Discussion 12: Python II

Planning Your Phase I

1. In the table below, write out Python code to execute the following commands on `my_dict.`

`my_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 my_dict	
Check if '10' is a value in my_dict	
Get a list of the keys in my_dict	

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

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

4. Can dictionaries have duplicate keys?

Dictionary Practice

`fav_numbers = {'Schuyler': 120, 'Matthew': 12, 'Mansi': 7}`

1. Increment each person's favorite number by the length of their name.

_____

_____

2. Print a list of all people from `fav_numbers` whose favorite numbers are even.

_____

_____

_____

_____

_____

3. Write a function that merges two dictionaries. Both dictionaries will have strings as keys and numbers as values. If you come across a key that's present in both dictionaries, add together its values in the merged dictionary.

```
>>> dict1 = {'Schuyler': 10, 'Matthew': 15}
>>> dict2 = {'Matthew': 5, 'Mansi': 4}
>>> merge_dicts(dict1, dict2)
{'Schuyler': 10, 'Matthew': 20, 'Mansi': 4}

def merge_dicts(d1, d2):
```

4. Assume we have executed the code below in the Python interpreter. What will be displayed after each of the following code snippets executes? If the result is an error message, just write "Error." Assume that these lines are executed independently, NOT sequentially. (Continued on the next page)

```
>>> 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
```

_____

Recursion in Python

1. In the table below, translate the Snap! code to Python. This Python code will be useful in writing recursive functions in Python.

all but first of (my list)	
all but last of (my list) Note: This block doesn't actually exist in Snap!	
all but first of (all but last of (my list))	
5 in front of (my list)	
append (my list) (your list) ◀▶	

Note: The Python syntax for the first three rows would be the same whether your variable is a list or a string.

2. Translate the following code from Snap! to Python:
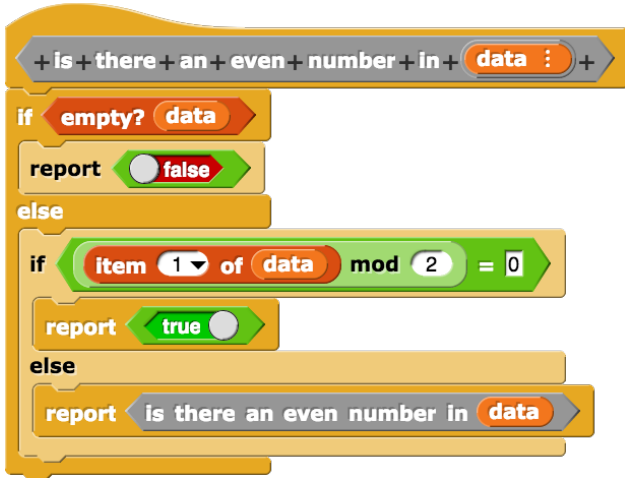
3. Write a recursive function in Python that removes certain items from a list, as described below. It should create a new list, not mutate the input list.

```
>>> delete_elements(4, [4, 5, 6])
[5, 6]
>>> delete_elements(7, [4, 5, 6])
[4, 5, 6]
>>> delete_elements(4, [4, 5, 6, 4])
[5, 6]

def delete_elements(x, lst):
```