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

**CSC249 data strcuture and algorithms**

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

LAB 05 **BAGS**

# Objectives

- Understand the bag collection

- Create a list-based implementation of bag

The textbook shows us array-based and link-based implementations of the bag collection. In this lab, you will create a list-based implementation of bag. In this implementation, a Python list is directly used to store the bag’s items.

First, download the file listbag.py from Blackboard, which has the code of the ListBag class. This class is incomplete. Your task is to add code there to complete the class definition.

The ListBag class only needs one data attribute: self.items, which is a list that stores the data items. Initially, it is empty. If a source collection is provided to the \_\_init\_\_ method, fill the list with the data items from the source collection.

You need to write code for the following methods:

\_\_init\_\_(self, sourceCollection = None): This method creates an empty list, which will be used to store the bag’s items. It fills the list with the data items from the source collection if it is present.

isEmpty(self): This method returns True if the bag is empty, or False otherwise.

\_\_len\_\_(self): This method returns the number of items in the bag.

\_\_str\_\_(self): This method returns the string representation of the bag. Suppose the bag has four items in there: 17, 4, 2, 6. This method returns this string: {17, 4, 2, 6}

\_iter\_\_(self): This method supports iteration over a view of the bag.

\_\_add\_\_(self, other): This method returns a new bag containing the contents of self and other.

\_\_eq\_\_(self, other): This method returns True if self equals other, or False otherwise.

count(self, item): This method returns the number of instances of item in the bag.

clear(self): This method makes the bag become empty.

add(self, item): This method adds item to the bag.

remove(self, item): This method removes item from the bag. It raises KeyError if item is not in the bag.

The main method in the file listbag.py creates a few test cases. The code is already written. Do not change the code in the main method.

Expected output:

The list of items added: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

The bag's size: 10

The bag's string: {1, 2, 3, 4, 5, 6, 7, 8, 9, 10}

The bag is empty: False

4 is in the bag: True

0 is in the bag: False

Add 7 in bag

The bag's string: {1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 7}

Remove 10 from bag

The bag's string: {1, 2, 3, 4, 5, 6, 7, 8, 9, 7}

How many instances of 7? 2

c = ListBag(b)

b == c? True

d = ListBag(22, 4)

b == d? False

e = b + d

bag e's string: {1, 2, 3, 4, 5, 6, 7, 8, 9, 7, 22, 4}

c.clear()

bag c's string: {}

After you have completed the definition of the ListBag class, submit the file listbag.py to Blackboard for credit.

# Grading rubric

The \_\_init\_\_ method [12 points]

The isEmpty method [8 points]

The \_\_len\_\_ method [8 points]

The \_\_str\_\_ method [8 points]

The \_\_iter\_\_ method [8 points]

The \_\_add\_\_ method [10 points]

The \_\_eq\_\_ method [10 points]

The count method [10 points]

The clear method [8 points]

The add method [8 points]

The remove method [10 points]