Rajarata University of Sri Lanka

Faculty of Applied Sciences

COM 1407 – Computer Programming

Practical – Structures in C

Outline

- Define, Initialize and Access to the C Structures
- Develop programs using Structures
- Structures within structures and structures as pointers

1 Defining a Structure

- A **structure** is another user defined data type available in C that allows to combine data items of different kinds.
- The difference between Arrays and structures is, **Arrays** can only hold same kind (data type) of data items but **Structure** can hold different type of data items.
- Structures are used to represent a record.
- To define a structure, you must use the **struct** statement. The **struct** statement defines a new data type, with more than one member.
- **Syntax**

```
struct [structure tag] {

   member definition;
   member definition;
   ...
   member definition;
} [one or more structure variables];
```

- Each member definition is a normal variable definition, such as int i; or float f; or any other valid variable definition.
- At the end of the structure's definition, before the final semicolon, you can specify one or more structure variables but it is optional.
- Consider following example.

```
struct Books {
   char  title[50];
   char  author[50];
   char  subject[100];
   int   book_id;
} book;
```

2 Accessing Structure Members

- To access any member of a structure, **we use the member access operator (.)**.
- The member access operator is coded between the structure variable name and the structure member that we wish to access.
- Consider following example

```c
#include <stdio.h>
#include <string.h>

struct Books {
   char  title[50];
   char  author[50];
   char  subject[100];
   int   book_id;
};

int main( ) {

   struct Books Book1;
   struct Books Book2;


   strcpy( Book1.title, "C Programming");
   strcpy( Book1.author, "Nuha Ali");
   strcpy( Book1.subject, "C Programming Tutorial");
   Book1.book_id = 6495407;


   strcpy( Book2.title, "Telecom Billing");
   strcpy( Book2.author, "Zara Ali");
   strcpy( Book2.subject, "Telecom Billing Tutorial");
   Book2.book_id = 6495700;

   printf( "Book 1 title : %s\n", Book1.title);
   printf( "Book 1 author : %s\n", Book1.author);
   printf( "Book 1 subject : %s\n", Book1.subject);
   printf( "Book 1 book_id : %d\n", Book1.book_id);


   printf( "Book 2 title : %s\n", Book2.title);
   printf( "Book 2 author : %s\n", Book2.author);
   printf( "Book 2 subject : %s\n", Book2.subject);
   printf( "Book 2 book_id : %d\n", Book2.book_id);

   return 0;
}
```

3 Array of Structures

- As you know, C Structure is collection of different data types (variables) which are grouped together. Whereas, array of structures is collection of structures. This is also called as structure array in C.
- **Syntax**
  `struct student record[2];`
- Consider following example.

```
#include <stdio.h>
#include <string.h>

struct student
{
     int id;
     char name[30];
     float percentage;
};

int main()
{
     int i;
     struct student record[2];


     record[0].id=1;
     strcpy(record[0].name, "Raju");
     record[0].percentage = 86.5;


     record[1].id=2;
     strcpy(record[1].name, "Surendren");
     record[1].percentage = 90.5;


     for(i=0; i<2; i++)
     {
         printf("Records of STUDENT : %d \n", i+1);
         printf("Id is: %d \n", record[i].id);
         printf("Name is: %s \n", record[i].name);
         printf("Percentage is: %f\n\n",record[i].percentage);
     }
     return 0;
}
```

4 C structures using typedef

- typedef is a keyword used in C language to assign alternative names to existing data types. It is mostly used with user defined data types, when names of the data types become slightly complicated to use in programs.
- You can use typedef with structure to define a new data type and then use that data type to define structure variables directly.
- Using **typedef** avoids having to write *struct* every time you declare a variable of that type.

```
#include <stdio.h>
#include <string.h>

typedef struct Books {
   char title[50];
   char author[50];
   char subject[100];
   int book_id;
} Book;

int main( ) {

   Book book;

   strcpy( book.title, "C Programming");
   strcpy( book.author, "Nuha Ali");
   strcpy( book.subject, "C Programming Tutorial");
   book.book_id = 6495407;

   printf( "Book title : %s\n", book.title);
   printf( "Book author : %s\n", book.author);
   printf( "Book subject : %s\n", book.subject);
   printf( "Book book_id : %d\n", book.book_id);

   return 0;}
```

5 Structures as Function Arguments

- You can pass a structure as a function argument in the same way as you pass any other variable.
- Consider following example.

```
#include <stdio.h>
#include <string.h>

struct Books {
   char  title[50];
   char  author[50];
   char  subject[100];
   int   book_id;
};

void printBook( struct Books book );

int main( ) {

   struct Books Book1;
   struct Books Book2;

   strcpy( Book1.title, "C Programming");
   strcpy( Book1.author, "Nuha Ali");
   strcpy( Book1.subject, "C Programming Tutorial");
   Book1.book_id = 6495407;

   strcpy( Book2.title, "Telecom Billing");
   strcpy( Book2.author, "Zara Ali");
   strcpy( Book2.subject, "Telecom Billing Tutorial");
   Book2.book_id = 6495700;

   printBook( Book1 );
   printBook( Book2 );

   return 0;
}

void printBook( struct Books book ) {

   printf( "Book title : %s\n", book.title);
   printf( "Book author : %s\n", book.author);
   printf( "Book subject : %s\n", book.subject);
   printf( "Book book_id : %d\n", book.book_id);
}
```

6 Pointers to Structures

- You can define pointers to structures in the same way as you define pointer to any other variable.
- **Syntax**

```
struct Books *struct_pointer;
```

- You can store the address of a structure variable in the above defined pointer variable. To find the address of a structure variable, place the '&'; operator before the structure's name.
- To access the members of a structure using a pointer to that structure, you must use the → operator
- Consider following example.

```
#include <stdio.h>
#include <string.h>

struct Books {
   char  title[50];
   char  author[50];
   char  subject[100];
   int   book_id;
};

void printBook( struct Books *book );
int main( ) {

   struct Books Book1;
   struct Books Book2;

   strcpy( Book1.title, "C Programming");
   strcpy( Book1.author, "Nuha Ali");
   strcpy( Book1.subject, "C Programming Tutorial");
   Book1.book_id = 6495407;

   strcpy( Book2.title, "Telecom Billing");
   strcpy( Book2.author, "Zara Ali");
   strcpy( Book2.subject, "Telecom Billing Tutorial");
   Book2.book_id = 6495700;

   printBook( &Book1 );
   printBook( &Book2 );

   return 0;
}
```

```
void printBook( struct Books *book ) {

   printf( "Book title : %s\n", book->title);
   printf( "Book author : %s\n", book->author);
   printf( "Book subject : %s\n", book->subject);
   printf( "Book book_id : %d\n", book->book_id);
}
```

7 Nested Structures

- Nested structure in C is structure within structure. One structure can be declared inside other structure as we declare structure members inside a structure.

```
#include <stdio.h>
structParticipant
{
char Name[20];
char Country[25];
float Score;s
intAge;
};
structTeam
{structParticipant captain;
intwins;
intlosses;
structParticipant member[2];
};
intmain()
{
structTeam slTeam= {{"Aravinda","LKR", 8.8,25},10,2,{
{"Kumara","LKR", 8.0,25},{"Dilip","LKR", 7.8,25}}
};
printf("Captain : %s\n",slTeam.captain.Name);
printf("Team Members\n");
printf("%s\n", slTeam.member[0].Name);
printf("%s\n", slTeam.member[1].Name);
printf("Match Score\nWins: %i\nLosses:%i\n",
slTeam.wins, slTeam.losses);
return 0;
}
```

8 Exercises

- Write a program to define a structure to store Students Name, ID and Average marks and print them. (user inputs above data).
- Define a structure to store Name, Feet and Arms. Fill the fields with data and print them out as follows (using typedef)
    - ✓ A human has 2 legs and 2 arms.
    - ✓ A dog has 4 legs and 0 arms.
    - ✓ A television stand has 4 legs and 0 arms.
    - ✓ A chair has 4 legs and 2 arms.
- Define a structure to store Name, Employee number and Salary of 5 Employees using an array. Get the input from user and print them.