Juhwan Lee

CS-163

July 27th 2020

Program #3 Design Write Up

Program #3 is going to be about working with Table Abstraction. A Table ADT allows us to search on the “value” of the data without requiring a particular location to be known. It allows us to store the data using non-linear techniques. There are two non-linear pointer based data structures and those are hash table and binary search tree. In Program #3, we will be using hash table. A hash table comes to mind as a good alternative for programs that require working by “value” but that does not require that the data be sorted. A hash table is a data structure that maps keys to values. This uses a hash function to compute indexes for a key. Based on the hash table index, we can store the value at the appropriate location. The whole benefit of using a hash table is due to it’s very fast access time. Hash tables are very useful in situation where an individual wants to quickly find their data by the “value” or “search key”. The client program uses “key” instead of an “index” to get to the data. The key is then mapped through a hash function which supplies an index. We have index, it is now direct access. However, the hash function may lead to collision that is two or more keys are mapped to same value. Chaining avoids collision. The idea is to make each cell of hash table point to a linear linked list of datas that have same function value. Therefore, the data structure of hash table is going to be an array of linear linked lists. When we try to get information from hash table, in the best case scenario, the runtime performance will be O(1) and in the worst case scenario, the runtime performance will be O(length of linear linked list). So in order to increase the runtime performance, we should reduce the number of chaining, in other words, we should keep every linear linked list short. To do that, we want to create a hash function that creates an index larger than the array size so that the mod operation will help in distributing the data. Using hash table, we will create a program called “CS Comic

Collector” that will help us collect and catalog comics that we are interested in. We will be using an external data files for this program. The information that we want to keep track of includes:

1. The name of the comic (Amazing Spider-Man)
2. The publisher (e.g. DC Comics, Marvel, Dark Horse, etc.)
3. The superhero (e.g. Spider Man)
4. The issue (Issue #2 of the Amazing Spider-Man)
5. A review (from you or someone else)
6. Other items you are interested in

The data file’s name will be “Comic.txt” and the delimiter will be a ‘ | ‘. And the Table ADT are going to have these functions:

1. Add new comic into the table
2. Retrieve information about the comics entered by the superhero name
3. Remove a comic based on the name of the comic book
4. Display all comics featuring a superhero
5. Display all (not ordered)

The data will originally be retrieved from an external data file. To properly evaluate the hash function chosen, we need a large data set. However the program will not be removing or modifying the data from the file. For this program, superhero name will be used as key for the hash function. So all comics with a particular superhero will be inserted into the same chain.