M02-HW-KEY

August 19, 2023

1 Metadata

```
Course:   DS 5100
Term:     Fall 2023
Module:   M02 Homework
Author:   R.C. Alvarado
Date:     19 August 2023 (revised)
```

2 Student Info

- Name:
- Net ID:
- URL of this file in GitHub:

3 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 GitHib.

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.

10 Points

4 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
```

5 Tasks

5.1 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)

[1]:
```python
# Put code here
```

[3]:
```python
gradebook = {
    'Jon': 95,
    'Mike': 84,
    'Jaime': 99
}
```

[5]:
```python
print(gradebook)
```

```
{'Jon': 95, 'Mike': 84, 'Jaime': 99}
```

5.2 Task 2

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

[6]:
```python
# Put code here
```

[8]:
```python
print(gradebook['Mike'])
```

```
84
```

5.3 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)

[9]:
```python
# Put code here
```

[10]:
```python
print(gradebook['Jeff'])
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
Cell In[10], line 1
----> 1 print(gradebook['Jeff'])

KeyError: 'Jeff'
```

5.4 Task 4

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

[11]:
```
# Put code here
```

[12]:
```
names = ['Alex', 'Patrick', 'Tom', 'Joe', 'Alex']
```

[13]:
```
print(names)
```

```
['Alex', 'Patrick', 'Tom', 'Joe', 'Alex']
```

5.5 Task 5

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

[14]:
```
# Put code here
```

[15]:
```
print(sorted(names))
```

```
['Alex', 'Alex', 'Joe', 'Patrick', 'Tom']
```

5.6 Task 6

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

[13]:
```
# Put code here
```

[14]:
```
print(set(names))
```

```
{'Patrick', 'Joe', 'Tom', 'Alex'}
```

5.7 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)

[16]:
```
# Put code here
```

[17]:
```
td = {
    'Alex': [2, 1],
    'Patrick': [4],
    'Tom': [1],
    'Joe': [3]
}
```

[18]:
```
print(td)
```

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

5.8 Task 8

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

[19]:
```
# Put code here
```

[20]:
```
sum(td['Alex'])
```

[20]:
```
3
```

5.9 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)

[21]:
```
# Put code here
```

[22]:
```
list1 = sorted(list(td.keys()))
list2 = sorted(list(set(names)))
```

[23]:
```
list1 == list2
```

[23]:
```
True
```

[24]:
```
list1
```

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

[25]:
```
list2 = set(names)
```

[26]:
```
list2
```

[26]:
```
{'Alex', 'Joe', 'Patrick', 'Tom'}
```

[27]:
```
type(list2)
```

[27]:
```
set
```

[28]:
```
list1 == list2
```

```
[28]: False
```

```
[29]: list1 == sorted(list2)
```

```
[29]: True
```

```
[30]: type(sorted(list2))
```

```
[30]: list
```