COEN 352 Summer 2020 Assignment 1

This assignment is based on an ADT called ***dictionary***. A dictionary is a collection of records. Each record consists of a unique ***key*** in the whole collection and a ***value*** associated with the key. Each record is formed from a pair of key and value, called key-value pair. The notation of the key-value pair is <key, value>. Hence, the notation of a dictionary is < <k1, v1>, <k2, v2>, … >. The textbook section 4.4 provides the definition of operations on a ***dictionary*** ADT. Please have a read. The source code archive attached with this description document provides the following java source code:

File Name	Content Description
ADTDictionary.java	Dictionary ADT as in Java interface
ALDictionary.java	Java class of dictionary ADT implementation using the array-based list
KVpair.java	Java class of key-value pair
DictionaryJUnitTest.java	JUnit 5 test case of any dictionary ADT implementation. This is optional to your convenience to test.
DictionaryManualTest.java	A Java program to test any dictionary ADT implementation without using JUnit 5. This is optional to your convenience to test.
tf02930030.xltm	Warehouse inventory data samples

The source code package has already provided an example of how the dictionary ADT is implemented using array-based list. The test case Java program using JUnit 5 or without JUnit 5 is also provided to test any implementation. You can choose either DictionaryManualTest.java or DictionaryJUnitTest.java to test your programs.

Problem 1. (10 marks) Given above example Java code, please further program a list-based dictionary implementation.

Problem 2. (6 marks) Use the asymptotic analysis to analyze the (1) average case, and (2) worst case time cost of each method implementation of Problem 1.

Problem 3. (10 marks) Given above example Java code, please further program a double list-based dictionary implementation.

Problem 4. (6 marks) Use the asymptotic analysis to analyze the (1) average case, and (2) worst case time cost of each method implementation of Problem 3. (6 marks) Please fill in the table in the answer sheet.

Problem 5 (40 marks). Define a database called WarehouseInventory. This WarehouseInvestory stores a collection of records. The record's template and sample data are available from

https://templates.office.com/en-us/warehouse-inventory-tm02930030 . It is also include in the attachment. (1) Program an Inventory class that represents the inventory record; (2) Program the WarehouseInventory database using your dictionary program developed either in Problem 1 or Problem 3. The key should be the SKU field in the inventory template. The key-value pair to be stored in the dictionary should be <String, Inventory>; (3) The database is initialized according to the sample file given; (4) Program the following functions of the WarehouseInventory database, and run the program as a Java application or a JUnit 5 test case

- Insert a record
- Remove a record given a key
- Find a record given a key
- Retrieve (return) the whole record given a key
- Retrieve records that have the value "Reordered"
- Return the number of inventories in the database
- Return the total value of all the inventories

Submission Specification

1. When you develop your solution, programs to Problem 1, 3, and 5 should be within the same project, and thus under one ***src*** folder.
2. The src should contain all the Java files
3. Your Problem 5 solution program should have a main function to run your database program as a Java application.
4. The src folder + your answer sheet for Problem 2 and 4 should be archived together as a single file, following the naming convention
   [SID_1]_[SID_2]_A1.zip or
   [SID_1]_[SID_2]_A1.gz or
   [SID_1]_[SID_2]_A1.tar

   No .rar file is accepted. Do not following the naming convention will cause delays in releasing the marking grade.

5. Submission is due on July 17 23:59. Submission is through Moodle site ONLY. Submission in emails is not accepted.

Answer Sheet for Problem 2 and Problem 4.

Asymptotic Analysis for List-based Dictionary

Operation	Worst case time cost	Average case time cost
clear		
find		
insert		
remove		
removeAny		
size		

Asymptotic Analysis for Double List-based Dictionary

Operation	Worst case time cost	Average case time cost
clear		
find		
insert		
remove		
removeAny		
size		