Programming Assignment 3

Due Oct 6, 2020 by 11:59pm **Points** 8 **Submitting** a file upload
Available Sep 28, 2020 at 5pm - Oct 6, 2020 at 11:59pm 8 days

This assignment was locked Oct 6, 2020 at 11:59pm.

1. Even (2 points)

Write code to generate a list of all even numbers that are divisible by 3 between 3 and 1000.

Hint: use list comprehension.

For extra practice, make this a function to find all numbers divisible by a parameter k between k and another parameter n

2a. Union (1 points)

Make a union function which takes in 2 lists as parameters and returns a list. It should return a list that contain all elements that are in either list. Your results list should contain no duplicates.

For example, if the two lists are [1, 5, 6, 5, 2] and [3, 5, 1, 9] then the result list is [1, 5, 6, 2, 3, 9].

Note that the list sizes don't have to be equal.

Note that the input lists could have duplicates, but your resulting list should not.

The order of the elements in the result list is not important. Also, neither list passed in should be modified. You should probably check this last point by printing out one of the original lists after the call. That is, something like:

```
a = [1,2]
b = [2,3,4]
print(union(a,b)) # should print [1,2,3,4]
```

```
print(a) # a should be [1,2]
print(b) # b should b [2,3,4]
```

2b. Intersection (1 points)

The other function you should write is intersection. Everything described above applies here as well, only this function should return a list that contains only the elements that are in both lists (no duplicates). So for the a and b above:

print(intersection(a,b)) # should print [2]

Note: do not use Python Sets to do this - this problem is for you to practice using list.

Again, the lists passed in should not be modified. You should probably check this last point by printing out one of the original lists after the call. That is, something like:

3. rotate (2 points)

Write a function that takes a list and an integer k and returns a **new** list that is rotated k spots to the left with wrap around.

For example: if the list is [1, 4, 7, 13, 9] and k = 2 then the new list is shifted three spots to the left with wrap around so new list becomes: [7, 13, 9, 1, 4]

Note: You could have list of length 10, and have a k larger than that like 15, make sure to consider how that would work.

Again, the list passed in should not be modified. You should probably check this last point by printing out one of the original lists after the call. That is, something like:

4. contactList (2 points)

For this problem you are to create a contact list with "parallel lists", one for names the other for their phone numbers.

Below you will find the starting code you are to use. It defines the global lists, testing code, and function signatures. It is your job to fill

in the functions.

Note: do **not** use Python Dictionaries to do this - this is for you to practice using Lists.

```
# Initially empty contact list
names = []
numbers = []

# Should display each of the names and associated numbers
def displayContactList():
''' TODO Your code here '''

# Add the name/number pair to the listts
def addContact(name, number):
''' TODO Your code here'''

# Return the number associated with the name, or None if name doesn't exist
def lookupNumber(name):
''' TODO Your code here'''

# Remove the contact name/number pair from the lists.  Do nothing if name doesn't        exist
def removeContact(name):
''' TODO Your code here '

# Testing code
addContact("Daniel", 1234)
addContact("Alyssa", 5678)
addContact("Riley", 3456)
displayContactList() # Should display 3 names/numbers
print(lookupNumber("Alyssa")) # 5678
removeContact("Alyssa")
displayContactList() # Displays Daniel/Riley's contacts
print(lookupNumber("Alyssa")) # None
```

Some Rubric (2)				
Criteria	**Ratings**			**Pts**
1)	**2 pts** **Full Marks**	**1 pts** **Half Marks**	**0 pts** **No Marks**	2 pts
2a)	**1 pts** **Full Marks**	**0.5 pts** **Half Marks**	**0 pts** **No Marks**	1 pts
2b)	**1 pts** **Full Marks**	**0.5 pts** **Half Marks**	**0 pts** **No Marks**	1 pts
3)	**2 pts** **Full Marks**	**1 pts** **Half Marks**	**0 pts** **No Marks**	2 pts
4)	**2 pts** **Full Marks**	**1 pts** **Half Marks**	**0 pts** **No Marks**	2 pts
				Total Points: 8