CS10 Lab 4 (Lesson 1 to 51)
(5 points total 1 point for each question)
1 test run for each question(must use the list, tuple or set data provided in blue)
(lists, tuples, sets)

1. Follow the Lab instructions and video on how to submit labs.
2. Follow the steps in the Lab instructions to separate each question with question number, short description of the question.
3. For each question you must provide the number of test runs specified in the question (use the sample test run data already provided plus the rest yourself). For each question that do not have the specified number of test runs, a zero will be given even if the program runs correctly.
4. You must use the data in the sample test runs that are given in the question. Provide your own data whenever there is no sample run data.
5. If the program does not run a zero will be given
6. Your test run must be exactly the same as the sample test run specifications provided. If the program runs but does not fulfill all the specifications stated in the question, a zero score will be given.
7. Use topics covered in class up till sets(Lesson 1 to Lesson 51). Using global variables and topics or tools not in Lesson 1 to Lesson 51 will result in a zero for that question.

   ** all functions must include docstrings and type hints(annotations). Programs with functions must have the statement at the end of the program before calling the main() function.

```
if __name__ == "__main__":
```

NO GLOBAL VARIABLES ALLOWED FOR THIS LAB, a zero will be given to any program with global variable. The break and continue statements are not allowed and a zero will be given for the program that contains those two statements.

1. Write a program to create a list of numbers in the range of 1 to 10. Then delete all the even numbers from the list and print the final list.
   def makeList() – function to create the list of numbers from 1 to 10
   1,2,3,4,5,6,7,8,9,10 (use a for loop to create the list, no user input required)
   def delEven() – function to delete even numbers from list
   def main()- call the above functions and print the original list and the list after deleting even numbers

   Sample Test run 1(your output should like below)

   Original List : [1,2,3,4,5,6,7,8,9,10]
   List after deleting even numbers: [1,3,5,7,9]

2. Write a program to remove all duplicates from a list.
   a. def createList()- allow the user to input these values into the list (10 integers only)
      1, 2, 3, 4, 5, 6, 7, 6, 5, 4
   b. def removeDuplicates() – function will remove duplicate values
   c. def main()- call the above functions and print the original list and the list after removing duplicates

Sample Test run 1(your output should like below, blue user input)

Enter a number to be added to a list: 1
Enter a number to be added to a list: 2
Enter a number to be added to a list: 3
Enter a number to be added to a list: 4
Enter a number to be added to a list: 5
Enter a number to be added to a list: 6
Enter a number to be added to a list: 7
Enter a number to be added to a list: 6
Enter a number to be added to a list: 5
Enter a number to be added to a list: 4

Original list : [1, 2, 3, 4, 5, 6, 7, 6, 5, 4]
List after removing duplicates : [1, 2, 3, 4, 5, 6, 7]

3. write a program that creates a tuple of numbers(both positive and negative). Make a new tuple that has only positive values from this tuple and sort it in descending order.

   a. def makeTuple()- function to allow user to enter these 10 values into a tuple
      -10, 1, 2, -9, 3, 4, -8, 5, 6 ,7
      hint : user enters a string of the 10 numbers separated by commas
      split each of the numbers by the comma using .split() to a list
      convert list to tuple
   b. def makePositive() – function will make a new tuple with the positive numbers in descending order from tuple created in makeTuple()
   c. def main()- call the above functions and print original tuple and new tuple with descending positive numbers.

Sample Test run (your output should like below, blue user input)

Enter the values in a tuple: -10, 1, 2, -9, 3, 4, -8, 5, 6, -7

Original Tuple : (-10, 1, 2, -9, 3, 4, -8, 5, 6, -7)
New Tuple with Positive numbers : (6, 5, 4, 3, 2, 1)

4. Write a program using list to **read a sequence of string values** and prints them, and highlighting the largest number (76.9 <== this is the largest number). These are the required functions
   a. def readValues() – to read in a series of string values by the user into a list.
   b. def findLargest()- pass the list into the function and find largest
   c. def display() – display the list and highlight the largest
   d. use a main function to call the above functions

   Sample Test run (your output should like below, user input string values in blue)

   Please enter values, Q to quit:
   34
   56.7
   4
   9
   76.9
   55.4
   Q

   Here are the numbers in the list
   34
   56.7
   4
   9
   76.9 <== this is the largest number
   55.4

The question states that you must input string values. Which means that your values that you enter cannot be

```
values = int(input("Please enter values, Q to quit"))
```

it will have to be

```
values = input("Please enter values, Q to quit")
```

since you do not know how many times to do loop , you will have to use a while loop to quit with 'Q'

In the function def findLargest() , you cannot use the method .max() to find the largest value, you will have to write the codes to find the largest. Which I believe we have done the codes in loops finding the largest number, you will have to apply list operations.

5. **(Sets)**Given the sets below, write python program to(functions not required just the codes to do items a to f):

set1 = {1,2,3,4,5}
set2 = {2,4,6,8}
set3 = {1,5,9,13,17}

a. Create a new set of all elements that are in set1 or set2, but not both
b. Create a new set of all elements that are in only one of the three sets set1,set2, and set3.
c. Create a new set of all elements that are exactly two of the sets set1, set2, and set3.
d. Create a new set of all integer elements in the range 1 through 25 that are not in set1.
e. Create a new set of all integer elements in the range 1 to 25 that are not in any of the three sets set1, set2, or set3.
f. Create a new set of integer elements in the range 1 to 25 that are not in all three sets set1, set2, and set3.

Sample Test run (your output should like below)

a. {1, 3, 5, 6, 8}
b. {3, 6, 8, 9, 13, 17}
c. {1, 2, 4, 5}
d. {6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25}
e. {7, 10, 11, 12, 14, 15, 16, 18, 19, 20, 21, 22, 23, 24, 25}
f. {1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25}

continue on next page…

Extra Notes:

Files

Open text files using the open, read/write, close

```
outfile = open("test.txt", "w")
outfile.write("Score")
outfile.close()
```

Opens a text using the *with* statement. Using the *with* statement will automatically closes the file

```
with open("test.txt,"w") as outfile:
        outfile.write("Score")  #need to indent
```

More List stuff

List Methods for modifying a list

- append(item) append item to end of list.
- insert(index, item) insert item into specified index
- remove(item) removes first item in the list that is equal to the specified item. ValueError raised if not found
- index(item) returns the index of the first occurrence of specified item. ValueError raised if not found.
- pop(index) if no index argument is specified, this method gets the last item from the list and removes it. Otherwise, this method gets the item at the specified index and removes it.

The pop() method

```
inventory = ["staff", "hat", "robe", "bread"]
item = inventory.pop()      # item = "bread" since no arguments remove from the end of list
                            # inventory = ["staff", "hat", "robe"]
item = inventory.pop(1)     # item = "hat" remove at index 1
                            # inventory = ["staff", "robe"]
```

The index() and pop() methods

```
inventory = ["staff", "hat", "robe", "bread"]
i = inventory.index("hat")          # 1
inventory.pop(i)                    # ["staff", "robe", "bread"]
```

try and except for trapping errors.

The hierarchy for five common errors

```
Exception
        OSError
                FileExistsError
                FileNotFoundError
        ValueError
```

How to handle a ValueError exception

```
try: number = int(input("Enter an integer: "))
        print("You entered a valid integer of " + str(number) + ".")
except ValueError:
        print("You entered an invalid integer. Please try again.")
print("Thanks!")
```

output with error:

```
Enter an integer: five
You entered an invalid integer. Please try again.
Thanks!
```

Code that handles multiple exceptions (example used with files)

```
filename = input("Enter filename: ")
movies = []
try:
        with open(filename) as file:
                for line in file: line = line.replace("\n", "")
                movies.append(line)
except FileNotFoundError:
        print("Could not find the file named " + filename)
except OSError:
        print("File found - error reading file")
except Exception:
        print("An unexpected error occurred")
```