ANA-522-OL1 Spring 2022
Mod01 Week02 Lab: Python Part Two
Due: Friday January 21st at midnight

Lists help programmers manage larger amounts of data by allowing several (or even many) values to be stored in one variable. This makes it practical to solve larger problems that involve many data values. You are expected to:

- Create a variable that holds a list of values
- Modify a list by appending, inserting, updating and deleting elements
- Search a list for a value
- Display some or all of the values in a list

0.1 Exercise 1: Below and Above Average

Write a program that read a number N from the user. Let the program generate N random values. Afterwards, your program should display the average of all of the random values generated. Then the program should display all of the above average values, followed by all of the average values (if any), followed by all of the below average values. An output of a running example:

> To generate random numbers use module random, then invoke one of the methods in the following :
> import random
> random.random() # Random float: 0.0 <= x < 1.0
> random.randrange(10) # Integer from 0 to 9 inclusive
> random.randint(1,10) # Integer from 1 to 10 inclusive
> See methods available in Python random module at https://www.w3schools.com/python/module_random.asp

0.2 Exercise 2: Random Lottery Numbers

In order to win the top prize in a particular lottery, one must match all 6 numbers on his or her ticket to the 6 numbers between 1 and 49 that are drawn by the lottery organizer. Write a program that generates a random selection of 6 numbers for a lottery ticket. Ensure that the 6 numbers selected do not contain any duplicates. Display the numbers in ascending order.

> To generate random numbers use module random, then invoke one of the methods in the following :
> import random

random.random() # Random float: 0.0 <= x < 1.0
random.randrange(10) # Integer from 0 to 9 inclusive
random.randint(1,10) # Integer from 1 to 10 inclusive
See methods available in Python random module at https://www.w3schools.com/python/module_random.asp

0.2.1 Function with List Exercise

0.3 Exercise 3:Remove Outliers

When analysing data collected as part of a science experiment it may be desirable to remove the most extreme values before performing other calculations.

Write a function that takes a list of values and a non-negative integer, N, as its parameters. The function should create a new copy of the list with both the n largest elements and the n smallest elements removed. Then it should return the new copy of the list as the sole function's result as return. The order of the elements in the returned list does not have to match the order of the elements in the original list.

To demonstrate the function you created in the program, call the function with a list of randomly generated numbers, that the function would remove the two largest and two smallest values from it, and return an updated new copy. Display the new copy of the list with the outliers removed, followed by the original list. Your program should generate an appropriate error message if the user enters less than 4 values in the case of N is 2. See the examples:

0.4 Exercise 4: Is a List already in Sorted Order?

Write a function that determines whether or not a list of values is in sorted order (either ascending or descending). The function should return True if the list is already sorted. Otherwise it should return False. Write a main program that reads a list of numbers from the user and then uses your function to report whether or not the list is sorted.

> One solution idea to check whether a list of values are in a correct order is to check all pairs of neighboring values on the list to see if all pairs are in correct order. As soon as there is one pair on the list has the incorrect order, then it's time to conclude that the whole list is not in correct order. Otherwise, all numbers on the list are in their correct order. Also make sure you consider these questions when completing this exercise: Is an empty list in sorted order? What about a list contains only one element?

Some examples:

```
(Scenario I)
Input the next number:1
Input the next number:2
Input the next number:3
Input the next number:4
```

```
Input the next number:5
[1, 2, 3, 4, 5]
This list is sorted.

(Scenario II)
Input the next number:1
Input the next number:2
Input the next number:3
Input the next number:5
Input the next number:4
[1, 2, 3, 5, 4]
This list is NOT sorted.

(Scenario III)
Input the next number:5
Input the next number:4
Input the next number:3
Input the next number:2
Input the next number:1
[5, 4, 3, 2, 1]
This list is sorted.

(Scenario IV)
[]
This list is sorted.

(Scenario V)
Input the next number:9
[9]
This list is sorted.
```

Dictionary is another data structure that can be used in Python programs to manage large amount of data. While many problems can be solved with lists or conditional statements, in some occasions solutions are well suited using dictionaries. You are expected to:

- Create a new variable that holds a dictionary
- Add a key-value pair to a dictionary
- Update the value associated with a key in a dictionary
- Iterate over all of the keys and/or values in a dictionary

0.5 Exercise 5: Reverse Lookup

Write a function named reverseLookup that finds all of the keys in a dictionary that map to a specific value. The function will take the dictionary and the value to search for as its only parameters. It will return a (possibly empty) list of keys from the dictionary that map to the provided value.

Include additional statements that demonstrates the reverseLookup function as part of your solution to this exercise.

Your program can create a dictionary or use the dictionary included and then show that the reverseLookup function works correctly when it returns multiple keys, a single key, and no keys.

The output from the given dictionary:

[1]:
```
# Reverse Loopkup

data = { 'A': 3,
         'B': 7,
         'C': 8,
         'D': 8,
         'E': 7,
         'F': 3,
         'G': 2,
         'H': 3,
         'I': 8,
         'J': 3
}
```

0.6 Exercise 6:Two Dice Simulation

In this exercise you will simulate 1,000 rolls of two dice. Begin by writing a func- tion that simulates rolling a pair of six-sided dice. Your function will not take any parameters. It will return the total that was rolled on two dice as its only result. Write the main program that uses your function to simulate rolling two six-sided dice 1,000 times.

As your program runs, it should count the number of times that each total occurs. Then it should display a table that summarizes this data. Express the frequency for each total as a percentage of the total number of rolls. Your program should also display the percentage expected by probability

theory for each total. Sample output is shown below.

```
Total    Simulated    Expected
           Percent     Percent
    2         2.90        2.78
    3         6.90        5.56
    4         9.40        8.33
    5        11.90       11.11
    6        14.20       13.89
    7        14.20       16.67
    8        15.00       13.89
    9        10.50       11.11
   10         7.90        8.33
   11         4.50        5.56
   12         2.60        2.78
```

0.7 Exercise 7:Write Out Numbers in English

While the popularity of cheques as a payment method has diminished in recent years, some companies still issue them to pay employees or vendors. The amount being paid normally appears on a cheque twice, with one occurrence written using digits, and the other occurrence written using English words. Repeating the amount in two different forms makes it much more difficult for an unscrupulous employee or vendor to modify the amount on the cheque before depositing it.

In this exercise, your task is to create a function that takes an integer between 0 and 999 as its only parameter, and returns a string containing the English words for that number. For example, if the parameter to the function is 142 then your function should return "one hundred forty two". Use one or more dictionaries to implement your solution rather than large if/elif/else constructs. Include a main program that reads an integer from the user and displays its value in English words. See example outputs:

```
123 :  One Hundred Twenty Three
812 :  Eight Hundreds Twelve
802 :  Eight Hundreds and Two
 14 :   Fourteen
 67 :   Sixty Seven
  9 :   Nine
```