Design #3
Author: Alves Silva, Otavio Augusto
Date: 05/16/2015
CS 163 - Program #3

1) How well did the data structure selected perform for the assigned application?

The data structures, which were linear linked list and array(Hash table), selected for this assignment were able to achieve the objective that was assigned with a great results. The LLL in the hash table was capable to work with an unknown upfront memory that the system would need in the future as well as the hash table provided a fast access to the data and an app with more than one keyword doesn't have more than one instance in the system, so the memory isn't being wasted because the nodes of this app are point to the same app. Therefore, the data structures proved a fast away to access the data and with a unknown upfront memory that would be necessary in the future.

2) Would a different data structure work better? If so, which one and why?

Yes, We could change the usage of the LLL by an array. However, the array is a complicated data structure to use with the amount of memory necessary is unknown. The array has direct access the information that would be necessary to do some process whereas a LLL has to traverse through the list to find the specific element. Therefore, an array is just a good choice if you already know the amount of memory that will be used by the program.

3) What was efficient about your design and use of the data structure?

My data structures were capable to achieve all requirements and design to be considered efficient. The hash table with the chaining resolution of collision worked every stable and well. The retrieve information was able to find the applications with a specific keyword even if in that chaining was more than one kind of keyword. It was capable to be replaced by another main program without affecting the client program. The data structure proved a good choice whereas we don't know the amount that will be necessary with a fast access to the data.

4) What was not efficient?

I had to use an auxiliary array of apps to keep the information about the inserted apps. It was necessary to display all the applications. Without it, the away to display all application would be really difficult and hard to manage, because an app could be point by more than one node. Therefore, the display could display repeated times the same app.

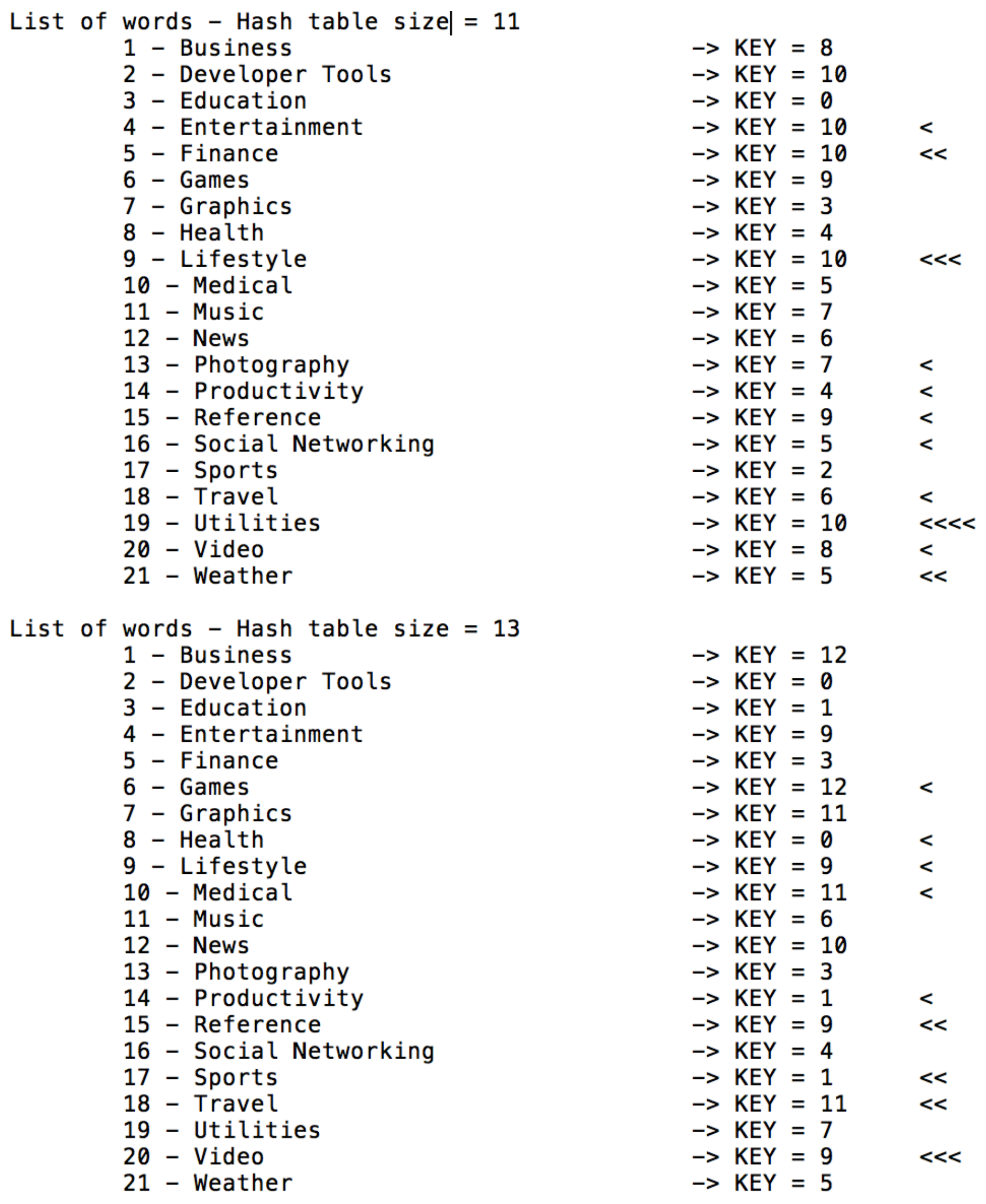
5) What would you do differently if you had more time to solve the problem?

I would change my away to work with the display all in my program. I would try to figure out how I could manage the display without repeated the apps in the terminal. Probably, I would have to use a dynamic allocated array of apps to keep the information of the previous apps without the preoccupation of an amount of the apps in the beginner of the program.

6) The discoveries about the hash table

Considering twenty-one keywords, which were found in the App store, I made the following discoveries.

- The hash function can change totally the shape and the amount of collisions of your hash table.
- If the size of the hash table is small probably we will have more collisions and the chaining of the table will be bigger.
- Improving the size of the hash table by two can do a visually change in the shape and the collisions of the hash table. However, the empty spaces of the hash table can be found more easily. Therefore, the wasting of memory is occurring.
- The following keywords were used for the tests.



```
List of words - Hash table size = 11
        1 - Business                           -> KEY = 8
        2 - Developer Tools                    -> KEY = 10
        3 - Education                          -> KEY = 0
        4 - Entertainment                      -> KEY = 10     <
        5 - Finance                            -> KEY = 10     <<
        6 - Games                              -> KEY = 9
        7 - Graphics                           -> KEY = 3
        8 - Health                             -> KEY = 4
        9 - Lifestyle                          -> KEY = 10     <<<
        10 - Medical                           -> KEY = 5
        11 - Music                             -> KEY = 7
        12 - News                              -> KEY = 6
        13 - Photography                       -> KEY = 7      <
        14 - Productivity                      -> KEY = 4      <
        15 - Reference                         -> KEY = 9      <
        16 - Social Networking                 -> KEY = 5      <
        17 - Sports                            -> KEY = 2
        18 - Travel                            -> KEY = 6      <
        19 - Utilities                         -> KEY = 10     <<<<
        20 - Video                             -> KEY = 8      <
        21 - Weather                           -> KEY = 5      <<

List of words - Hash table size = 13
        1 - Business                           -> KEY = 12
        2 - Developer Tools                    -> KEY = 0
        3 - Education                          -> KEY = 1
        4 - Entertainment                      -> KEY = 9
        5 - Finance                            -> KEY = 3
        6 - Games                              -> KEY = 12     <
        7 - Graphics                           -> KEY = 11
        8 - Health                             -> KEY = 0      <
        9 - Lifestyle                          -> KEY = 9      <
        10 - Medical                           -> KEY = 11     <
        11 - Music                             -> KEY = 6
        12 - News                              -> KEY = 10
        13 - Photography                       -> KEY = 3
        14 - Productivity                      -> KEY = 1      <
        15 - Reference                         -> KEY = 9      <<
        16 - Social Networking                 -> KEY = 4
        17 - Sports                            -> KEY = 1      <<
        18 - Travel                            -> KEY = 11     <<
        19 - Utilities                         -> KEY = 7
        20 - Video                             -> KEY = 9      <<<
        21 - Weather                           -> KEY = 5
```