Homework 3

100 Points

BINARY FILES

Project: Hardware Store Database // See next page

The program passes the input filename to the program as an argv parameter

Grading

1. Use of the argv parameter//file name – 5
2. Create binary file – 20
3. Insert – 20
4. Delete – 20
5. Find and display – 20
6. Error checking – 15

Copy and paste the following data to the input file named **hardware_db.txt**

```
6745,MOLLY BOLT:57
5675,SCREW DRIVER:199
1235,WIDGET:28
2341,WING NUT:89
8624,SLEDGE HAMMER:27
9162,FLASH LIGHT:25
7146,CEMENT BAGS:113
2358,VISE:44
1622,HAMMER:15
1832,THERMOSTAT:78
3271,NAIL:2345
4717,BRACE:234
9524,CLAMP:523
1524,SANDER:99
5219,SAW:211
6275,SAW BLADE:675
5392,BOLT (REGULAR):1311
5192,SCREW DRIVER:789
```

CIS 26B
Advanced C
Programming Assignments

Hardware Store Database

Write a program that that allows additions to, deletions from, or displays of database records in a hardware store database. The records in the database will consists of the following fields:
id – string length is exactly 4 characters (digits only) // key field
name – string length up to 20 characters(letters, spaces, and ( )). This string represents the name of a product that could be found at a hardware store.
qty – an integer. The number represents the quantity available for that product.

Your program must:

a. Create the empty hash file by writing an empty hash table (40 three-struct buckets – no overflow area) into a new file.
b. Do complete error checking for added records. You may assume that names consist only of letters and spaces, that the product id can be any 4-digit combination, and that the quantity range is 0 – 2000.
c. Report and reject record additions that would overflow the 3-record bucket for each hash location.
d. Put names in the file in one case of the other, but not mixed (all upper or all lower, your choice).
e. Abort if the input file is empty.
f. Report every error on the user's input line when s/he tries to add.
g. Use **`input.txt`** as a default input file name. If the user passes the input file's name to the program as an argv parameter, use that name. If this file is not found, use the default file. If the default file is not found, terminate the program. Assume that the input file is valid.

This challenging project involves more than 400 lines of code and uses everything from this chapter. It uses strtok() and other string.h functions.
When adding records, the program will prompt a user for a filename which has records that look like this:

```
1238,WELDING TORCH:18
1327,WALL MOUNT:90
8123,PLANE:67
5934,SOCKET SET:147
9524,ACETYLENE TORCH:10
5349,LEVEL:122
2756,RIVET:89
3495,BOLT (HEX):987
```

You will do batch insertions into the hashed file. Treat overflows in a bucket as an un-resolvable problem but report the error. The hash key is product id and you should sum the cube of the ASCII values of the characters making up the product id before dividing by 40 to find the bucket to put the record. On lookups, the user should enter id, name, and quantity with any amount of whitespace around the three tokens. It is a major task to parse, case, and space the requested input so that it agrees with your format in the hashed file. A valid name should consist of letters, spaces,

and (). You must follow all the rules for good token parsing. Run the program at least twice (one session is to test the lookup errors). Duplicate id not allowed.