COP 3223: C Programming
Spring 2009

Structures In C – Part 3

Instructor : Dr. Mark Llewellyn
markl@cs.ucf.edu
HEC 236, 407-823-2790
http://www.cs.ucf.edu/courses/cop3223/spr2009/section1

School of Electrical Engineering and Computer Science
University of Central Florida

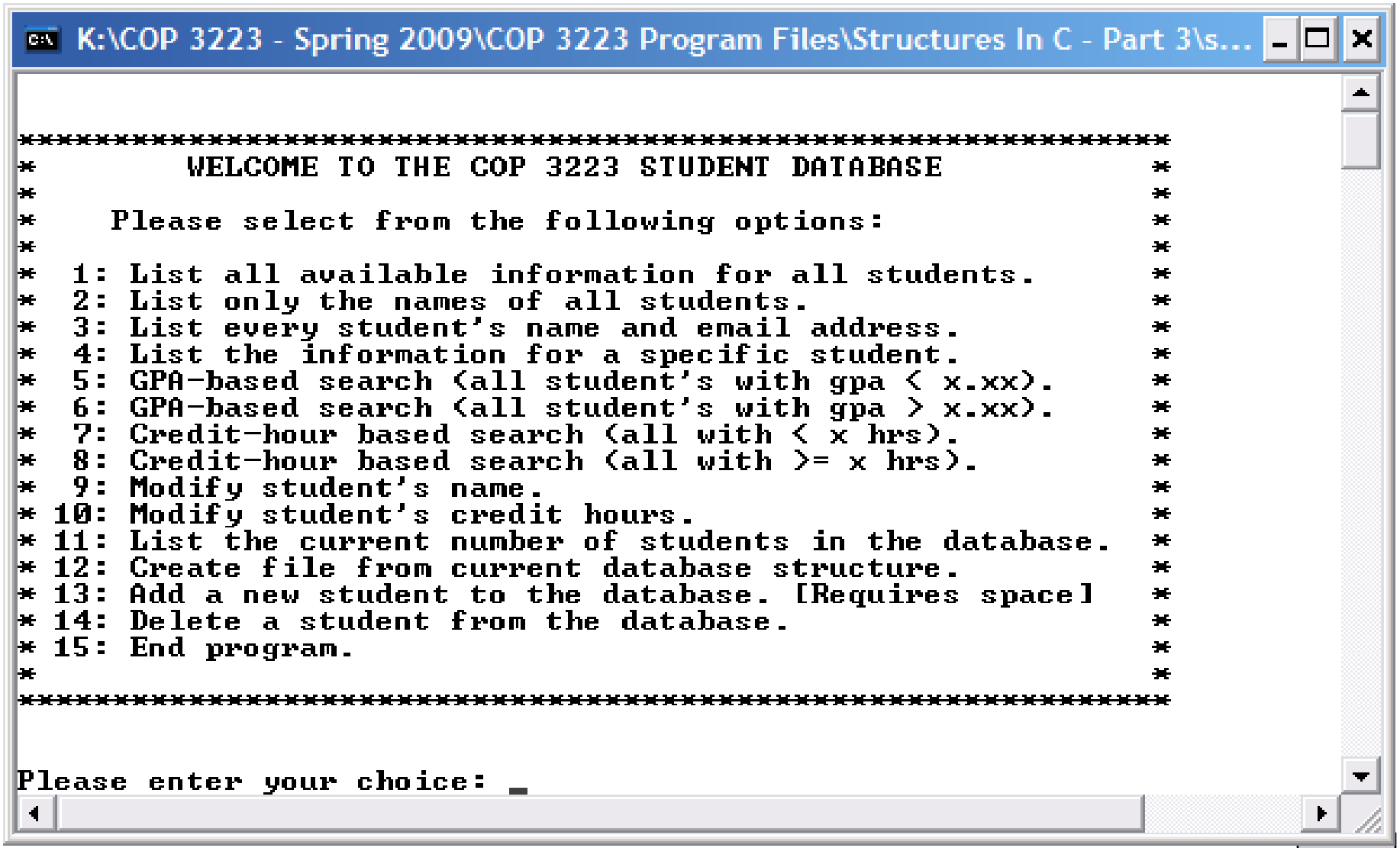
Nested Structures In C

- As a final look at nested structures, I've completed the simple database that we started in the previous set of notes. It includes, file I/O, pointers, functions of all sorts, arrays of nested structures, string handling/manipulation, every C control statement, keyboard input, etc. Virtually everything you've learned about C this semester is included in this example!
- I've tried to make the database a fairly complete example in terms of what the user can do to the data, how the data is read from the file.
- The program is menu-driven offering the user a total of 15 different options of various operations that I thought might be representative of typical operations on a database of this sort.
- There are 10 functions plus the main function in the program. The main function basically consists of a large switch statement that handles the user selection from the menu.

Nested Structures In C

- The program contains a total of 428 lines of code, so I'm not putting it in these notes! The complete program is on the code page for our course so feel free to download it and look at how the program was constructed.
- I've put a few selected sections of the program on the next few pages to illustrate certain points.
- You can certainly use this program as a model to create a simple database for any basic structure (object) that you would like to process. I've even included a user option that will create an output file of the current array of structures, so as the user enters new information and deletes old, the current state can be saved in a output file that has exactly the same structure as the input file, so that next time you run the program the previous state can be loaded from the new file.

The user-option menu



The preprocessor directives and the definition of the structures

The maximum number of students that can be stored in the array. For a larger file, increase this number.

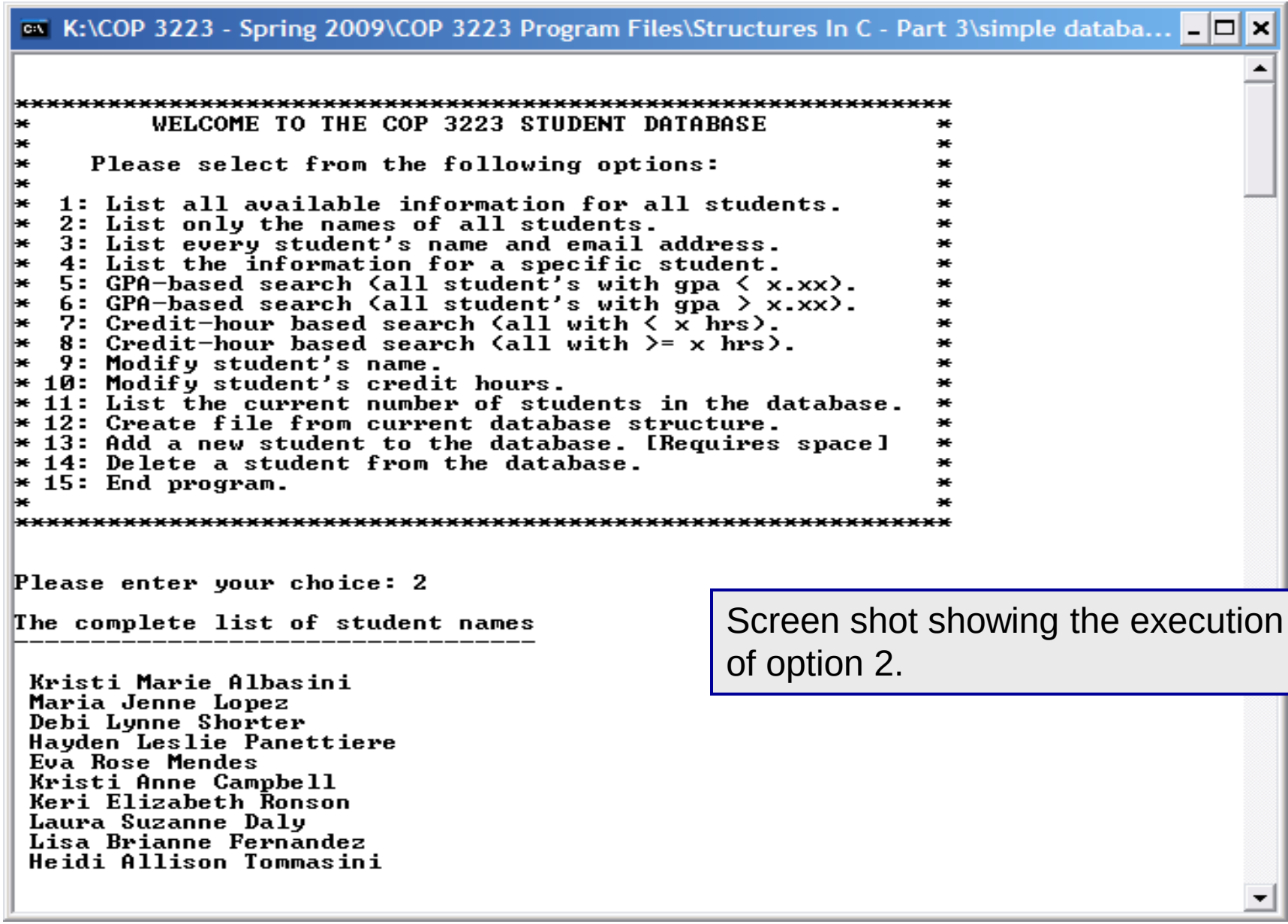
simple database.c

```
2  //This example builds on the program started i
3  //April 17, 2009    Written by: Mark Llewellyn
4
5  #include <stdio.h>
6  #define TRUE 1
7  #define FALSE 0
8  #define MAXLENGTH 15
9  #define EMAILSIZE 30
10 #define MAXNUMBEROFSTUDENTS 50
11 #define FILESTRING 25
12
13 struct personName {
14    char firstName[MAXLENGTH];
15    char middleName[MAXLENGTH];
16    char lastName[MAXLENGTH];
17 };
18
19 struct ucfStudent {
20    struct personName studentName;
21    char email[EMAILSIZE];
22    float gpa;
23    int creditHours;
24    int valid; //indicates if record is valid in the database - used when deleting
25               //TRUE if record is valid, FALSE if it has been deleted via option 14
26 };
27
28 typedef struct ucfStudent Student;
```

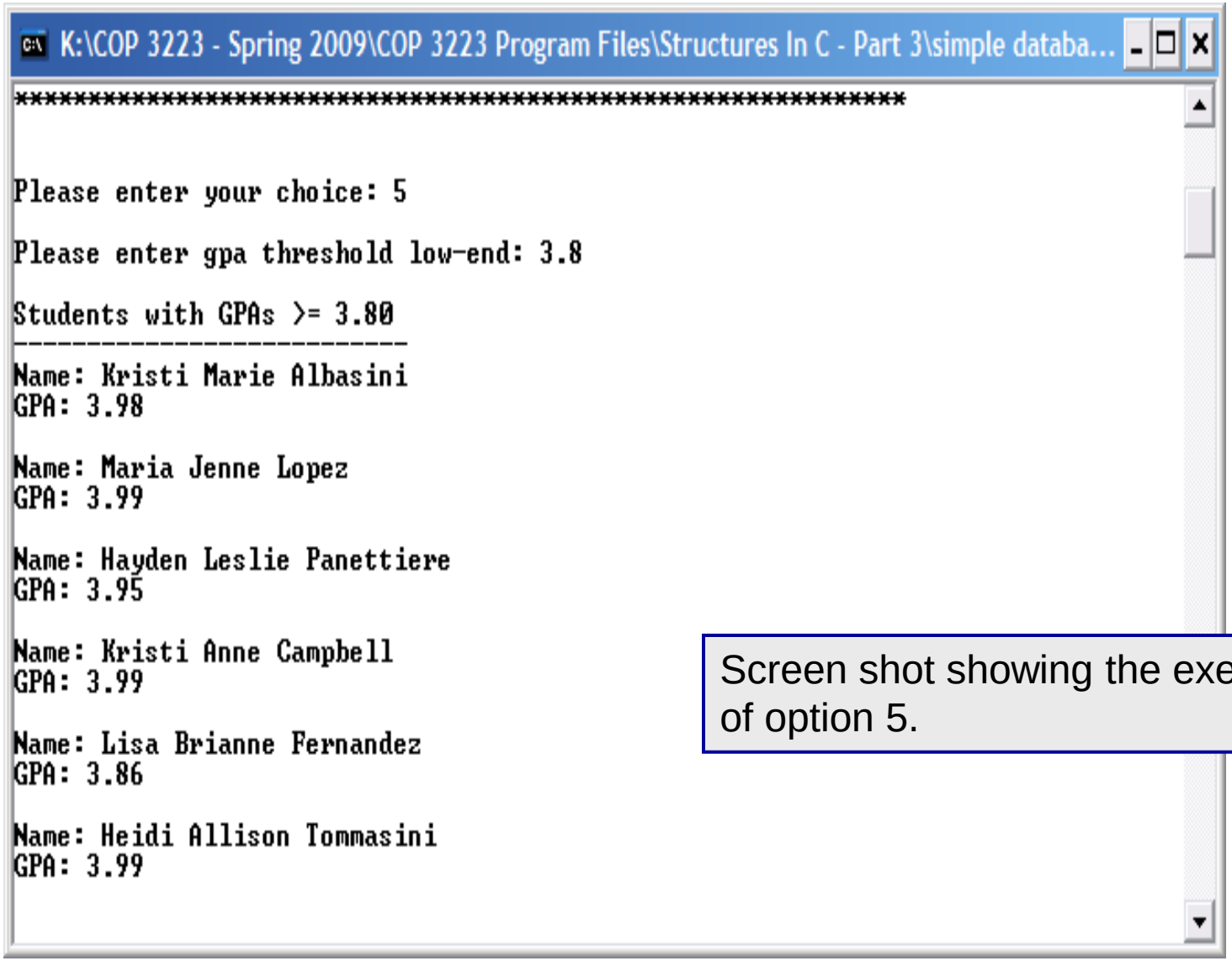

Start of main function – shows variable definitions and initial fill of the database array.

simple database.c

```
221 //*****************************BEGIN MAIN FU
222 int main()
223 {
224     int i; //loop control
225     int index; //array position returned by searches
226     int counter; //counts things as needed
227     FILE *inFilePtr; //input file pointer
228     Student studentDB[MAXNUMBEROFSTUDENTS]; //the database
229     Student newStudent;  //a new student's information
230     int numberOfRecordsInDB; //number of student records in the database
231     int userSelection; //menu item selected by the user
232     char searchName[MAXLENGTH]; //student's last name for searching
233     float lowGPA, highGPA; //gpa thresholds for searches: options 5 & 6
234     int lowCH, highCH; //credit hour thresholds for searches: options 7 & 8
235     char newLastName[MAXLENGTH]; //new student last name for option 9
236     int newCH; //new credit hour value for option 10
237
238     if ( (inFilePtr = fopen("database.txt", "r")) == NULL ) {
239          printf("Sorry, could'nt open input file\n");
240     }
241     else {
242        numberOfRecordsInDB = fillDatabase(studentDB, inFilePtr);
243        drawMenu();
244        printf("Please enter your choice: ");
245        scanf("%d", &userSelection);
246        printf("\n");
247        while (userSelection != 15){
```

Screen shot showing the execution of option 2.



Screen shot showing the execution of option 5.

Practice Problems

1. Add new functionality to the simple database program by adding a new user option to the main menu to perform some function that is not currently an option. Some suggested new features might be: search for all students with the same first name, or same last name; a search based on an email address, i.e., find the student with a specific email address.

2. Modify the simple database by adding more members to the structure, such as the student's major, student's mailing address (i.e., could be a separate structure similar to what we did with the student's name.)