Lab: Hash Map
CSC 207, “Algorithms and Object-Oriented Design”
Department of Computer Science
Grinnell College
March 11, 2021

Goal

The goal of this lab is to implement linear probing, which is used as a technique for accessing or inserting elements in a hash map.

Counting Letters

1. Write a program in a file called CountLetters.java that reports the occurrence of each of the 26 English letters found in /usr/share/dict/words file. The program should display one letter and its count on a separate line, e.g., a: 205807. Note that letters are not case-sensitive. In addition, the program ignores any character that is not an English letter.

Use an array to maintain these counts. How can you map English letters to the indices of this array?

Can the above program be applied to files with arbitrary character sets given the fixed array size that you used in the above exercise?

Counting Characters

Because of the limited array size, the above program does not work for counting the occurrence of arbitrary characters. A technique to address this problem is to determine the array index (hash code) through computing the mod of the character value by the length of the array. That way, every element has a proper index in the array. However, this technique needs to account for collisions that arise from inserting characters that map to the same index.

Write a class called CountChars.java that maps arbitrary characters to their counts. Your class should support the following methods:

- **CountChars ( )**: creates a new empty array. Each array entry stores a key-value Pair.
- **int find (char ch)**: returns the index of an entry in the array. This entry contains a Pair for key ch or is null. This method scans the array until it finds an open spot for placing a Pair or it finds an entry that already contains the Pair.
- **int put (char ch, int v)**: associates key ch with its value v in the map. It overwrites the value for the prior entry with ch, if it exists, and returns the old v. It returns null if there is not a mapping for ch.
- **int get (char ch)**: returns a value for key ch if it exists in the map. This method returns null if there is not a mapping for ch.
- **rehash ( )**: doubles the hash array length if there is not an unused entry in the array for
placing a new key-value Pair.

- getArray( ): returns the array that contains the key-value Pairs.

To resolve collisions, use the linear probing method. Set the initial hash array length to 26. To define the key-value pairs, use the Pair class from slide 12 of the class lecture called Sets and Maps. (Note that the Pair class does not need to be generic.) Implement proper getter and setter methods inside your Pair class. Do not use java.util.Map in your implementations!

Test your program by displaying characters and their counts for /usr/share/dict/words.

Acknowledgement

This lab is based on a material made by Peter-Michael Osera.