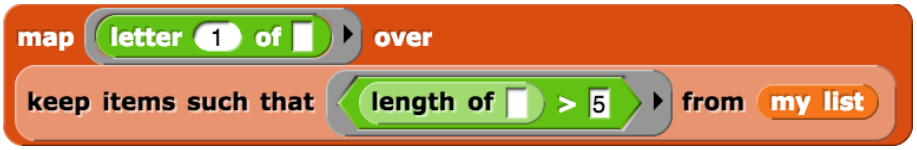
Discussion 11: Python Data Structures

List Comprehension Warm-Up

1. Consider the Snap! code given below:



Translate this expression into Python using a list comprehension.

_____

2. Write a list comprehension over `list_of_lists` (a list of sublists) that calculates the sum of each sublist and adds each sum to a new list. You can assume the sublists in `list_of_lists` contain only numbers.

For example if `list_of_lists = [[1, 2], [-5, 4]]`, your expression should output `[3, -1]`.

To find the sum of a list, you can call python's built-in `sum` function on the list.

```
>>>sum([1, 2])
3
```

_____

3. Write a list comprehension that finds the index of an item in a list. You may assume that the item appears only once in the list. If you get stuck, it may be easier to first write this function using a `for` loop, then translate your code into a list comprehension.

```
def find_index(item, lst):

    return _____
```

Planning Your Phase 1

1. In the table below, write Python code to execute the listed commands on `class_dict`.

```
class_dict = {'Math':'1A', 'English':'R1A'}
```

`Add the key 'CS' with the value '10'`	
`Access the value of 'Math'`	
`Change the value of 'Math' to '1B'`	
`Check if 'UGBA' is a key in class_dict`	
`Check if '10' is a value in class_dict`	
`Get a list of the keys in class_dict`	

2. Can you access a key, value pair in a dictionary by its index?

3. Are keys and/or values in a dictionary returned in a predictable order?

4. Can dictionaries have duplicate keys? What about duplicate values?

Dictionary Practice

```
fav_numbers = {'Dan': 18, 'Alonzo': 12, 'Oski': 7, 'Carol Christ': 152}

nums = [7, 12]
```

1. On the lines below, write Python code that increments each person's favorite number by the length of their name.

_____

_____

2. On the lines below, use a list comprehension to output a list of people whose favorite numbers are in `nums`.

_____

3. Write a function `merge_dicts` that takes two dictionaries as input, and returns a new dictionary that contains all entries from both input dictionaries. Your function should not modify the inputs. You can assume that both input dictionaries have strings as keys and numbers as values. For any keys present in both input dictionaries, the corresponding value in the output dictionary should be the sum of the values in the inputs.

```
>>> dict1 = {‘Dan’: 10, ‘Oski’: 15}
>>> dict2 = {‘Alonzo’: 5, ‘Oski’: 20, ‘Dan’: -10}
>>> merge_dicts(dict1, dict2)
{‘Dan’: 0, ‘Alonzo’: 5, ‘Oski’: 35}
```

4. Assume we have defined `food_dict` in the Python interpreter, as below. What will be displayed after each of the following lines executes? If the result is an error message, just write "Error." Each subproblem is independent and does not depend on the other subproblems.

```
>>> food_dict = {"fruit": "apple", "veggie": "carrot", "beverage": "water",
"grain": "rice"}
>>> len(food_dict)
```

_____

```
>>> list(food_dict)
```

_____

```
>>> food_dict[0]
```

_____

```
>>> (‘fruit’ in food_dict) and (‘apple’ in food_dict)
```

```
>>> ("fruit" in food_dict.keys()) and ("apple" in food_dict.values())
```

```
>>> for food in food_dict:
...     food += "s"
>>> food_dict
```

```
>>> def recursion_is_fun(dict1, dict2):
...     if dict2 == {}:
...        return dict1
...    dict2.pop(list(dict2)[0])
...    return recursion_is_fun(dict1, dict2)
>>> copy = food_dict
>>> recursion_is_fun(food_dict, copy)
```

```
>>> more_food = {"protein": "chicken"}
>>> food_dict["more food"] = more_food
>>> food_dict
```
