# CS50P W2 - Loops

### While Loops

While the **condition is true** the loop keeps iterating. The only way to break the loop is for the condition to return false. If the condition stays true, the loop will be infinite:

* **i** = 3
* **while i !=** 0:

print(“moew”)

\*Use **control + c** to break out of a **infinite loop**

1.

* **i** = 3
* **while i !=** 0:

print(“moew”)

**i = i - 1**

2.

* **i** = **1**
* **while i <=** 3:

print(“moew”)

**i = i + 1**

3.

* **i** = 0
* **while i <** 3:

print(“moew”)

**i += 1**  # **i += 1** == **i = i + 1**

### water.py

1.

* **from** soil **import** sample
* **def** main():

moisture = sample()

print(**f**"Moisture is **{**moisture**}**%")

**while** moisture **>** 20:

moisture = sample()

print(**f**"Moisture is **{**moisture**}**%")

print("Time to water!")

main()

\*While moisture is more than 20% (condition is True), keep sampling the soil. When the condition becomes False (moisture less or equal to 20%), **break the loop** and print “Time to water!”.

2.

* **def** main():

moisture = sample()

days = **0**

print(**f**"Moisture is **{**moisture**}**%")

**while** moisture **>** 20:

moisture = sample()

days **+=** 1

print(**f**"Day **{**days**}**: Moisture is **{**moisture**}**%")

print("Time to water!")

\*While moisture is more than 20% (condition is True), keep sampling the soil and **increase the day count** by 1.

### For Loops

A **for** loop iterates through a **list** of items.

\*Python automatically **initialize** (to 0) and **update i**

1.

* **for** **i in [0, 1, 2]**:

print(“meow”)

2.

* **for** **i in range(3)**:

print(“meow”)

\*The **range()** built-infunction takes one argument (the **number of values** you want back), **starting at 0** up to (but not through) the specified value.

3.

* **for** **\_ in range(3)**:

print(“meow”)

\*The “pythonic” way is to represent “**i**” with an underscore “**\_**”

4.

* **print(“**meow**” \* 3)**

\*This code will output `meowmeowmeow`

5.

* **print(“**meow**\n” \* 3** **end=””)**

\*Adding a “**\n**” after the string and removing the automatic one after the print will output :

meow

meow

meow

#### User Input

1.

* **while** **True**:

n = int(input("What's n? "))

if n **<** 0:

continue

else:

**break**

\*If the user enters a negative (< 0) number the loop keeps asking for input until **entering a positive number breaks the loop**.

2.

* **while** **True**:

n = int(input("What's n? "))

if n **>** 0:

**break**

* **for \_** **in** range(n):

print(“meow”)

3.

* **def** **main()**:

meow(3)

* **def** **meow(**n**)**:

**for** \_ in range(n):

print("meow")

main()

4.

* **def** main():

number = get\_number()

meow(number)

* **def** get\_number():

**while** True:

n = int(input("What's n? "))

**if** n **>** 0:

break

**return** n

* **def** meow(n):

**for** **\_** **in** range(n):

print("meow")

main()

### Lists

* student**s** = **[**"Hermione", "Harry", "Ron"**]**

1.

* print(**students[**0**]**)

print(**students[**1**]**)

print(**students[**2**]**)

2.

* **for** **student in** student**s**:

print(student)

3.

* **for** i **in** **range(len(**students**))**:

print(**students[i]**)

4.

* **for** i **in** **range(len(**students**))**:

print(**i** + 1, **students[i]**)

\*1 Hermione

2 Harry

3 Ron

#### **Append elements to a list**

1.

* **results** = ["Mario", "Luigi"]

results.append("Princess")

results.append("Yoshi")

results.append("Koopa Troopa")

results.append("Toad")

print(results)

2. Append multiple elements at a time :

* results.append(**[**“Bowser”, “Donkey Kong Jr.”**]**)

\*This will result in a separate sublist within the original list. The extend() method is better suited to append multiple elements to the list

results.remove(**[**“Bowser”, “Donkey Kong Jr.”**]**)

results.**extend([**“Bowser”, “Donkey Kong Jr.”**])**

#### **Remove elements from a list**

* **results** = ["Mario", "Luigi", "Princess", "Yoshi", "Koopa Troopa",

"Toad", "Bowser", "Donkey Kong Jr."

]

results.remove(“Bowser”)

#### **Insert elements into a list** in a specific **index** insert(index, element)

* **results** = ["Mario", "Luigi", "Princess", "Yoshi", "Koopa Troopa",

"Toad", "Donkey Kong Jr."

]

results.**insert(**0,“Bowser”**)**

#### **Reverse order of the elements of a list**

* **results** = ["Mario", "Luigi", "Princess", "Yoshi", "Koopa Troopa",

"Toad", “Bowser”, "Donkey Kong Jr."

]

results.**reverse()**

Output : ["Donkey Kong Jr.", “Bowser”,"Toad", "Koopa Troopa",

"Yoshi", "Princess", "Luigi", "Mario"

]

### Dictionaries

Data structure that allows us to associate **keys** with **values**.

* **students** = **{**

"Hermione"**:** "Gryffindor",

"Harry"**:** "Gryffindor",

"Ron"**:** "Gryffindor",

"Draco"**:** "Slytherin"**}**

print(**students[**"Hermione"**]**)

\*We use the **key** as an index to retrieve the **value** associated with it.

1.

* **for** student **in** students:

print(**student**)

\*Printing the iterator “**student**” will print the **keys** in the list :

Hermione

Harry

Ron

Draco

2.

* **for** student **in** students:

print(student, **students[**student**]**)

\*Prints the keys “student”, and the values “**students[**student**]**”

Hermione Gryffindor,

Harry Gryffindor

Ron Gryffindor

Draco Slytherin

3.

* **for** student **in** students:

print(student**,** **students[**student**],** **sep=”, ”**)

\*Prints the keys “**student**”, and the values “**students[student]**”

Hermione**,** Gryffindor

Harry**,** Gryffindor

Ron**,** Gryffindor

Draco**,** Slytherin

#### report.py

**def** main():

spacecraft = **{**"name"**:** "Voyager 1", "distance"**:** 163**}**

print(create\_report(spacecraft))

**def** create\_report(spacecraft):

return **f**"""

========= REPORT =========

Name: {**spacecraft[**"name"**]**}

Distance: {**spacecraft[**"distance"**]**} AU

==========================

"""

\*Notice the capabilities of the **f-strings**

Adding Keys :

1.

* **spacecraft** = **{**"name"**:** "Voyager 1", "distance"**:** 163**}**

**spacecraft[**“distance”**]** = 0.01

\*We can add a key-value pair to a dictionary by *indexing into the key* and assigning a new value.

2.

* **spacecraft** = **{**"name"**:** "james Webb Space Telescope"**}**

spacecraft.**update({**“distance”**:** 0.01, “orbit”**:** “Sun”**})**

Error Checking :

* **def** create\_report(spacecraft):

return **f**"""

========= REPORT =========

Name: {spacecraft.**get(**“name”, “Unknown”**)**}

Distance: {spacecraft.**get(**"distance", “Unknown”**)**} AU

Orbit: {spacecraft.**get(**"orbit", “Unknown”**)**}

==========================

"""

\***get(**key**,** default\_value**)** :

spacecraft.get(“distance”, “Unknown”) will try to access the value associated with the “distance” key, if no value is found, it will default to returning “Unknown”

#### distances.py

**distances** = **{**

"Voyager 1"**:** 163**,**

"Voyager 2"**:** 136**,**

"Pioneer 10"**:** 80**,**

"New Horizons"**:** 58**,**

"Pioneer 11"**:** 44

**}**

1. Accessing keys

* **def** main():

**for** name **in** distances**.keys()**:

print(**f**"{name} is {distances[name]} AU from Earth”)

\*If the keys in the dictionary are different, we can use the **keys()** method to access each of those keys.

\*Notice the use of an **f-string** and the distances[name]

2. Accessing values

* **def** main():

**for** name **in** distances**.values()**:

print(**f**"{name} is {distances[name]} AU from Earth”)

3. Converting values (AU to meters)

* **def** main():

**for** distance **in** distances**.values()**:

print(**f**"{distance} AU is {convert(distance)} AU from Earth”)

* **def** convert(au):

returnau **\*** 149597870700

### Lists of Dictionaries

**students** = **[**

**{**"name"**:** "Hermione", "house"**:** "Gryffindor", "patronus"**:** "Otter"**},**

**{**"name"**:** "Harry", "house"**:** "Gryffindor", "patronus"**:** "Stag"**},**

**{**"name"**:** "Ron", "house"**:** "Gryffindor", "patronus"**:** "Jack Russell terrier"**},**

**{**"name"**:** "Draco", "house"**:** "Slytherin", "patronus"**:** None**}**

**]**

\*The keyword ”None” means there is no value associated with the “patronus” key in the last dictionary.

1.

* **for** **student in** students:

print(**student[**“name”**]**)

\*The iterator **student** now represents the **dictionaries** in the list

\*Prints values associated with “name” key in each dictionary

2.

* **for** **student in** students:

print(**student[**“name”**], student[**“house”**], student[**“patronus”**], sep=”, ”**)

\*Prints values associated with the “name”, “house”, and “patronus” **keys** (separated by a comma) in each dictionary :

Hermione**,** Gryffindor**,** Otter

Harry**,** Gryffindor**,** Stag

Ron**,** Gryffindor**,** Jack Russell terrier

Draco**,** Slytherin**,** None

### Mario

1.

* **def** **print\_column(**height**)**:

**for** \_ **in** range(height):

print("**#**")

2.

* **def** **print\_column(**height**)**:

print(“**#**\n” **\*** height, end=””)

3.

* **def** **print\_row(**width**)**:

print(“**?**” **\*** width)

#### Nested loops

1.

* **def** **print\_square(**size**)**:

**for** i **in** range(size):

**for** j **in** range(size):

print("**#**")

**print()**

* **def** **print\_square(**size**)**:

# For each row in square

**for** i **in** range(size):

# For each brick in row

**for** j **in** range(size):

# Print brick

print("**#**")

# Add new line at the end of the row

**print()**

2.

* **def** **print\_square(**size**)**:

**for** i **in** range(size):

print("**#**" **\*** size)

3.

* **def** **print\_square(**size**)**:

**for** i **in** range(size):

print\_row(size)

**def** print\_row(width):

print(“#” **\*** width)

### Letters.py

1.

**def** main():

print(write\_letter("Mario", "Princess Peach"))

print(write\_letter("Luigi", "Princess Peach"))

print(write\_letter("Daisy", "Princess Peach"))

print(write\_letter("Yoshi", "Princess Peach"))

**def** write\_letter(receiver, sender):

return **f**"""

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

Dear **{**receiver**}**,

You are cordially invited to a ball at

Peach's Castle this evening, 7:00 PM.

Sincerely,

**{**sender**}**

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

"""

main()

2. Create a list and loop through it (using **i** as an iterator)

* **def** main():

**names** = ["Mario", "Luigi", "Daisy", "Yoshi"]

for **i** in range(len(names)):

print(write\_letter(names[**i**], "Princess Peach"))

3. Pythonic simplified alternative syntax (using **name** as an iterator)

* **def** main():

**names** = ["Mario", "Luigi", "Daisy", "Yoshi"]

forname in **names**:

print(write\_letter(names, "Princess Peach"))