

Metadata

Course: DS 5100
Term: Summer 2023 Residential
Module: M02 Homework
Author: R.C. Alvarado
Date: 7 July 2023

Student Info

- Name: Jacqui Unciano
- Net ID: jdu5sq
- URL of this file in GitHub: <https://github.com/jacquiunciano/DS5100-jdu5sq/tree/main/lessons/M02/hw02.ipynb>

Instructions

In your **private course repo on Rivanna**, write a Jupyter notebook running Python that performs the numbered tasks below. For each task, create a code block to perform the task.

Save your notebook in the `M02` directory as `hw02.ipynb`.

Add and commit these files to your repo.

Then push your commits to your repo on GitHub.

Be sure to fill out the **Student Info** block above.

To submit your homework, save the notebook as a PDF and upload it to GradeScope, following the instructions.

Data

Table 1: GRADES

name	grade
Jon	95
Mike	84
Jaime	99

Table 2: TOUCHDOWNS

name	touchdowns
Alex	2
Patrick	4
Tom	1
Joe	3
Alex	1

Tasks

Task 1

Using the data in Table 1, create a dictionary called `gradebook` where the keys contain the names and the values are the associated grades. Print the dictionary. (1 PT)

```
In [2]: # Put code here
gradebook = {
    "Jon":95,
    "Mike":84,
    "Jaime":99
}
gradebook
```

```
Out[2]: {'Jon': 95, 'Mike': 84, 'Jaime': 99}
```

Task 2

Index into the `gradebook` to print Mike's grade. Do NOT use the `get()` method for this. (1 PT)

```
In [3]: # Put code here
gradebook["Mike"]
```

```
Out[3]: 84
```

Task 3

Attempt to index into `gradebook` to print Jeff's grade. Show the result. Do NOT use the `get()` method for this. (1 PT)

```
In [4]: # Put code here
gradebook["Jeff"]
```

```
-----
KeyError                                Traceback (most recent call last)
Cell In[4], line 2
      1 # Put code here
----> 2 gradebook["Jeff"]

KeyError: 'Jeff'
```

Task 4

Using Table 2, build a list from the names called `names` and print it. (1 PT)

```
In [6]: # Put code here
names = ["Alex", "Patrick", "Tom", "Joe", "Alex"]
names
```

```
Out[6]: ['Alex', 'Patrick', 'Tom', 'Joe', 'Alex']
```

Task 5

Sort the list in ascending order and print it. (1 PT)

```
In [8]: # Put code here
sorted(names)
```

```
Out[8]: ['Alex', 'Alex', 'Joe', 'Patrick', 'Tom']
```

Task 6

Build a set from the names in Table 2 and print it. (1 PT)

```
In [9]: # Put code here
set(names)
```

```
Out[9]: {'Alex', 'Joe', 'Patrick', 'Tom'}
```

Task 7

Build a dictionary from the touchdowns data, calling it `td`, and print it. Use lists to store the values. Remember that dictionary keys must be unique. (1 PT)

```
In [19]: # Put code here
td = {
    "Alex": [2, 1],
    "Patrick": 4,
    "Tom": 1,
    "Joe": 3
}
td
```

```
Out[19]: {'Alex': [2, 1], 'Patrick': 4, 'Tom': 1, 'Joe': 3}
```

Task 8

Compute the sum of Alex's touchdowns using the appropriate built-in function.

```
In [11]: # Put code here  
sum(td["Alex"])
```

```
Out[11]: 3
```

Task 9

Get the keys from `td` and save them as a sorted list `list1`. Then get a set from `names` and save them as a sorted list called `list2`. Compare them with a boolean operator to see if they are equal. (2 PTS)

```
In [23]: # Put code here  
list1 = sorted(td.keys())  
list1
```

```
Out[23]: ['Alex', 'Joe', 'Patrick', 'Tom']
```

```
In [24]: list2 = sorted(set(names))  
list2
```

```
Out[24]: ['Alex', 'Joe', 'Patrick', 'Tom']
```

```
In [25]: list1 == list2
```

```
Out[25]: True
```