Kimberly Arenas

Lab 5

Introduction:

Caches are used as instructions that a computer can utilize to manage stored information data on the computer, therefore, improving performance by keeping recent or often used data in particular memory locations. In this lab I had to implement cache “Least Recently Used” (LRU) which discards the least recently used items first and implement a Max heap to print out the most frequent elements in list of words.

Proposed solution and implementation:

The code from the heap class was taken from the resources our professor provided us with. The code had to be modified so that it could read the strings instead of integers. Then a percolate down method had to be included additionally so that it worked for extracting the maximum integer of a list. Thirdly the code had to have some methods that worked with the given text file I had created to test out the heap function and problem. Lastly there was a print method created to not only print the strings in the text file but the number of times that particular string had appeared once manipulated inside the heap.

In order to build the LRU class I had to take a look at the instruction to build it and this included having the following methods constructed: get(), put(), size(), and max\_capaticy(). Before even building the methods mentioned above I had to define class constructors as to how they were needed to build the whole class itself. Then on the get() method is supposed to return the value of an existing key therefore if the key did not exist we must return -1. The method put() as instructed is the most important or center of the cache. Both size() and max\_capacity() methods were declared as constructors and the methods themselves return values that correspond to their algorithm.

Additionally the program commences by asking the user what they would like to use: LRU cache or Heaps! This also serves to differentiate the run times of both classes especially since it was instructed that all the functions run in O(1) time.

Experimental Results:

LRU TEST:

A screenshot of a cell phone

Description automatically generatedA screenshot of a cell phone

Description automatically generatedA close up of text on a black background

Description automatically generated

HEAPS TEST:

A screenshot of a cell phone

Description automatically generatedA picture containing text

Description automatically generatedA picture containing text

Description automatically generated

Above we can notice that LRU testing takes a shorter amount of time to run than heaps take to run. To ensure the testing was accurate I made sure that heaps printed were only 10 such as the LRU testing was limited to only 10 elements. With this we can concluse that both run in O(1) time as instructed.

Conclusion:

It was interesting to create a cache algorithm in a link list similar manner since I had not been the best with that in a previous lab.