

Python Recap: Dictionaries

School Year 2023-2024

Valsalice

Dictionaries

Group data together using keys

With **variables**:

```
num1 = 42
num2 = 100
num3 = 10

print(num1)
print(num2)
print(num3)
```

With a **dict**:

```
nums = {"num1": 42, "num2": 100, "num3": 8}

print(nums)
```

Dictionaries

Anatomy of a dictionary:

1. Uses curly brackets **{ }**
2. Elements separated by comma **,**
3. Elements specified with a colon as **key : value**

```
nums = {"num1": 42, "num2": 100, }
```

```
data = {"foo": 8.2, 100: "bar"}
```

Exercise

Complete the **2_0.py** & **2_1.py** programs.

- **2_0**: Write a program that creates a new dictionary **letters** containing the three letters **a**, **b**, **c** as keys and assigning them the integer values **1**, **2** and **3**
- **2_1**: Write a program that creates a new dictionary called **pets** which stores the names of my three pets together with their age (as an integer):
 - a. **Snowball** is 3 years old
 - b. **Flopsie** is 5 years old
 - c. **Schnitzel** is 1 year old

Solution 2_0

```
letters = {"a": 1, "b": 2, "c": 3}  
  
print(letters)
```

Solution 2_1

```
pets = {"Snowball": 3, "Flopsie": 5, "Schnitzel": 1}  
print(pets)
```

Accessing Dictionary Elements

To access dictionary elements you can use the **[index]** operator.

NOTE: You can only access keys that exist



```
heights = {"Charles": 175, "Adam": 160, "Florence": 180}
```

```
print(heights["Adam"])
```

```
print(heights["Florence"])
```

ERROR:

```
print(heights["Dan"])
```

Exercise

Complete the **2_2.py** program.

Write a program that given a dictionary **grades** prints out:

1. The grade of student **Bob**
2. The grade of student **Zethus**
3. The sum of the grades of **Bob** and **Alice**
4. The average of the grades of **Alice**, **Bob** and **Charlie**

Solution 2_2

```
grades = {"Alice": 9.0, "Bob": 7.5, "Charlie": 8.3, "Zethus": 6.0}

# 1
print(grades["Bob"])

# 2
print(grades["Zethus"])

# 3
print(grades["Bob"] + grades["Alice"])

# 4
print((grades["Alice"] + grades["Bob"] + grades["Charlie"]) / 3)
```

Modifying Dictionaries

You can modify dicts in 2 ways:

```
data = {"a": 42, "b": 3}
```

1. To **insert** a new element you can use a new key

```
data["c"] = 800
```

```
data["d"] = 4.5
```

2. To **modify** an existing elements you can assign to the key

```
data["a"] = 10
```

```
data["b"] = 3.2
```

Exercise

Complete the **2_3.py** program.

Write a program that given a dictionary **scores** does the following:

1. Prints out the score of **Rob**
2. Adds a score of 4 for new user **Dan**
3. Prints the score of **Dan**
4. Replaces the score of **Rob** with the number 6
5. Prints the updated score of **Rob**
6. Prints the final **scores** dictionary

Solution 2_3

```
scores = {"Rob": 10, "Michelle": 2}

# 1
print(scores["Rob"])
# 2
scores["Dan"] = 4
# 3
print(scores["Dan"])
# 4
scores["Rob"] = 6
# 5
print(scores["Rob"])
# 6
print(scores)
```

Removing Dictionary elements

You can remove elements in a dict with the **del** function.

```
data = {"a": 42, "b": 3}
del data["a"]
print(data)
```

```
data = {"a": 42, "b": 3}
del data["b"]
print(data)
```

```
data = {"a": 42, "b": 3}
del data["a"]
del data["b"]
print(data)
```

Exercise

Complete the **2_4.py** program.

Write a program that given a dictionary **money** does the following:

1. Deletes the entry for **Rob**
2. Prints the updated **money** dictionary
3. Removes **40** euro from **Dan**
4. Prints the updated **money** dictionary
5. Adds **40** euro to **Adam**
6. Prints the updated **money** dictionary

Solution 2_4

```
money = {"Adam": 100, "Rob": 200, "Dan": 60}

# 1
del money["Rob"]
# 2
print(money)
# 3
money["Dan"] -= 40
# 4
print(money)
# 5
money["Adam"] += 40
# 6
print(money)
```

Consolidation Exercise

Complete the **2_5.py** program.

Write a program that given a list of **names**, a matching list of **measurements** and an empty dictionary **heights**:

1. Using a **for** loop add each person's name and their corresponding height into **heights**. For example person **Adam** must have a matching height of 175.
2. Calculate (and print) the sum of the heights of **Adam**, **Dan** and **Rob**
3. Add a new entry in **heights** for **Charlie** who has a height of 190
4. Print the length of **heights**. HINT: You can use the **len** function
5. Print the name of the tallest person. HINT use **max** and use a **for**-loop!
6. Remove the entry in **heights** for **Dan**
7. Print out the final dictionary **heights**

Solution 2_5

```
names = ["Dan", "Rob", "Adam", "Matt"]
measurements = [140, 165, 155, 142]
heights = {}

# 1
for i in range(len(names)):
    heights[names[i]] = measurements[i]

print(heights)

# 2
print(heights["Adam"] + heights["Dan"] + heights["Rob"])

# 3
heights["Charlie"] = 190

# 4
print(len(heights))

# 5
max_height = max(measurements)
for i in range(len(names)):
    name = names[i]
    if heights[name] == max_height:
        print(name)

# 6
del heights["Dan"]

# 7
print(heights)
```

Additional Dictionary Functions

Additional functions that operate on dicts

- Get the length of the dict: **len**

```
len({"a": 42, "b": 3})
```

```
len({6: 4})
```

```
len({})
```

- Get the keys of a dict as a list: **.keys()**

```
list({"a": 42, "b": 3}.keys())
```

- Get the values of a dict as a list: **.values()**

```
list({"a": 42, "b": 3}.values())
```

Exercise

Complete the **2_6.py** program.

Write a program that given a dictionary `colors`,

1. Prints the length of the dictionary.
2. Converts the keys of the dictionary to a list and prints it.
3. Converts the values of the dictionary to a list and prints it.

Solution 2_6

```
colors = {"red": "#FF0000", "green": "#00FF00", "blue": "#0000FF"}

# 1
print(len(colors))

# 2
print(list(colors.keys()))

# 3
print(list(colors.values()))
```

Iterating Dictionaries

Python provides multiple ways to **iterate over dicts**.

The most used methodologies are:

Key-iteration:

```
data = {"a": 4, "f": 1, "z": 8}

for key in data:
    value = data[key]
    print(key, value)
```

For-each loop:

```
data = {"a": 4, "f": 1, "z": 8}

for key, value in data.items():
    print(key, value)
```

The output of the two snippets is identical

Exercise

Complete the **2_7.py** & **2_8.py** programs.

```
inventory = {"apples": 30, "bananas": 45, "cherries": 25}
```

- **2_7**: Given the dictionary `inventory`, iterate over the dictionary and print each fruit and its quantity in the format:
"There are [quantity] [fruit]"
- **2_8**: Given the dictionary `inventory`, iterate over the dictionary and calculate the total quantity of fruits. Print the total.

Solution 2_7

```
inventory = {"apples": 30, "bananas": 45, "cherries": 25}

for key, value in inventory.items():
    print("There are " + str(value) + " " + key)
```

Solution 2_8

```
inventory = {"apples": 30, "bananas": 45, "cherries": 25}

total = 0

for key, value in inventory.items():
    total += value

print(total)
```


Consolidation Exercise

Complete the **2_9.py** program.

Write a program that performs the following tasks:

1. Create a dictionary `student_marks` with at least five students' names as keys and their marks as values. Marks go from `2` to `10`.
2. Add a new student `"Emily"` with a mark of `8.7`.
3. Calculate and print the average mark of the class.
4. Print the name of the student with the highest mark (use a **loop** for this).

Solution 2_9

```
# 1
student_marks = {"John": 76, "Sarah": 82, "Alex": 90, "Rita": 88, "Tom": 79}

# 2
student_marks["Emily"] = 87

# 3
average mark = sum(student_marks.values()) / len(student_marks)
print("Average mark: " + str(average_mark))

# 4
student_name = ""
max mark = -1
for student, mark in student_marks.items():
    if mark > max mark:
        max mark = mark
        student_name = student
print("Highest mark student: " + student_name)
```

End of Python Recap

Don't hesitate to reach out
on Classroom with any questions!