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

**CSC121 PYTHON Programming**

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

LAB 11 Dictionaries, Sets and Exception Handling

# Objectives

In this lab assignment, students will learn:

- How to create and use dictionaries

- How to create and use sets

- How to write code to handle exceptions

- How to apply exception handling in Python programs

# Goals

In this lab assignment, students will demonstrate the abilities to:

- Create and use dictionaries

- Create and use sets

- Write code to handle exceptions

- Apply exception handling in Python programs

# Instruction and Problems

Write a Python program for each of the problems in this lab. Please use PyCharm to type and test your programs.

## Problem 1

This program is about dictionaries. We want to use a dictionary to store 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 **Lab11P1.py**. Submit the file for credit.

## Problem 2

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) Use set comprehension to select odd numbers from the union and store them in a set. Display this set.

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

(f) Find and display the symmetric difference of 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}

Odd numbers in union of set1 and set2: {1, 9, 7}

Intersection of set1 and set2: {1, 7}

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

Save your Python program in a file named **Lab11P2.py**. Submit the file for credit.

## Problem 3

This program is about exception handling. In Lab 02 we wrote a program for a hotdog stand to calculate total amount due from a customer. We want to add exception handling to the program. The hotdog stand sells hotdogs, potato chips and sodas. Hotdogs are $2.50 each. Potato chips are $1.50 per bag. Sodas are $1.25 per can. The program asks the user to enter number of hotdogs, chips and sodas ordered by the customer. It calculates and displays the total amount due. Numbers of hotdogs, chips and sodas entered should be integers. If the user enters anything that cannot be converted to an integer, display an error message and set the number of that item to 0. For example, if the user enters 3.5 hotdogs, display “Invalid input. Number of hotdogs must be an integer” and set number of hotdogs to 0. Do not ask the user to reenter number of hotdogs.

The following an example:

How many hotdogs? 3.5

Invalid input. Number of hotdogs set to 0.

How many bags of chips? two

Invalid input. Number of chips set to 0.

How many cans of sodas? 2

Please pay this amount: 2.5

Save your Python program in a file named **Lab11P3.py**. Submit the file to Blackboard for credit.

## Problem 4

Rewrite your program in Problem 3. This time if what is entered by the user cannot be converted to an integer, display an error message and ask the user to reenter it until an integer is entered. The following is an example.

How many hotdogs? 3.5

Invalid input.

How many hotdogs? 3

How many bags of chips? two

Invalid input.

How many bags of chips? 2.0

Invalid input.

How many bags of chips? 2

How many cans of soda? 2

Please pay this amount: 13.0

Save your Python program in a file named **Lab11P4.py**. Submit the file to Blackboard for credit.

# Grading rubric for Program 1

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]

# Grading rubric for Program 2

Create sets of random numbers [5 points]

Find union of sets [5 points]

Create set of odd numbers from union [5 points]

Find intersection of sets [5 points]

Find symmetric difference of sets [5 points]

# Grading rubric for Program 3

Exception handling [10 points]

Other statements [10 points]

# Grading rubric for Program 4

Exception handling [10 points]

Asking for input repeatedly until it is valid [10 points]

Other statements [10 points]