CSCI 3901

Software Development Concepts

Faculty of Computer Science

Lab 1: "BASIC PROBLEM SOLVING"

Kishan Kahodariya B00864907

Dhruv Patel B00868931

Objective

Working Independently or in group of two students, write code to implement a Map data Structure in Java. In addition to this, create a constructor along with get() method and put() method.

List of items that needed Clarification

- Data Type of key
- Data Type of value that will be associated with the key
- Can a key have NULL value?
- Is the key which is used to access the value attached with it, Case-sensitive? (i.e. If ”usa” & “USA” are same or not)
- What if user try to retrieve a object which isn’t stored in the data structure?
- If Memory allocation is needed to be considered for efficiency?
- Will the size of Data Structure be Static or User defined?

Decision on items that needed Clarification

- Either we set the default data type for both, Key and Value or we can set Key and value to generic Data Type which can allow making Key-Value pair of different data types
- We can allow key to have NULL value and use it to assign value in future or if in future, user is determined to have ‘no NULL value policy’ then we simply can choose to throw ‘NullPointerException’.
- It’s better to have a code where keys, that are used to retrieve value are Case Sensitive because we don’t know in what context is the key used and what it represents.
- Depending on the data type of Key and Value, memory is allocated. So it is better to assign data type at the start according to user needs as this would help in allocating memory wisely.

- In terms of size of data structure, we decided to keep the size dynamic which means that there's no limit to number of objects, the list will expand accord to user input.

How will the data be stored?

- In terms of storing the data, Array as well as List can be used.
- But list have certain advantage over array in terms of various operations that can be performed to traverse easily across the list. So we have chosen a dynamic List to store the Key-Value pair.

What difficulties did we expect?

- One of the most important difficulty we expected is that whether we should allow the user to choose the data type (i.e. Integer, Character, String) of Key and Value because this would complicate how we traverse the Value from the list for requested Key.
- What if List gets full and user still want to add new object to the list?

How did we know that the code is working?

- We have tested our code for different possible combination for data types of Key-Value pair, by using get() and put() method for the same. Below is the table to highlight our results.
- Please note that the data type of the Key-Value pair was needed to be explicitly defined beforehand along with changed to the get() and put() method for traversal.
- But we also tried figuring out a way where all data types combinations could work in single code. In other words we want our code to allow user to choose Key of any data type without compromising the output of get() and put() method.
- Table of all possible Key-Value Pair Data Type Combination

	Key (Data-type)	Value (Data-type)	Get() (if works?)	Put() (if works?)	Traversal (successful)
1.	String	Integer	Partial	Partial	YES
2.	Integer	String	YES	YES	YES
3.	String	Char	Partial	Partial	YES
4.	Char	String	YES	YES	YES
5.	Char	Integer	YES	YES	YES
6.	Integer	Char	YES	YES	YES
7.	String	String	Partial	Partial	YES
8.	Integer	Integer	YES	YES	YES
9.	Char	Char	YES	YES	YES
10.	Generic	Generic	Not Tested	Not Tested	Not Tested

- One thing we briefly observed is that when we defined Key of String data type, it also allowed data types integer and char to be used as key for the fact that java doesn't differentiate between these when the variable is declared as String.

Analysis

- When we started planning on how to construct the code, we realize that it needs to work for every possible data type for which we started with String and Integer data type for Key-Value pair respectively.
- Initially we planned to use a static array where the size is user-defined for storing the pair but with list, we can operate on the stored data if need arises. So we choose to use List instead.
- Along the way we realized what if the list gets full and user still want to add object and that's when we decided to use a dynamic size list which expands according to user's input.
- As mentioned in the table, we are still trying to figure out a way in which Key-Value pair can be of generic data type, allowing user to create keys of different data type without external input.

Code Implementation that can used as an approach for solving other problem

- Dynamic size approach that we used for deciding the size of the list can be used when the problem deals with issue where user is not sure about the size of his/her data and when he/she wants to add new data in future in the current data structure.
- Our untested approach of using generic data type for Key-Value pair would surely help with problems where the key parameter are of different data types. For example, user have two databases with Key and Value forming the 2 columns and data type of Key & Value of both databases are different from each other. Now if user want to add data of both databases to his Map data structure, then he/she can do so with the help of just one program.