Nested data and iteration

Warm-up

1. Write a function called `word_freq_counts` that takes in a string and then prints a table of words in the string in alphabetical order together with the number of times each word occurs. Case should be ignored. A sample run of the program might look this this:

```
freq_counts('ThiS is String with Upper and lower case Letters String is awesome')
```

```
and 1
awesome 1
case 1
is 2
letters 1
lower 1
string 2
this 1
upper 1
with 1
```

Model 1 Nested Lists

Elements in a list can be of sequence type (string or list), for example, in a list of words, each element is a string type. Similarly, here is an example of a list of lists:

```
states = [
    ['AL','AK','AZ','AR'],
    ['CA', 'CO','CT'],
    ['DC','DE'],
    ['FL'],
    ['GA'],
    ['HI'],
    ['ID','IL','IN','IA']
]
```

The `states` list contains sub-lists with states that start with the same letter.

2. Evaluate each expression in order and record the output for each line in the second column.

Python code	Output
`print(states[0])`	
`print(states[-1])`	
`print(states[4][-1])`	
`print(states[5][0])`	
`print(len(states))`	
`print(len(states[1]))`	
`print(len(states[3]))`	
`print(len(states[3][0]))`	
`print(len(states[3][1]))`	
`print(states[3][0][0])`	

3. What does the following code snippet print?

```
for sublist in states:
    letters = ''
    for state in sublist:
        letters += state[1]
    print(letters)
```

4. Modify the code in the previous problem to print all the letters inside the list, that is: `'ALAKAZARCACOCTDCDEFLGAHIIDILINIA'`

5. Write a function called `max_states` that takes in the list of states and returns the maximum size of its sublists.

6. Write a function called `min_states` that takes in the list of states and returns the first sublist with minimum size.

7. **Challenging:** Modify the code in the previous problem to print all the unique letters inside the list, that is: `'ACDFGHILKZROTEN'`

Model 2 Nested Dictionaries

Collections/containers (sequence-type like strings and lists, and dictionaries/maps) can be nested in arbitrary ways. For example, the following data could be described as a "dictionary of dictionaries of integers and lists of strings":

```
movies = {
    "Casablanca": {
        "year": 1942,
        "genres": ["Drama", "Romance", "War"],
    },
    "Star Wars": {
        "year": 1977,
        "genres": ["Action", "Adventure", "Fantasy"],
    },
    "Groundhog Day": {
        "year": 1993,
        "genres": ["Comedy", "Fantasy", "Romance"],
    },
}
```

8. Evaluate the following expressions in the order that they are listed:

Python code	Output
`movies`	
`movies["Casablanca"]`	
`movies["Casablanca"]["year"]`	
`movies["Casablanca"]["genres"]`	
`type(movies)`	
`type(movies["Casablanca"])`	
`type(movies["Casablanca"]["year"])`	
`type(movies["Casablanca"]["genres"])`	
`len(movies)`	
`len(movies["Casablanca"])`	
`len(movies["Casablanca"]["year"])`	
`len(movies["Casablanca"]["genres"])`	
`for key in movies:` `  print(key)`	
`for key, val in movies.items():` `  print(key, val)`	

9. Explain the `TypeError` you encountered.

10. In the expression `movies["Casablanca"]["genres"]`, describe the purpose of the strings `"Casablanca"` and `"genres"`.

11. When iterating a dictionary using a `for` loop (i.e., `for x in movies`), what gets assigned to the variable?

12. What is wrong with the following code that attempts to `print` each movie?

```
for i in range(len(movies)):
    print(movies[i])
```

13. Write nested loops that outputs (prints) every *genre* found under the `movies` dictionary. Trace your code to ensure that it outputs a total of nine lines.

14. Each movie in Model 2 has a title, a year, and three genres.

a) Is it necessary that all movies have the same format?

b) Name one advantage of storing data in the same format:

c) Show how you would represent The LEGO Movie (2014) with a runtime of 100 min and the plot keywords "construction worker" and "good cop bad cop".