

# COP 3223: C Programming Spring 2009

## Structures In C – Part 2

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

- Nesting one kind of structure inside another structure is a useful technique in many situations. Often hierarchies of structures are developed to more accurately reflect the real-world situation that is being modeled or represented by the application program.
- Consider the following two structure definitions:

```
struct personName {
    char firstName[MAXLENGTH];
    char middleName[MAXLENGTH];
    char lastName[MAXLENGTH];
};
struct ucfStudent {
    struct personName studentName;
    double gpa;
    int creditHours;
}student1;
```



```
4 #include <stdio.h>
5 #define MAXLENGTH 10
6
7 struct personName {
8     char firstName[MAXLENGTH];
9     char middleName[MAXLENGTH];
10    char lastName[MAXLENGTH];
11 };
12
13 struct ucfStudent {
14     struct personName studentName;
15     float gpa;
16     int creditHours;
17 }student1 = {"Kristi", "Marie", "Albasini", 3.98, 115};;
18
19 int main()
20 {
21     struct ucfStudent student2;
22
23     printf("Enter student's first name: ");
24     scanf("%s", &student2.studentName.firstName);
25     printf("\nEnter student's middle name: ");
26     scanf("%s", &student2.studentName.middleName);
27     printf("\nEnter student's last name: ");
28     scanf("%s", &student2.studentName.lastName);
29     printf("\nEnter student's gpa: ");
30     scanf("%f", &student2.gpa);
31     printf("\nEnter student's number of credit hours: ");
32     scanf("%d", &student2.creditHours);
33
```

This version of the program uses normal structure variables with no pointers to the structure. Thus all referencing of the members is done with the dot operator.



```

33
34 printf("\n\n");
35 printf("Information for student 1\n");
36 printf("-----\n");
37 printf("Name: %s %s %s\n", student1.studentName.firstName,
38         student1.studentName.middleName, student1.studentName.lastName);
39 printf("GPA: %4.2f\n", student1.gpa);
40 printf("Credit Hours: %d\n", student1.creditHours);
41 printf("\n\n");
42
43 printf("Information for student 2\n");
44 printf("-----\n");
45 printf("Name: %s %s %s\n", student2.studentName.firstName,
46         student2.studentName.middleName, student2.studentName.lastName);
47 printf("GPA: %4.2f\n", student2.gpa);
48 printf("Credit Hours: %d\n", student2.creditHours);
49
50 printf("\n\n");
51 system("PAUSE");
52 return 0;
53 } //end main function
54

```



```
C:\Courses\COP 3223 - C Programming\Spring 2009\COP 3223 Pro...
Enter student's first name: Maria
Enter student's middle name: Jenne
Enter student's last name: Lopez
Enter student's gpa: 3.99
Enter student's number of credit hours: 97

Information for student 1
-----
Name: Kristi Marie Albasini
GPA: 3.98
Credit Hours: 115

Information for student 2
-----
Name: Maria Jenne Lopez
GPA: 3.99
Credit Hours: 97

Press any key to continue . . . .
```



# Nested Structures In C

- Nesting one kind of structure inside another structure is a useful technique in many situations. Often hierarchies of structures are developed to more accurately reflect the real-world situation that is being modeled or represented by the application program.
- Consider the following two structure definitions:

```
struct personName {
    char firstName[MAXLENGTH];
    char middleName[MAXLENGTH];
    char lastName[MAXLENGTH];
};
struct ucfStudent {
    struct personName studentName;
    double gpa;
    int creditHours;
}student1, *ptrToStudent;
```



# Nested Structures In C

- To access the members of a nested structure requires two applications of the dot operator (or the pointer structure operator if dealing with a pointer to a structure).
- To access the last name of `student1` would require the following expression:

```
student1.studentName.lastName
```

- If we have made the assignment:

```
ptrToStudent = &student1
```

then to access the last name of the student referenced by the pointer `ptrToStudent` would require the following expression:

```
ptrToStudent->studentName.lastName
```



```
1 //Structures In C - Part 2 - nested structures example 2
2 //April 16, 2009      Written by: Mark Llewellyn
3
4 #include <stdio.h>
5 #define MAXLENGTH 10
6
7 struct personName {
8     char firstName[MAXLENGTH];
9     char middleName[MAXLENGTH];
10    char lastName[MAXLENGTH];
11 };
12
13 struct ucfStudent {
14     struct personName studentName;
15     float gpa;
16     int creditHours;
17 } student1 = {"Kristi", "Marie", "Albasini", 3.98, 115};
18
19 int main()
20 {
21     struct ucfStudent student2, student3;
22     struct ucfStudent *ptrToStudent;
23
24     printf("Enter student's first name: ");
25     scanf("%s", &student2.studentName.firstName);
26     printf("\nEnter student's middle name: ");
27     scanf("%s", &student2.studentName.middleName);
28     printf("\nEnter student's last name: ");
29     scanf("%s", &student2.studentName.lastName);
30     printf("\nEnter student's gpa: ");
31     scanf("%f", &student2.gpa);
32     printf("\nEnter student's number of credit hours: ");
```

This version of the program uses both normal structure variables (student1, student2, and student3) and a pointer to the structure (ptrToStudent). Referencing of the members is done with a mix of the dot operator and the structure pointer operator.



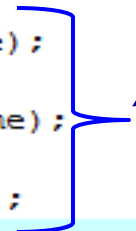


```


33     scanf("%d", &student2.creditHours);
34     printf("\n\n");
35
36     //set ptrToStudent to point to address of student3
37     ptrToStudent = &student3;
38     printf("Enter student's first name: ");
39     scanf("%s", &ptrToStudent->studentName.firstName);
40     printf("\nEnter student's middle name: ");
41     scanf("%s", &ptrToStudent->studentName.middleName);
42     printf("\nEnter student's last name: ");
43     scanf("%s", &ptrToStudent->studentName.lastName);
44     printf("\nEnter student's gpa: ");
45     scanf("%f", &ptrToStudent->gpa);
46     printf("\nEnter student's number of credit hours: ");
47     scanf("%d", &ptrToStudent->creditHours);
48
49     printf("\n\n");
50     printf("Information for student 1\n");
51     printf("-----\n");
52     printf("Name: %s %s %s\n", student1.studentName.firstName,
53           student1.studentName.middleName, student1.studentName.lastName);
54     printf("GPA: %4.2f\n", student1.gpa);
55     printf("Credit Hours: %d\n", student1.creditHours);
56     printf("\n\n");
57
58     printf("Information for student 2\n");
59     printf("-----\n");
60     printf("Name: %s %s %s\n", student2.studentName.firstName,
61           student2.studentName.middleName, student2.studentName.lastName);
62     printf("GPA: %4.2f\n", student2.gpa);
63     printf("Credit Hours: %d\n", student2.creditHours);
64     printf("\n\n");

```

Note mix of referencing operators



```
65
66 printf("Information for student 3\n");
67 printf("-----\n");
68 printf("Name: %s %s %s\n", ptrToStudent->studentName.firstName,
69         ptrToStudent->studentName.middleName, ptrToStudent->studentName.lastName);
70 printf("GPA: %4.2f\n", ptrToStudent->gpa);
71 printf("Credit Hours: %d\n", ptrToStudent->creditHours);
72
73 printf("\n\n");
74 system("PAUSE");
75 return 0;
76 } //end main function
77
```



Note mix of  
referencing  
operators



```
Enter student's first name: Maria
Enter student's middle name: Jenne
Enter student's last name: Lopez
Enter student's gpa: 3.99
Enter student's number of credit hours: 97
```

```
Enter student's first name: Debi
Enter student's middle name: Lynne
Enter student's last name: Shorter
Enter student's gpa: 3.77
Enter student's number of credit hours: 89
```

Information for student 1

```
-----
Name: Kristi Marie Albasini
GPA: 3.98
Credit Hours: 115
```

Information for student 2

```
-----
Name: Maria Jenne Lopez
GPA: 3.99
Credit Hours: 97
```

Information for student 3

```
-----
Name: Debi Lynne Shorter
GPA: 3.77
Credit Hours: 89
```

Press any key to continue . . .



# Nested Structures In C

- In addition to more realistic modeling capabilities provided by nesting structures, another reason to nest structures is to treat some data as units of data to simplify parameter passing to functions.
- For example, if we wanted to represent a student's name as consisting of a first, middle, and last name and we did not have the person name structure, we would have needed to include 2 additional members in the `ucfStudent` structure. Thus, it would resemble the following:

```
struct ucfStudent {
    char firstName[MAXLENGTH];
    char middleName[MAXLENGTH];
    char lastName[MAXLENGTH];
    double gpa;
    int creditHours;
}student1;
```



# Nested Structures In C

- If we wanted to construct a function that just printed the name members for a given student, the function would require three parameters.

The function prototype might look like:

```
void printName (struct ucfStudent . fName [MAX] ,  
                struct ucfStudent . mName [MAX] ,  
                struct ucfStudent . lName [MAX] ) ;
```

The function call might look like:

```
printName (student1 . firstName ,  
           student1 . middleName ,  
           student1 . lastName) ;
```



# Nested Structures In C

- Whereas, with our nested structure definition, the function would require just a single parameter and might look like the following:

The function prototype might look like:

```
void printName(struct personName name);
```

The function call might look like:

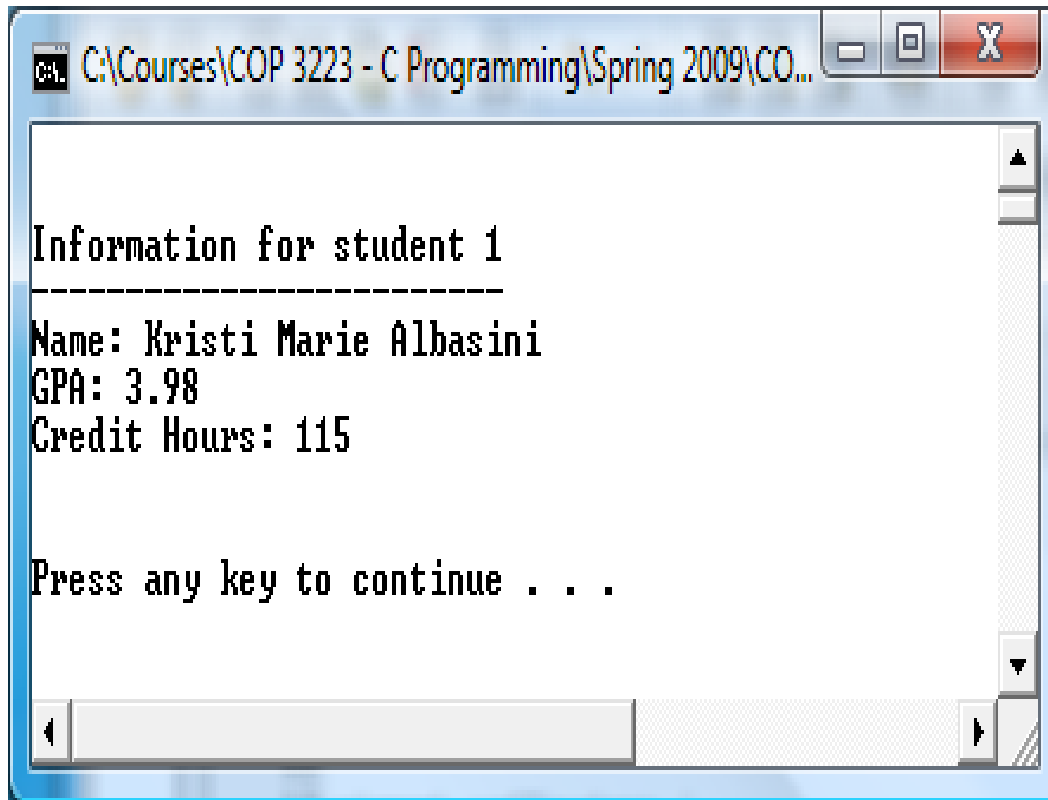
```
printName(student1.studentName);
```

- The small sample program on the next page illustrates this technique.



```
2 //April 16, 2009    Written by: Mark Llewellyn
3
4 #include <stdio.h>
5 #define MAXLENGTH 10
6
7 struct personName {
8     char firstName[MAXLENGTH];
9     char middleName[MAXLENGTH];
10    char lastName[MAXLENGTH];
11 };
12
13 struct ucfStudent {
14     struct personName studentName;
15     float gpa;
16     int creditHours;
17 } student1 = {"Kristi", "Marie", "Albasini", 3.98, 115};
18
19 void printName(struct personName name)
20 {
21     printf("Name: %s %s %s\n", name.firstName, name.middleName, name.lastName);
22     return;
23 } //end printName function
24
25 int main()
26 {
27     printf("\n\n");
28     printf("Information for student 1\n");
29     printf("-----\n");
30     printName(student1.studentName);
31     printf("GPA: %4.2f\n", student1.gpa);
32     printf("Credit Hours: %d\n", student1.creditHours);
33
```





C:\Courses\COP 3223 - C Programming\Spring 2009\CO...

```
Information for student 1
-----
Name: Kristi Marie Albasini
GPA: 3.98
Credit Hours: 115

Press any key to continue . . .
```





# Arrays Of Structures

- An extremely common and useful technique is to construct arrays of structures. That is to say, the elements of the array are structures rather than primitive data types like integers or characters.
- Arrays of structures can be used to simulate a simple database.
- For example, if we wanted to construct a database containing information about students at UCF using our `ucfStudent` structure we could simply use an array containing 50,000 of these structures such as:

```
struct ucfStudent studentDB[50000];
```

- Let's build such a structure, but let's assume there are only 10 students instead of 50000+!



```
1 //Structures In C - Part 2 - array of nested structures - a database example
2 //An array of 10 structures each storing information about students is built and
3 //accessed with various functions
4 //April 16, 2009    Written by: Mark Llewellyn
5
6 #include <stdio.h>
7 #define MAXLENGTH 15
8 #define EMAILSIZE 30
9 #define NUMBEROFSTUDENTS 10
10
11 struct personName {
12     char firstName[MAXLENGTH];
13     char middleName[MAXLENGTH];
14     char lastName[MAXLENGTH];
15 };
16
17 struct ucfStudent {
18     struct personName studentName;
19     char email[EMAILSIZE];
20     float gpa;
21     int creditHours;
22 };
23
24 typedef struct ucfStudent Student;
25
26
```



```
27 void printStudentInfo(Student localStudent)
28 {
29     printf("Name: %s %s %s\n", localStudent.studentName.firstName,
30           localStudent.studentName.middleName,
31           localStudent.studentName.lastName);
32     printf("GPA: %4.2f\n", localStudent.gpa);
33     printf("Credit Hours: %d\n", localStudent.creditHours);
34     printf("Email: %s\n\n", localStudent.email);
35     return;
36 } //end printStudentInfo function
37
38 int main()
39 {
40     int i; //loop control
41     Student studentDB[NUMBEROFSTUDENTS] =
42     {
43         {"Kristi", "Marie", "Albasini", "kma@phycics.ucf.edu", 3.98, 115},
44         {"Maria", "Jenne", "Lopez", "maria@gmail.com", 3.99, 97},
45         {"Debi", "Lynne", "Shorter", "shorterdeb@harris.com", 3.77, 89},
46         {"Hayden", "Leslie", "Panettiere", "savethecheerleader.com", 3.95, 24},
47         {"Eva", "Rose", "Mendes", "lagirl@google.com", 3.78, 110},
48         {"Kristi", "Anne", "Campbell", "brazilian@gmail.com", 3.99, 45},
49         {"Keri", "Elizabeth", "Ronson", "imkeri@yahoo.com", 2.99, 20},
50         {"Laura", "Suzanne", "Daly", "laurab@yahoo.com", 3.65, 44},
51         {"Lisa", "Brianna", "Fernandez", "lbf@gmail.com", 3.86, 65},
52         {"Heidi", "Allison", "Tommasini", "heidi@cs.unlv.edu", 3.99, 118}
53     };
--
```



Information for student 0

Name: Kristi Marie Albasini  
GPA: 3.98  
Credit Hours: 115  
Email: kma@phycics.ucf.edu

Information for student 1

Name: Maria Jenne Lopez  
GPA: 3.99  
Credit Hours: 97  
Email: maria@gmail.com

Information for student 2

Name: Debi Lynne Shorter  
GPA: 3.77  
Credit Hours: 89  
Email: shorterdeb@harris.com

Information for student 3

Name: Hayden Leslie Panettiere  
GPA: 3.95  
Credit Hours: 24  
Email: savethecheerleader.com

Information for student 4

Name: Eva Rose Mendes  
GPA: 3.78  
Credit Hours: 110  
Email: lagirl@google.com

Information for student 5

Name: Kristi Anne Campbell  
GPA: 3.99  
Credit Hours: 45  
Email: brazilian@gmail.com

Information for student 6

Name: Keri Elizabeth Ronson  
GPA: 2.99  
Credit Hours: 20  
Email: imkeri@yahoo.com

Information for student 7

Name: Laura Suzanne Daly  
GPA: 3.65  
Credit Hours: 44  
Email: laurab@yahoo.com

Information for student 8

Name: Lisa Brianne Fernandez  
GPA: 3.86  
Credit Hours: 65  
Email: lbf@gmail.com

Information for student 9

Name: Heidi Allison Tommasini  
GPA: 3.99  
Credit Hours: 118  
Email: heidi@cs.unlv.edu

Press any key to continue . . . .



# Practice Problems

1. Rewrite the program on page 3 so that only pointers to structures, including the nested structure, are used to access the members of the structure.
2. Rewrite the student database program that begins on page 14 so that the data to be entered into the array (i.e., the database) is read from a file rather than via the initializers.

