy that’s used to manipulate and analyze tabular data.

* Others: scikit-learn, statsmodels, …..

1. Module: a simple Python file containing a collection of functions and global variables.

* Can be accessed from within a package or a library.
* Used to organize functions, classes, and other data in a structured way.
* Modules are set up thro separate files that contain necessary classes and functions.

1. Global variables: variables that can be accessed from anywhere in a program or script.
2. Commonly used Python modules:

* Math: provides access to math functions.
* Random: is used to generate random numbers.

Useful when selecting random numbers from a list; shuffling elements randomly; or working with random sampling.

# Introduction to NumPy

1. Vectorization: enables operations to be performed on multiple components of a data object at the same time.
2. Vectorized approach simpler, easier to read, and faster to execute because While loops iterate over one element at a time, vector operations compute simultaneously in a single statement.
3. Vectors also take up less memory space,
4. When used NumPy, first have to import it. When we import NumPy, we import it as NP (aliasing).
5. Import statement: used the import keyword to load an external library, package, module, or function into your computing environment.
6. No need to import again unless you restart your notebook.
7. Aliasing: lets you assign an alternate name – or alias - by which you can refer to something.
8. np is a standard alias.
9. If didn’t give NumPy an alias of np, type out “NumPy” in order to access its array function.

# Basic array operations

1. N-dimensional array (ndarray): the core data object of NumPy.
2. ndarray is a vector.
3. ndarray are mutable, so you can change the values they contain.
4. But, cannot change the size of array w/o reassigning it.
5. All of its elements must be the same data type.
6. dtype: a NumPy attribute used to check the data type of the contents of an array.
7. ndarray can be multidimensional.
8. A one-dimensional array is neither a row nor a column.
9. shape: a NumPy attribute used to check the shape of an array.
10. ndim: a NumPy attribute used to check the number of dimensions of an array.
11. reshape ( ): NumPy method used to change the shape of an array.

A screenshot of a computer program

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer program

Description automatically generatedA screenshot of a computer program

Description automatically generatedA screenshot of a computer program

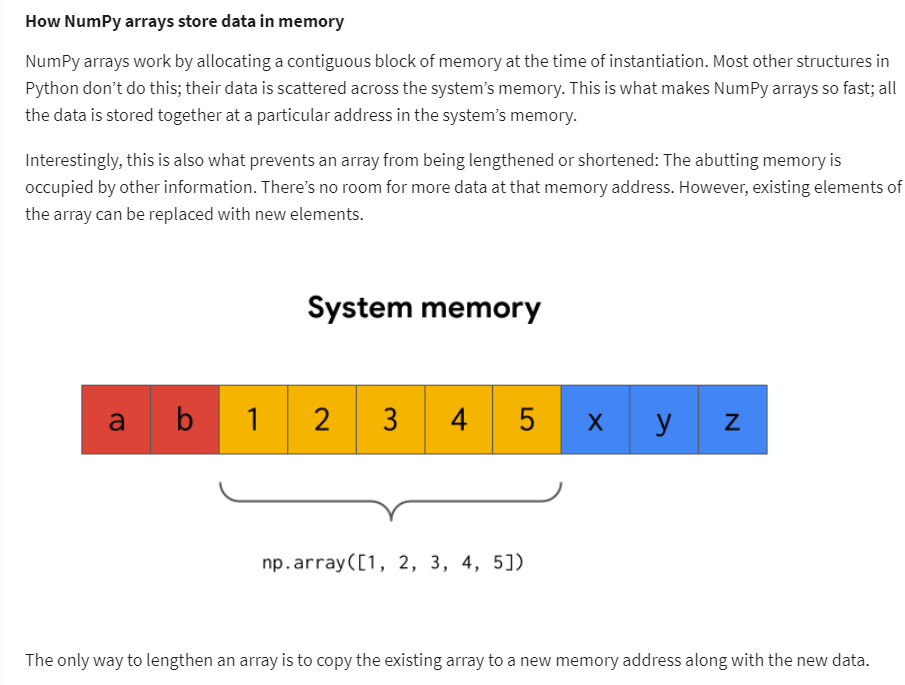
Description automatically generatedA screenshot of a computer program

Description automatically generated

A screenshot of a computer

Description automatically generatedA screenshot of a computer program

Description automatically generatedA screenshot of a computer program

Description automatically generated

# Introduction to pandas

1. tabular data: data that is in the form of a table, with rows and columns.
2. A spreadsheet is a common example of tabular data.
3. Core pandas object classes:

* DataFrame
* Series

1. DataFrame: a two-dimensional, labeled data structure with rows and columns.
2. A spreadsheet or a SQL table
3. Dataframes use to structure, manipulate, and analyze data in Pandas

# pandas basics

# CSV file: “comma-separated values”. A plaintext file that uses commas to separate distinct values from one another.

# Series: a one-dimensional, labeled array.

# NaN: how null values are represented in pandas, which stands for “not a number”

# iloc [ ]: a way to indicate in pandas that you want to select by integer-location-based position.

# loc [ ]: used to select pandas rows and columns by name.

# A screenshot of a computer Description automatically generatedA screenshot of a computer program Description automatically generatedA screenshot of a computer Description automatically generatedA screenshot of a computer Description automatically generatedA screenshot of a computer code Description automatically generatedA screenshot of a computer program Description automatically generatedA screenshot of a computer program Description automatically generatedA screenshot of a computer program Description automatically generatedA screenshot of a computer program Description automatically generatedA screenshot of a computer program Description automatically generatedA screenshot of a computer Description automatically generatedA screenshot of a computer Description automatically generatedA screenshot of a computer program Description automatically generatedA screenshot of a computer program Description automatically generatedA screenshot of a computer program Description automatically generatedA screenshot of a computer Description automatically generated

# Boolean masking

1. Boolean masking: A filtering technique that overlays a Boolean grid onto a dataframe in order to select only the values in the dataframe that align with the True values of the grid.

A screenshot of a computer

Description automatically generatedA screenshot of a test

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer program

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer program

Description automatically generatedA screenshot of a computer

Description automatically generated

# Grouping and aggregation

1)

- groupby ( ): a pandas DataFrame method that groups rows of the dataframe together based on their values at one or more columns, which allows further analysis of the groups.

1. agg ( ): short for “aggregate”. A pandas groupby method that allows you to apply multiple calculations to groups of data.

A screenshot of a computer program

Description automatically generatedA screenshot of a computer code

Description automatically generatedA screenshot of a computer program

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer program

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer code

Description automatically generatedA screenshot of a computer program

Description automatically generatedA screenshot of a computer

Description automatically generated

# Merging and joining data

1. Pandas functions:

* concat ()
* merge ()

1. concat (): (means link or join together) a pandas function that combines data either by adding it horizontally as new columns for existing rows, or vertically as new rows for existing columns.
2. Capable of handling many data-specific complexities that arise, which allows for a high degree of user control.

A white background with blue lines

Description automatically generated

1. 2 axes of a dataframe are zero, which runs vertically over rows;
2. 1 which runs horizontally across columns.

3)

A screenshot of a computer

Description automatically generated

1. This data has 4 planets, radii, and their number of moons.

But, it’s missing the data for Jupiter, Saturn, Uranus, and Neptune.

1. We want to add this data, which exists as a separate df.

A screenshot of a computer

Description automatically generated

1. The 2nd dataset with info about Jupiter, Saturn, Uranus, and Neptune.
2. This dataset has the same format as the data in df1 dataframe.

It has the same columns for planer, radius, and moons.

A screenshot of a computer

Description automatically generated

1. To combine 2 df, we’ll want to add df2 as new rows below df1.
2. To concatenate the 1st dataset with info about Mercury, Venus, Earth, and Mars with the 2nd, which has info about Jupiter, Saturn, Uranus, and Neptune, we call pf concat and insert a list of df we want to concatenate.
3. Then, we need to include an axis keyword argument.

This instructs the function to combine the data either side-by-side or one on top of the other.

1. We want our resulting dataframe to have 8 rows and 3 columns, means we want to combine data vertically.

We want to add new data by extending axis zero, the vertical axis.

1. Notice each row retains its index number from its original dataframe.
2. If you want the numbering to restart, just reset the index.
3. We include “drop equals true” argument because otherwise a new index column will be added to the dataframe, which we don’t want in this case.

A screenshot of a computer

Description automatically generated

1. merge (): a pandas function that joins 2 dataframes together; it only combines data by extending along axis (1) horizontally.

A screenshot of a computer

Description automatically generated

1. now, we want to add the data for the planet type, whether it has rings, its average temperature, whether it has a magnetic field, and whether it has life on it.
2. This data exists as a separate dataframe, but it’s missing Mercury and Venus, and new planets from other star systems: Janssen and Tadmor.
3. For 2 datasets to connect, they need to share a common point of reference. Both datasets must have some aspect of them that is the same in each one. They are known as **keys**.
4. Keys: the shared points of reference between different dataframes - what to match on.
5. In our case, keys are the planets.

Each dataframe contains planets for us to match on.

A screenshot of a graph

Description automatically generated

1. “inner” join: Only the keys that are in both dataframes get included in the merge.

A screenshot of a graph

Description automatically generated

1. “outer” join: join the data so all the keys form both dataframes get included in the merge.

A colorful chart with different colored squares

Description automatically generated with medium confidence

1. “left” join: join data so all the keys in the left dataframe are included, even if they aren’t in the right dataframe.

A screenshot of a graph

Description automatically generated

1. “right” join: join data so all the keys in the right dataframe are included, even if they aren’t in the left dataframe.

A screenshot of a computer

Description automatically generated

1. First, we call the function and enter df3 and df4 as the left and right positional arguments, respectively.
2. Then, we include keyword argument “on”, which lets us specify what our keys to match on should be.

In this case, we want to use “planet” column.

1. Now, we have the “how” keyword argument. This is where we enter the kind of join we want.

This time, try “inner” join.

1. This merged the data and only kept the planets that appeared in both dataframes.
2. This means we’re missing data for Mercury and Venus from the left df as well as for Janssen and Tadmor from the right df.

A screenshot of a computer

Description automatically generated

1. This time, try “outer” join.

Our function call will remain the same except for the “how” keyword argument, which we’ll set to “outer”.

1. As expected, this results in a dataframe that contains all the keys from both initial dataframes.

Notice that, because Janssen and Tadmore aren’t in the left dataframe, they don’t have info for radius and moons, so these columns get filled in with NaN.

1. Similarly, because Mercury and Venus aren’t in the right dataframe, they are too missing some info in the final table, which is represented by NaN.

A screenshot of a computer

Description automatically generated

1. Again, the function gets the same syntax except for the “how” argument, which is set to “left”.
2. This results in a dataframe that retains all the keys from the left dataframe and only the keys from the right dataframe that exist in the left dataframe too.

So, Janssen and Tadmor are excluded.

A screenshot of a computer

Description automatically generated

1. As expected, the result is a dataframe that has all the keys from the right dataframe, but none of the keys from the left that weren’t also in the right.

So, Mercury and Venus are excluded.

WEEK 5

1. Experiential learning: the idea of understanding through doing.

Data tidying: structuring datasets to facilitate analysis.

Tidy dataset:

* Easy to manipulate, model, and visualize
* Each variable is a column
* Each observation is a row
* Each type of observation unit is a table