

# PYTHON CLASS 15

---

LOOPS, LISTS AND DICTIONARIES



# LOOP EXERCISES

---

- Create a loop that counts from 0 to 100
- Create a loop that multiplies the numbers from 1-20.
- Create a loop that adds random numbers to a list.
- Use a loop to find the largest and smallest numbers, and the average.

# WAYS TO CONTROL ACTION INSIDE A LOOP

---

- You can use if-statements and **break** or **continue** to control action

```
li = [2,4,6,8,10,11,14,16,18]
```

```
for x in li:  
    if x % 2 != 0:  
        break  
    print(x)
```

we want to stop  
the loop  
completely if we  
have an odd  
number

```
li = [1,4,9,16,25,0,36,49,64]
```

```
for x in li:  
    if x == 0:  
        continue  
    else:  
        print((li.index(x) + 1) / x)
```

we just want  
to skip the  
number to  
avoid dividing  
by 0

# LISTS AND LOOP PRACTICE

---

- Practice using continue and break together with loops and lists.
  - Make a list of strings. We want to find the first string that is longer than 5 letters. If the length of the string is longer than 5, stop the loop.
  - Make a list of integers. Divide them by 8. If the remainder is 0, go to the next number. If it is not 0, add the number to a new list.

# MORE LIST PRACTICE

---

- Make a list of nucleic acids (DNA). Make a function to change it into a string. Check if the string contains this “ATTACAG.”
- Make a list of random Japanese sounds. Make a function to change it into a string. Check if the string contains a word (you choose the word).



An open dictionary is shown from a top-down perspective, lying flat. The pages are filled with dense, small text, typical of a reference work. A vertical white line runs down the center of the image, separating the left and right pages. The background is dark, making the light-colored pages stand out.

THINK ABOUT A  
LANGUAGE DICTIONARY.  
HOW DO YOU USE IT?

# REAL WORLD DICTIONARIES

# LISTS IN PYTHON, REVIEW

---

- Remember how lists work.

```
li = ["L","I","S","T"]
```

```
print(li[0])
```

```
li.append("q")
```

```
for x in li:
```

```
    if x.isupper():
```

```
        print(x + " is a capital letter")
```

# DICTIONARIES IN PYTHON

---

- Dictionaries use curly brackets, `{}`.
- The data in a dictionary is in pairs: the first item is the **key**. The second item is the **value**.
- In the dictionary below, both the key and the value are strings, but you can use any combination – date:date, date:string, int:date,, int:string, string:bool, etc.
- You can even have dictionaries or lists inside your dictionary!
- Just like lists, we use `di[x]` to get the information in location x.

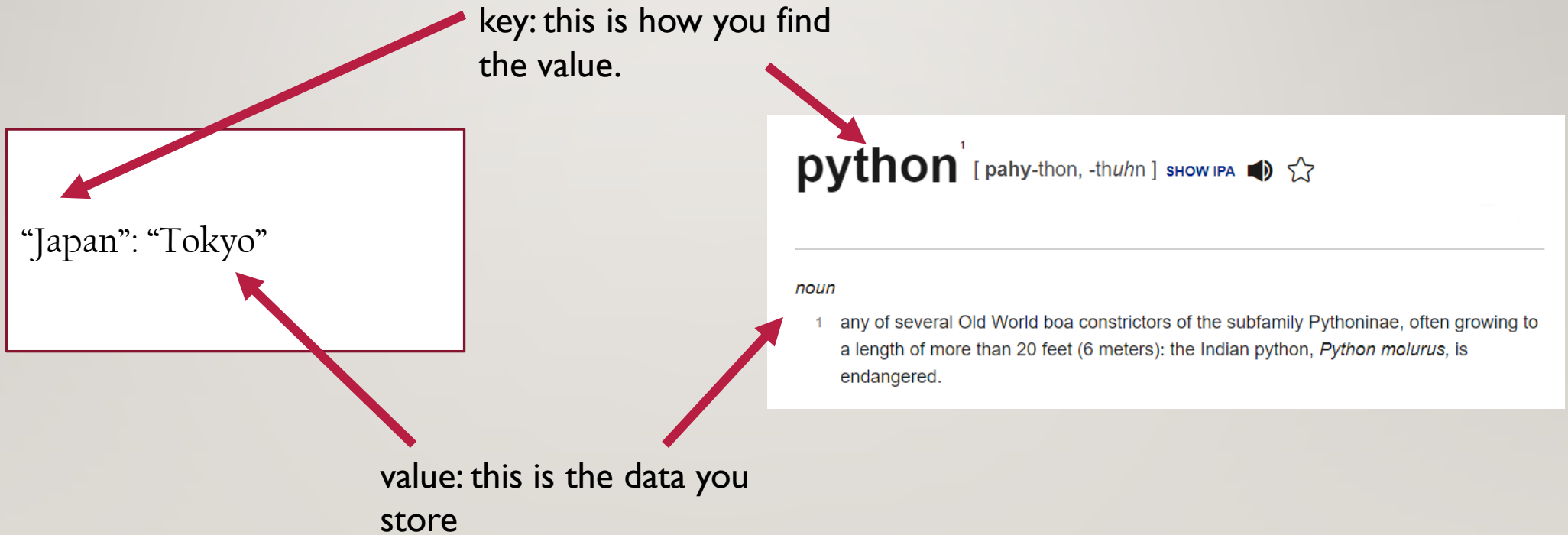
```
di = {"Japan": "Tokyo", "USA": "Washington, DC", "France": "Paris"}
```

```
print(di["USA"]) # this will print Washington, DC
```



# KEY-VALUE PAIR

---



# DIFFERENCES FROM LIST

---

- A list is ordered – the elements in it always have the same position. It is a *sequence*.
- A dictionary is not ordered. Each element is accessed by a **key**, not by a position (**index**). The order of the keys might change each time you run your program (on the same or on different computers). (Dictionaries in the newest version of Python are also ordered.)
- Speed – if you need to find a specific element quickly, dictionaries are **much, much, much** faster. You need to know the position of an item in a list, but in a dictionary, you only need to know the key.

# MOVIE DATABASE

---

- Pick your favorite actor, actress, director, etc.
- Make a dictionary of their films. Use the release year as the key, and the movie title as the value. Put at least 4 movies in the dictionary at the beginning.
- Add 2 movies to it using this syntax.

```
movie_dictionary[year] = "title"
```

- Be careful about using the same year / key: this replaces the old value.

# DICTIONARY AND LOOPS

---

- Use for to go through your movie database and print each movie title.
- Use this syntax to change the name of a movie.

```
movie_dictionary[year] = "title"
```

- Use `movie_dictionary.keys()`
- Use `movie_dictionary.values()`
- Remove an item using `movie_dictionary.pop(year)`
- Get a value using `movie_dictionary.get(year)` or `movie_dictionary[year]`

# ANOTHER REAL WORLD EXAMPLE

---

- You have a student ID number, right?
- This ID number is like the key of a dictionary – the school can use the key to access lots of information related to each student – name, birthday, address, classes, number of absences, grades in each class, etc.
- Companies do the same thing with employees and with customers – assign a number or ID to them, and then store information related to that number.



# READING A FILE

---



# OPEN A FILE

---

- To open a file, we use the built-in function `open()`
- `open()` takes two arguments

–*file*

–*mode*

# KISS

---

Keep It Simple, Stupid!!

- Let's keep our file on the desktop so we can avoid typing “C:\\blah blah blah.myfile”
- Instead, we can just do this

```
my_file = open("Hello.txt", "r")
```

# Open arguments, 2

---

• `open(file, mode)`

- `mode` is how you want to open the file
- “r” - reads the file. Error if no file
- “a” - appends. Creates the file if no file.
- “w” - opens the file to write. Creates the file if no file
- “x” - create the file. Error if the file exists.

# MODE（ファイルの読み方）

- You can also say how the file should be read.
  - “b” is for binary（二進法）→ (normal) People cannot read this format, but computers can. にしんほう
  - “t” is for text (This is the default.)

```
my_file = open("Hello.txt", "rb")
```



The model combines  
r/w/x with t or b.

This will open for  
reading in binary  
mode.



# FILE FUNCTIONS

- `file.close()` - closes the file
- `file.read()` - this gives you the data/information in the file (as a string)
- `file.write(str)` – this adds a string to the file
- `file.writelines(sequence)` – this adds a list of strings to the file

# PRACTICE

```
my_file = open("Hello.txt", "w")  
print(my_file.name)  
print(my_file.mode)  
  
my_file.close()
```

Two things to be careful about: if you open the file, you always need to close it.

If your program crashes before you close it, you'll have to restart Python.



# PRACTICE WRITING INFORMATION TO A FILE

```
my_file = open("Hello.txt", "w")
```

```
my_file.write("Hello")
```

```
my_file.close()
```

# PRACTICE READING DATA FROM A FILE

```
my_file = open("Hello.txt", "r")  
  
data = my_file.read()  
  
my_file.close()  
  
print(data)
```

# PRACTICE READING DATA FROM A FILE

- Go to Mr. Hunter's Github page → [github.com/davidcbhunter/POP2022](https://github.com/davidcbhunter/POP2022)
- Download these two files
  - Movies.txt
  - Movies\_Revenue.txt
- Practice opening them, reading the data, and closing them.
- Use the **str** function **split** or **splitlines** to make a list, then a dictionary.
- Use a for-loop to print the information.
- Use a for-loop to find the total revenue, average revenue, and the largest revenue.

