Practice Exercises for Nested Data Structures

Solve each of the practice exercises below. Each problem includes two CodeSkulptor3 links: one for a template that you should use as a starting point for your solution and one to our solution to the exercise.

1. Write an expression that defines a list `nested_list` consisting of five empty lists. Empty lists template --- Empty lists solution
2. Write an expression that defines a list `nested_list` of length five whose items themselves are lists consisting of three zeros. List of zero lists template --- List of zero lists solution
3. In Python, a list comprehension is one line statement that can be used to define simple, but interesting lists succintly. Create a list `zero_list` consisting of 3 zeroes using a list comprehension. As an extra challenge, create the list `nested_list` from the previous question using a nested list comprehension. List comprehension template --- List comprehension solution
4. Given the list `nested_list` as defined in the provided template, write an expression that returns the item in `nested_list` that has value 7. Return 7 template --- Return 7 solution
5. Consider the list `nested_list` as defined in the provided template. Attempting to modify one item in `nested_list` has the unexpected effect of modifying several items. Examine this example and enter an explanation for this behavior. Reference template --- Reference solution
6. Write an expression `list_dicts` that defines a list consisting of five empty dictionaries. List of empty dicts template --- List of empty dicts solution
7. Write a function `dict_copies(my_dict, num_copies)` that takes a dictionary `my_dict` and an integer `num_copies` and returns a list consisting of `num_copies` copies of `my_dict`. Dict copies template --- Dict copies solution
8. Write a function `make_dict_lists(length)` that takes an integer `length` returns a dictionary whose keys are in `range(length)` and whose corresponding values are lists of zeros whose length match the key. Make dict lists template --- Make dict lists solution
9. **Challenge:** Define a dictionary `grade_table` whose keys corresponds to names in the first column of the table below and whose corresponding values are a list of the grades in the name's row. Simple grade table template --- Simple grade table solution

Names	Assign #1	Assign #2	Assign #3	Assign #3
Joe	100	98	100	13
Scott	75	59	89	77
John	86	84	91	78

10. **Challenge:** Define a function `make_grade_table(name_list, grades_list)` that takes a list of names `name_list` and a list of grade lists `grades_list` and returns a dictionary whose keys corresponds to names `name_list` and whose corresponding values are the items `grades_list`. As a

challenge, use the Python function zip() to simplfy the logic of your loop that creates the output dictionary.
Make grade table template --- Make grade table solution

Mark as completed