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 integer array with separate chaining to resolve conflicts.

Step 1

For this assignment, you need to choose a subject for your dictionary. For my version of the assignment, I chose Pokémon – you may not also use Pokémon – you need to pick your own subject. Pick a subject that allows you to gather enough information to fulfill the minimum requirements of the assignment.

Structure

You will need to create a `struct` to hold each entry of the dictionary. My Pokémon example is

```
typedef struct Pokemon
{
      char *name;
      float height;
      float weight;
      char gender;
      char *category;
      char *abilities;
      int evolution;
      int national_pokedex_number;
      struct Pokemon *next_ptr;
}
POKEMON;
```

In this `struct`, the `char` pointers are used instead of `char` arrays since it is not known how much space those entries will need. Memory to hold those fields will be dynamically allocated using `malloc()` and the pointer returned by `malloc()` will be stored in the struct.

For your assignment, you will need a `struct` with a minimum of 5 fields – the parts shown in **bold** are required.

```
typedef struct YourStructName
{
      char *chararrayfield;
      float floatfield;
      char charfield;
      int intfield;
      struct YourStructName *next_ptr;
}
YOURSTRUCTNAME;
```

Your `struct` must be `typedef`ed and you will be expect to use the `typedef` throughout the program. You may add more fields if you want your dictionary to hold more information. Your member names must adequately describe the data being stored in the variable. **The field you choose for your hashing must contain a space. Adding a space at the beginning or end is NOT including a space. Your chosen field must include a space somewhere inside it.** The field you choose for hashing must

be the first field listed in your structure. Put your typedef in a text file and name it `Code6_xxxxxxxxxx_struct.txt` where `xxxxxxxxxx` is your student id.

Input File

Your input file must contain a minimum of 30 pipe delimited records that represent your dictionary entries. Here's the first 10 lines of my Pokédex file.

```
Pichu|1.00|4.4|B|Tiny Mouse|Static|1|172
Pikachu|1.04|13.2|B|Mouse|Static|2|025
Raichu|2.07|66.1|B|Mouse|Static|3|026
Charmander|2.00|18.7|B|Lizard|Blaze|1|004
Charmeleon|3.7|41.9|B|Flame|Blaze|2||005
Charizard|5.7|199.5|B|Flame|Blaze|3|006
Squirtle|1.08|19.8|B|Tiny Turtle|Torrent|1|007
Wartortle|3.03|49.6|B|Turtle|Torrent|2|008
Blastoise|5.03|188.5|B|Shellfish|Torrent|3|009
Igglybuff|1.00|2.2|B|Balloon|Cute Charm, Competitive|1|174
```

The fields in each entry must be pipe delimited and must be listed in the same order as your `struct`. Save this as a text file named `Code6_xxxxxxxxxx_InputFile.txt` where `xxxxxxxxxx` is your student id.

Hashing Function

Pick one of the methods discussed in class for creating your hash function (numeric keys, alphanumeric keys, folding). Pick which field(s) from your `struct` will be used for the hash function. Remember that whatever method you choose should result in a value that you can MOD with the hash table define to get an index that will always be within your array.

Write a small C program that prompts for input and calls your hash function and prints the result of the hash. You are proving that your hash function will work. Here's an example – your version may be slightly different depending on your input. THIS CODE IS ONLY FOR SHOWING THAT YOUR HASH FUNCTION WORKS. It is ONLY an example – modify it as needed to show that your hash function will work.

```
#include <stdio.h>
#define HASHTABLESIZE 10
int MyHashFunction(pass in value being used in hash)
{
      perform hash method
      return value % HASHTABLESIZE;
}

int main(void)
{
      char HashValue[20];
      printf("Enter value ");
      scanf("%s", HashValue);
      printf("The hash value for %s is %d\n", HashValue, MyHashFunction(HashValue));
      return 0;
}
```

Compile your version and run and confirm that your function works. Save this code as `Code6_xxxxxxxxxx_HashFunction.c` where `xxxxxxxxxx` is your student id.

Step 2

Submit your input file, `struct` and hashing function for approval. Submit these to me by email. If you want to zip the files, you can, but you can just send them as attachments. No pictures or screenshots will be accepted.

Three files must be submitted via email

`Code6_xxxxxxxxxx_struct.txt`

`Code6_xxxxxxxxxx_InputFile.txt`

`Code6_xxxxxxxxxx_HashFunction.c`

Once you get approval, you will receive the rest of the instructions for the assignment. If you do not get approval, any code you turn in for Coding Assignment 6 will be assigned a grade of 0 regardless.