\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**CSC249 Data structure and algorithms**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**PYTHON COLLECTIONS LAB**

# Objectives

# In this lab, students will learn:

# - How to create and use lists

# - How to create and use tuples

# - How to create and use sets

# - How to create and use dictionaries

# Instruction and Problems

Write a Python program for each of the problems in this lab. Please use PyCharm to type and test your programs. Submit the Python files to Blackboard for credit. In this lab, you should submit 4 Python files, one for each problem.

## Program 1

This program is about lists. A teacher wants a program to give extra points to students who fail a test. Write a Python program to do the following:

1. Ask the user to enter 5 test scores. Store the scores in a list. Display the list.
2. Copy all 5 test scores to another list. Use a loop to examine each test score in the new list. If the score is below 60, add 10 extra points to the score. Display the list.
3. Compare the old score and new score of each student. If the old score and new score are different, display the two scores.

The following is an example.

Enter a test score: 45

Enter a test score: 77

Enter a test score: 88

Enter a test score: 52

Enter a test score: 90

All scores: [45.0, 77.0, 88.0, 52.0, 90.0]

Students who scored below 60 get 10 extra points.

All scores: [55.0, 77.0, 88.0, 62.0, 90.0]

Students whose scores have changed:

Old score: 45.0 New score: 55.0

Old score: 52.0 New score: 62.0

Save your Python program in a file named **LabCollectionsP1.py**. Submit the file to Blackboard for credit.

## Program 2

This program is about tuples. Write a Python program to do the following:

1. Use a for loop and a random integer generator to generate 10 random integers in 1 through 15. Store the random integers in a tuple. Display the tuple. [Hint: you may want to store the random integers in a list first and then convert the list to a tuple]
2. Create a new tuple. Copy the first three elements of the tuple in part (a) to this tuple. Display this tuple.
3. Create a new tuple. Copy the last three elements of the tuple in part (a) to this tuple. Display this tuple.
4. Concatenate the two tuples in part (b) and part (c). Display the concatenated tuple.
5. Sort the concatenated tuple. Display the sorted tuple.

The following is an example. There is no user input in this program.

Tuple of 10 random numbers: (12, 3, 4, 7, 5, 9, 7, 3, 1, 7)

Tuple of first 3 numbers: (12, 3, 4)

Tuple of last 3 numbers: (3, 1, 7)

Two tuples concatenated: (12, 3, 4, 3, 1, 7)

Two tuples concatenated and sorted: (1, 3, 3, 4, 7, 12)

Save your Python program in a file named **LabCollectionsP2.py**. Submit the file to Blackboard for credit.

## Program 3

This program is about sets. Write a Python program to do the following:

(a) Generate 5 random integers between 1 and 10, inclusive. Store the random integers in a set named set1. Display the set. Please note that the set may have less than 5 elements because some of the random integers generated may be redundant.

(b) Generate 5 random integers between 1 and 10, inclusive. Store the random integers in another set named set2. Display the set. Please note that the set may have less than 5 elements because some of the random integers generated may be redundant.

(c) Find and display the union of set1 and set2.

(d) Find and display the intersection of set1 and set2.

(e) Find and display the symmetric difference between set1 and set2.

The following is an example.

set1: {9, 10, 1, 7}

set2: {8, 1, 7}

Union of set1 and set2: {1, 7, 8, 9, 10}

Intersection of set1 and set2: {1, 7}

Symmetric difference between set1 and set2: {8, 9, 10}

Save your Python program in a file named **LabCollectionsP3.py**. Submit the file to Blackboard for credit.

## Program 4

This program is about dictionaries. We want to use a dictionary to store the frequency count of each letter in a string. Write a Python program to do the following:

(a) Ask the user to enter a string. Convert all letters to uppercase. Count the frequency of each letter in the string. Store the frequency counts in a dictionary. You should count letters only. Do not count any other characters such as digits and space. Display the dictionary.

(b) Ask the user to enter a letter. Convert it to uppercase. Check whether the letter is in the dictionary. If it is not, display the message “Letter not in dictionary”. Otherwise, display the frequency count of that letter, remove the letter from the dictionary and display the dictionary after that letter has been removed.

(c) Create a list to store the letters that are in the dictionary. Sort and display the list.

The following is an example.

Enter a string: Magee, Mississippi

Dictionary: {'M': 2, 'A': 1, 'G': 1, 'E': 2, 'I': 4, 'S': 4, 'P': 2}

Choose a letter: s

Frequency count of that letter: 4

Dictionary after that letter removed: {'M': 2, 'A': 1, 'G': 1, 'E': 2, 'I': 4, 'P': 2}

Letters sorted: ['A', 'E', 'G', 'I', 'M', 'P']

Save your Python program in a file named **LabCollectionsP4.py**. Submit the file to Blackboard for credit.

# Grading rubric

Program 1:

Creating score list [5 points]

Modifying score list [10 points]

List scores that have changed [10 points]

Program 2:

Creating tuple of random integers [6 points]

Copying elements of tuple [6 points]

Concatenating tuples [6 points]

Sorting tuple [7 points]

Program 3:

Create sets of random numbers [7 points]

Find union of sets [6 points]

Find intersection of sets [6 points]

Find symmetric difference of sets [6 points]

Program 4:

Create dictionary [5 points]

Check whether letter chosen by user is in dictionary [5 points]

Display frequency count of letter chosen by user [5 points]

Remove letter from dictionary [5 points]

Sort the list of letters in dictionary [5 points]