
CSCI 3901 Lab 1: Basic Problem Solving

Friday, 16.09.2021

Team Members

Subash Narayanan [B00899481]

Nadipineni Hemanth Kumar [B00899473]

Items in the list that need clarification

1. What data structure must be used to implement MAP ?.
2. The data type of keys and values that should be used ?.
3. Length of the MAP that we need to work on ?.
4. The functional limitations of the MAP, what Level of functionalities of the MAP must be implemented along with **put()** and **get()** ?.
5. The input is hardcoded or passed by the user through the console, in other words, interactive MAP ?.
6. How to handle Handling wrong input or incorrect data from the user ?.
7. How to present the data to the user and interact and help them handle MAP easily?.

Decisions on the items that need clarification

1. We chose to create a custom **Class** called **Pair** with **KEY & VALUE** variables and we used an Array list of **Pair[class]** objects to implement our MAP.
 2. Since we did not have a clear specific mention of data type requirement for **KEY** and **VALUE** we decided to go with **Int** data type to store **KEY** and **String** datatype to store **VALUE**.
-


3. Since we used an Array list of objects we did not initialize the length of the MAP, so there were no constraints on the fixed length of the MAP.
4. We decide to implement **put(), get(), remove(), display(), size()** functionalities of the MAP concept and restricted them to the above 5 functionalities.
5. Since it's important to test the MAP in real-time, we decided to give a console input and output functionality where the user can display the entire MAP using **display()**, insert data using **put()** key-value pair, **get()** value based on the key, and remove an element using **remove()**.
6. We have conditional checks on the incorrect data and pass out a message acknowledging the incorrect command or the incorrect data type value passed to the MAP.
7. Since we have taken a console-based approach the user can input the key-value pair and display all the elements in it and remove the items easily giving them greater control on the MAP.

How you showed that your work (so far) is working

We have made our MAP interactive with the console, where the user can easily interact with MAP implemented, functionalities that can be accessed through commands of the same name, like **put(), get(), remove(), display(), quit(), size()**. So, the user can add keys and values when prompted and operate and display the current state of the MAP and evaluate accordingly.

What you did well in developing the implementation that you could use as an approach to coding a solution to another problem

We created a custom data type **Pair** [class] for storing key-value pair and created an array list out of it, which helped us to understand the flexibility of the program to implement abstract concepts in the context of MAP, and we are sure this will be the case as we proceed through real-world scenarios where our existing abstract data structures and data types may not be able to handle all the functionalities that



present itself, we will have to create our own custom type to handle data and new functionality accordingly.

We can implement some Abstract concepts like stack and Queue whies functionalities can be mimicked using the same approach, and scope for much more as described in the previous sentences.