CS2XB3 Assignment 1 – Implementing Abstract Data Types

Department of Computing and Software
McMaster University
Instructor: Dr. Reza Samavi

January 15th, 2017

Due on: **January 29th , *2017* at 22:00.**

1. Assignment Problem Part 1

1.1 An ADT for a Dictionary

A dictionary or an associative array is a commonly used data structure that is composed of a collection of (key, value) pairs. Each key is **unique** and **may appear only once** within the dictionary.

This assignment requires you to create your own abstract data type for a dictionary. The ADT should contain the following methods:

Insert	Insert a new (key, value) pair.
GetValue	Returns the value of the given key.
GetKey	Returns all keys of the given value.
Remove	Remove the (key, value) pair of the given key.
Compare	Returns true if values of two keys are the same.
Contains	Checks the existence of the given key.
Count	Returns the total number of (key, value) pairs currently in the dictionary.
IsEmpty	Returns whether the Dictionary is empty or not.
printKeys	Prints out all keys in the following format: *{key1, key2,…,keyN}*

The ADT should be called “dictionary” and contain two Arrays. The first array is used to store keys and should be an array of Integers. The second array is used to contain the value and should be an array of Strings. The elements of the same index in the two arrays make a (key, value) pair.

You can assume that the **maximum size of your dictionary is 20 (i.e. the size of the arrays)**. Your task is to create the proper constructors, accessor methods, and instance methods based upon the table above. Proper formatting and commenting is required.

1.2 Implementation:

Before you start implementing this ADT, you should create a project in your workspace in Eclipse called ‘CS2xb3_A1_*lastname_initials’*. Replace *lastname* with your last name and *initials* with your initials. Your entire implementation should be included in this project.

Just like in the case of Lab walk-through #2, you will need to

1. Build an API. What would the API look like? The above table looks similar, but you have to decide on the return type and parameters of each method. You should create a text file named API.txt in the root directory of your Eclipse project and include your API documentation in this file.
2. Implement the abstract data type that you create. What are the required instance variables, accessor methods and instance methods? Make sure encapsulation and reuse are conserved.

1.3 Testing:

Upon completion of the ADT, create a new java class (in the same Eclipse Project) to test the ADT that you have implemented. This class should be called testDictionary. Within this class, you need to create test methods to ensure that the ADT is working properly. Use the following method signatures to create test methods that test each of the methods in your ADT. Each test method must test not only the expected behaviour, but also edge cases.

```
public static void testInsert()
public static void testGetValue()
public static void testGetKey()
public static void testRemove()
public static void testContains()
public static void testCount()
public static void testIsEmpty()
public static void testPrintKeys()
```

The test cases used in the test methods must **read pairs of (key, value) from a text file** called input.txt located in the root directory of your Eclipse project. Each row in your input file should consist of one comma-separated pair of (key, value). Each test case should display appropriate messages at the beginning, during, and at the end of execution. The results of the test methods must be **written to a text file** in the root directory called output.txt. **Read and follow the instructions in Chapter 1 of your Algorithm textbook (pp. 36-41) on how to read/write from/to an external text file.**

Sample Output

```
Entering testInsert…
Test case 1 passed.
Test case 2 passed.
…
testInsert completed.
```

Assignment 1 Submission

In every class that you create, include the following header with the appropriate information:

```
/*            Student Information
 *            ------------------
 *            Student Name: Lastname, Firstname
 *            Student Number: 000000000
 *            Course Code: CS/SE 2XB3
 *            Lab Section: 00
 *
 *            I attest that the following code being submitted is my own
individual work.
 */
```

All relevant documents in this assignment should be included in your Eclipse project. You can only submit the exported Eclipse project as a single zip file to the relevant assignment dropbox in Avenue. You first need to save everything in the project and then export your project as described below:

1. In Eclipse, right-click on the name of the project, select Export->General->Archive File.
2. Ensure that just your project has a check-mark beside it, and select a path and filename to export the project to. Ensure that your export project has a file extension of '.zip'. **IMPORTANT:** You MUST export the FULL Eclipse project. Submitting individual files (e.g. java/class files) will NOT be counted towards your submission. Click 'Finish' to export.
3. Verify the zip file by opening it and ensuring that it has the same folder structure as in Eclipse (it may have some extra files or folder such as 'bin', which is okay).
4. Go to Avenue and upload your zipped project to 'Assignment 1 Submission' Dropbox.

Assignment 1 Marking

Assignment 1 is worth 5% of the course marks. Your grade for this assignment will be determined based on the following rules:

- A submitted assignment that does not compile or run gets 0 credit.
- A solution that runs but is partially correct gets partial credit (depending on the progress towards a full solution).
- A runnable program that generates the correct results but is not efficient (e.g., does not maintain encapsulation, reuse, and other characteristics of objected oriented programming) will score less than an efficient program.
- Providing adequate, concise, and meaningful comments throughout your code is part of the solution grade (i.e., a piece of code that correctly solves a problem without (or with inadequate) comments will score less than a well-commented piece of code that does the same).
- The work you submit must be your own. If you include libraries from any sources other than your own or from the course material (course lecture notes and lab notes/instructions) you must acknowledge them and explicitly give proper credit with meaningful and concise comments inside your code (when using methods from the external libraries). The included libraries should not be a substantial part of your assignment. Copying full or partial codes from other resources and including them inside your code is strictly forbidden. Your work will be checked for plagiarism to account for this. Both copying assignments and allowing others to copy your assignments are strictly forbidden and will be treated as an academic offence.

- Every hour after an assignment deadline 2% will be deducted from the assignment mark. After 48 hours, the student will get 0 credit for the missing assignment. However, you should still submit your assignment as failing to submit an assignment will result in an incomplete grade in this course (you will not receive credit for the course, even if your partial average at the end of the course is above 50).
- If you miss an assignment due date for a legitimate reason (the legitimacy will be determined by the Associate Dean's office through MSAF and supplied documentation and NOT by the course instructor or the TA), it is your responsibility to immediately contact (email) the assigned TA and cc the course instructor to arrange for the submission of the missed work.

VERY IMPORTANT: ONLY THE LAST SUBMISSION WILL BE GRADED. LATE SUBMISSION WILL BE APPLIED BASED ON THE TIMESTAMP OF THE LAST SUBMISSION.