Assignment 6

Version 1.0 (last update: Nov. 9, 23:00)

Changes highlighted in yellow

Due date: Tue, Nov 16, 11:59 PM

Summary and Purpose

For this assignment, you will be writing a program that performs a complex data base operation using hash tables, arrays and set. Some example code is provided to help you to understand the data available and how to use it.

Deliverables

You will be submitting:

1) A file called **`a6.c`** that contains a program (including a main function that performs as described below). Do not include any of the functions from the instructor supplied file "hash.c" in your code. #include the file hash.h in your code.
2) A **`makefile`** that complies a6.c into a file a6.o and links the file a6.o with the file hash.o (based on hash.c provided by the instructor), to produce an executable called a6.
3) If you submit any *.rel, *.idx, *.hash, or *.txt files to your git repository you will be deducted 2 marks per file. (Do not add a *.txt entry to your git ignore file or you will not be able the receive feedback on your A6.)

You will submit all of your work via git to the School's gitlab server

This is an individual assignment. Any evidence of code sharing will be investigated and, if appropriate adjudicated using the University's Academic Integrity rules.

Operation of the program

Your program will accept 2 command line parameters. The first will be a building designation that matches one of the building designations in the file building.txt. The second will be a room number that matches one of the room numbers in the file room.txt.

Your program should search the database and find all classes that occur in that room. And use the following print statement to print out the details for each class:

```
printf( “%s*%s %s %s - %s\n”, subject, courseno, days, from, to );
```
Your program should not print anything else. If there are no classes in the room, the program should quit without printing anything.

For the first 80% of the assignment, the print statements are allowed to contain duplicates and need not be in any particular order.

Sample code

In addition to a file hash.c and a file hash.h a number of example programs are provided by the instructor, along with a makefile. The following are some examples of how to use the example program to accomplish the same task that you are required to do (albeit in a very manual and painstaking way).

```
./get_idx building ALEX
```

This will print the index associated with the building Alexander Hall.

```
./get_idx room 200
```

This will print the index associated with room number 200.

```
./query code -1 building 3 building_3.set
```

This will search for all codes that are associated with building index 3 in the code_building.rel relation file. It will store the resulting set of codes in the file building_3.set which is a file consisting of 1 char for each code, set to 0 if the code is NOT associated with the building and set to 1 if the code IS associated with the building.

```
./query code -1 room 49 room_49.set
```

This will search for all codes that are associated with room index 49 in the code_room.rel relation file. It will store the resulting set of codes in the file room_49.set which is a file consisting of 1 char for each code, set to 0 if the code is NOT associated with the room and set to 1 if the code IS associated with the room.

```
./and building_3.set room_49.set alex200.set
```

This will computer the set intersection of the building 3 and room 49 sets and store the result in the set alex200.set.

```
./set2idx alex200.set
```

This will print out the codes that are in the alex200 set.

```
./query code 876 subject -1 876subject.set
```

This will search for all subjects that are associated with code index 876 in the code_subject.rel relation file. It will store the resulting set of subjects in the file 876subject.set which is a file consisting of 1 char for each subject, set to 0 if the subject is NOT associated with the code and set to 1 if the subject IS associated with the code. Because codes are unique, only one subject should be equal to 1.

```
./set2idx 876subject.set
```

This will print out the subject index of the subject contained in the 876subject set.

```
./get_string subject 96
```

This will print out the subject string for index 96.

The last 3 instructions can be repeated in modified form to extract the courseno, days, from and to values.

The last 3 instructions can also be repeated for all other code indices in in the alex200 set.

The Last 20%

The above, constitutes 80% of the assignment. If you complete it, you can get a grade up to 80% (Good). The rest of the assignment is more challenging and will allow you to get a grade of 80-90% (Excellent) or 90-100% (Outstanding). Make sure you complete the first part well, before proceeding to the following additional part.

1. Remove duplicate entries from your output.
2. Convert single entries with multiple weekdays into multiple entries (one for each week day), and sort all entries by start day and time.
3.

You can write additional helper functions as necessary to make sure your code is modular, readable, and easy to modify.

Testing

You are responsible for testing your code to make sure that it works as required. The CourseLink web-site will contain some test programs to get you started. However, we will use a different set of test programs to grade your code, so you need to make sure that your code performs according to the instructions above by writing more test code.

Your assignment will be tested on the standard SoCS Virtualbox VM (http://socs.uoguelph.ca/SoCSVM.zip) which will be run using the Oracle Virtualbox software (https://www.virtualbox.org/wiki/Downloads). If you are developing in a different environment, you will need to allow yourself enough time to test and debug your code on the target machine. We will NOT test your code on YOUR machine/environment.

Full instructions for using the SoCS Virtualbox VM can be found at: https://wiki.socs.uoguelph.ca/students/socsvm.

All compilations and linking must be done with the **`-Wall -pedantic -std=c99`** flags and compile and link **without any warnings or errors**.

Git

You must submit your .c, .h and makefile using git to the School's git server. Only code submitted to the server will be graded. Do ***not*** e-mail your assignment to the instructor. We will only grade one submission; we will only grade the last submission that you make to the server and apply any late penalty based on the last submission. So once your code is complete and you have tested it and you are ready to have it graded make sure to commit and push all of your changes to the server, and then do not make any more changes to the A2 files on the server.

Academic Integrity

Throughout the entire time that you are working on this assignment. You must not look at another student's code, now allow your code to be accessible to any other student. You can share additional test cases (beyond those supplied by the instructor) or discuss what the correct outputs of the test programs should be, but do not share ANY code with your classmates.

Also, do your own work, do not hire someone to do the work for you.

Grading Rubric

a6	12
style	2
makefile	2
no duplicates	2
sorted	2
Total	20

Ask Questions

The instructions above are intended to be as complete and clear as possible. However, it is YOUR responsibility to resolve any ambiguities or confusion about the instructions by asking questions in class, via the discussion forums, or by e-mailing the course e-mail.