Coding Assignment 6

CSE 3318

Coding Assignment 6 is over creating a hash table and using it to do the 3 basic dictionary actions - insert, delete and search.

Your hash table will be created using an array with separate chaining to resolve conflicts.

Step 3

You are at this step because you have completed Steps 1 and 2 and have had them approved. REMINDER – if you skipped the approval process, any code submitted will not be graded and will automatically be assigned a 0.

You have been provided with my version of Coding Assignment 6 (`Code6_1000074079.c`) and the `Pokedex.txt` input file. Compile and run this program and explore the menu and be familiar with the different functions. You will have some tasks to perform with these functions.

Now, change my version of the C file to reflect your struct and your input file and your hash function (be sure to change the name of the program or you will lose 10 points during grading). You may **NOT** alter your approved data to make it fit closer to mine in order to need fewer coding changes. You need to change the code to fit **YOUR** information. All references to Pokémon and Pokédex should be altered to refer to your information. Any fields/code not needed should be **removed** (not just commented out). A rubric criterion will check for this during grading.

ALL of the Pokémon names in my Pokédex have single names with no spaces which means my code use `scanf()` to read the Pokémon's name. You were required to use a name/hash value that contained spaces. This means that you will need to alter the provided code to use `fgets()` instead of `scanf()`. Be sure to handle the newline.

Test your code and make sure all of the menu functions work correctly for your input data. All functions need to work.

Step 4

Using your version of the code, run the following tests. The tests will be explained using the Pokémon data as examples, but your tests should be done using your version and your data.

Test 1

Set `HASHTABLESIZE` to the number of lines in your input file. For example, `Pokedex.txt` has 37 lines so `HASHTABLESIZE` would be set to 37. Compile and run your program. Use menu option 4 to display your hash table. With `Pokedex.txt`, it would look like this.

```
Pokedex[0]-
Pokedex[1]-
Pokedex[2]-Wartortle|Igglybuff|
Pokedex[3]-
Pokedex[4]-Bulbasaur|
Pokedex[5]-
Pokedex[6]-Pikachu|Squirtle|Ponyta|Venusaur|Nidorina|
Pokedex[7]-
Pokedex[8]-
Pokedex[9]-Blastoise|Eevee|
Pokedex[10]-Golduck|
```

```
Pokedex[11]-Jigglypuff|Nidoqueen|
Pokedex[12]-Raichu|Nidoran|Nidoran|
Pokedex[13]-
Pokedex[14]-Charmander|Finneon|
Pokedex[15]-Ivysaur|
Pokedex[16]-
Pokedex[17]-
Pokedex[18]-Jynx|
Pokedex[19]-Persian|
Pokedex[20]-Nidorino|Nidorino|
Pokedex[21]-
Pokedex[22]-
Pokedex[23]-Charmeleon|
Pokedex[24]-Pichu|Growlithe|Arcanine|
Pokedex[25]-Lumineon|
Pokedex[26]-
Pokedex[27]-
Pokedex[28]-Wigglytuff|Vaporeon|Jolteon|
Pokedex[29]-Smoochum|
Pokedex[30]-
Pokedex[31]-
Pokedex[32]-Charizard|
Pokedex[33]-
Pokedex[34]-
Pokedex[35]-Rapidash|Ditto|
Pokedex[36]-Psyduck|Meowth|
```

Test 1

For this data, we can see that 20 out of 37 cells of the array contain linked lists. That means that 54% of the array is used (20 out of 37 is 54%). We can also see that the longest linked list contains 5 entries. Determine these same values using your version of the program and your input file. Answer Test 1 Question in the "Coding Assignment 6 Results.doc" you were given.

Test 2

Now, double the value of `HASHTABLESIZE` and rerun Test 1. Answer Test 2 Question in the "Coding Assignment 6 Results.doc".

Test 3

Now, set `HASHTABLESIZE` to a value that is half the number of records in the file. For the `Pokedex.txt` file that would mean setting `HASHTABLESIZE` to `19` (37/2=18.5 which I rounded up to 19). Rerun Test 1. Answer Test 3 Question in the "Coding Assignment 6 Results.doc".

Test 4

Set `HASHTABLESIZE` back to its original value (number of lines in the file). Compile the program with the conditional compile TIMING. This will turn on the ability to time how long (number of tics) a lookup takes. Run your program and run 10 different searches and record how many tics each search takes. Find the average and record that in the Test 4 Question in the "Coding Assignment 6 Results.doc".

Test 5

Set `HASHTABLESIZE` to 1. Compile the program with the conditional compile TIMING. This will turn on the ability to time how long (number of tics) a lookup takes. Run your program and run the same 10 searches as Test 4 and record how many tics each search takes. Find the average and record that time and answer Test 5 Question in the "Coding Assignment 6 Results.doc".

Test 6

Repeat Test 5 except only search for the last record of your input file. Do this 10 times and average the results. Record that average time and answer Test 6 Question in the "Coding Assignment 6 Results.doc".

Step 5

Submit the following files in one zip file named `Code6_xxxxxxxxxx.zip` where `xxxxxxxxxx` is your student ID.

`Coding_Assigment_6_Results_xxxxxxxxxx.doc`

`Code6_xxxxxxxxxx.c`

`Code6_xxxxxxxxxx_struct.txt` (must match original submission that was approved)

`Code6_xxxxxxxxxx_InputFile.txt` (must match original submission that was approved)

`Code6_xxxxxxxxxx_HashFunction.c` (must match original submission that was approved)

Approval email