**CS118**

**Programming Fundamentals**

LAB 10

## STRUCTURES in C

**NATIONAL UNIVERSITY OF COMPUTER AND EMERGING SCIENCES**

# C Structures

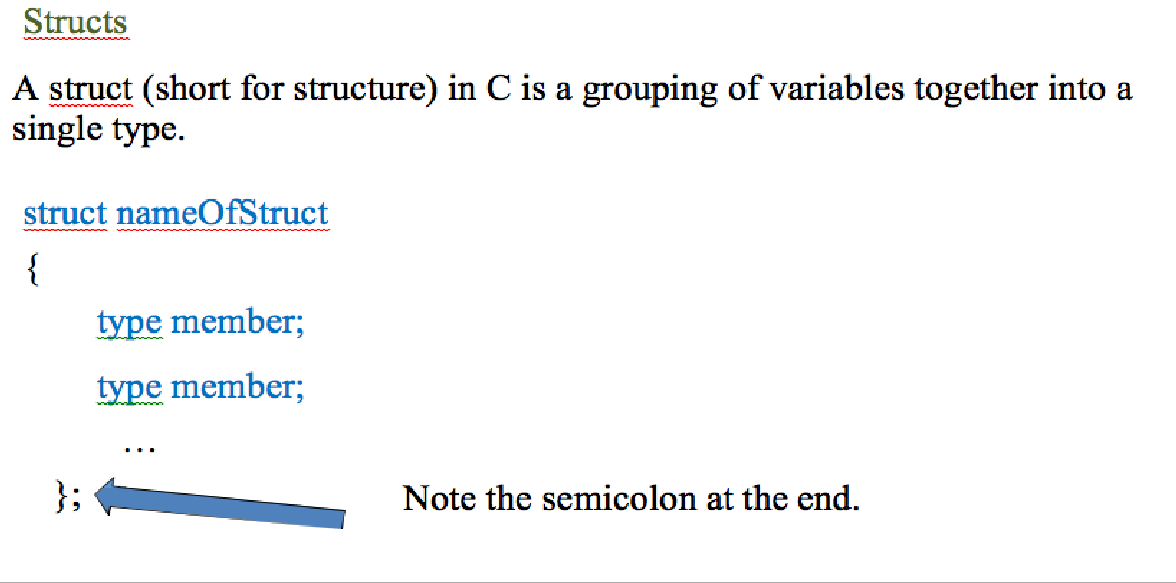
C arrays allow you to define type of variables that can hold several data items of the same kind but structure is another user defined data type available in C programming, which allows you to combine data items of different kinds.

Structure is the collection of heterogeneous data items unlike an array. Structure can also be defined as a collection of a fixed number of components in which the components are accessed by name. The components may be of different types. All the data items in a structure may or may not be of same data type. There are some data items, which are group of multiple values instead of a single value. In short, Structures are used to represent a record.

Example: Suppose you want to keep track of your books in a library. You might want to track the following attributes about each book:

* Title,
* Author,
* Subject
* Book ID

# Defining a Structure



struct Books

{

char title[50]; char author[50]; char subject[100]; int book\_id;

}b1,b2,phy;

**Struct Books b1,b2,phy;**

**Declaring a Structure**

To declare a Structure variable:

Syntax: struct nameOfStruct variable\_name; Example: struct Books Book1;

## Another Example:

struct Person {

char name[50];

int citNo;

float salary;

};

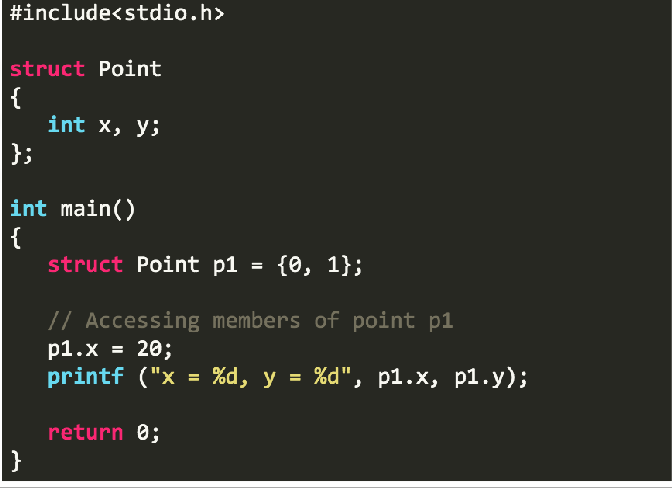
int main() {

struct Person **person1, person2, p[20]**; return 0;

}

**Accessing a Structure:**

Once structure is defined and its variable is created, we can access the individual members of the structure with the help of dot operator ( . ). The dot operator is also called as member access operator.



## Another Example:

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

struct Books

{

char char char

int

title[50]; author[50]; subject[100];

book\_id;

} Book1, Book2;

int main( )

{

/\* book 1 specification \*/

strcpy( Book1.title, "CProgramming"); strcpy( Book1.author, "Nuha Ali");

strcpy( Book1.subject, "C ProgrammingTutorial"); Book1.book\_id = 6495407;

/\* book 2 specification \*/

strcpy( Book2.title, "TelecomBilling"); strcpy( Book2.author, "Zara Ali");

strcpy( Book2.subject, "Telecom Billing Tutorial");

Book2.book\_id = 6495700;

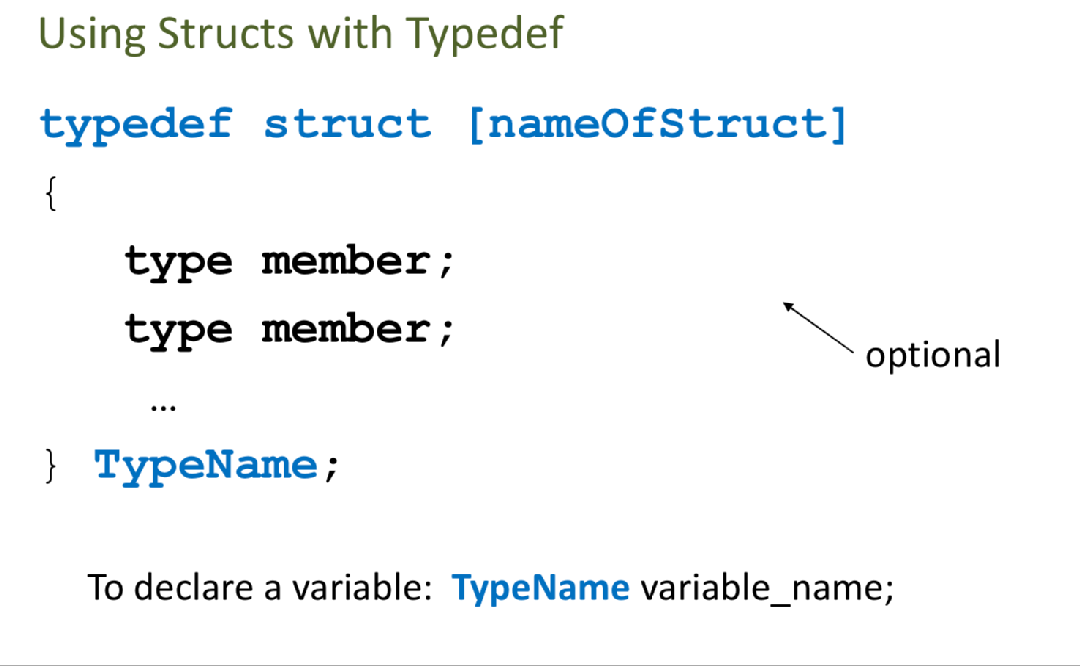
/\* print Book1 info \*/

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

**Keyword typedef**

We use the keyword to create an alias name for data types. It is commonly used with structures to simplify the syntax of declaring variables.

typedef



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);

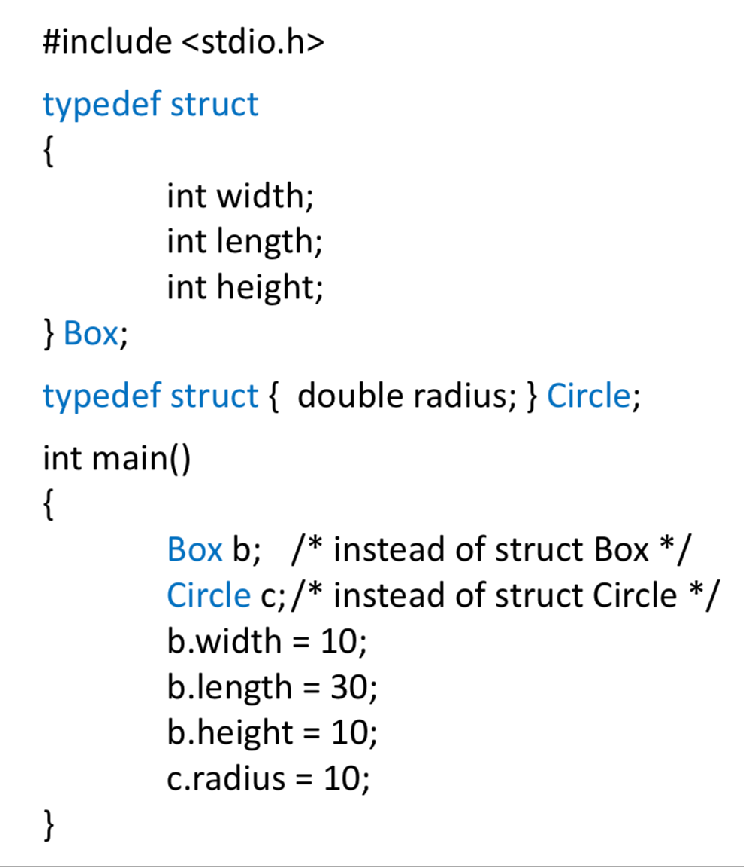
/\* print Book2 info \*/

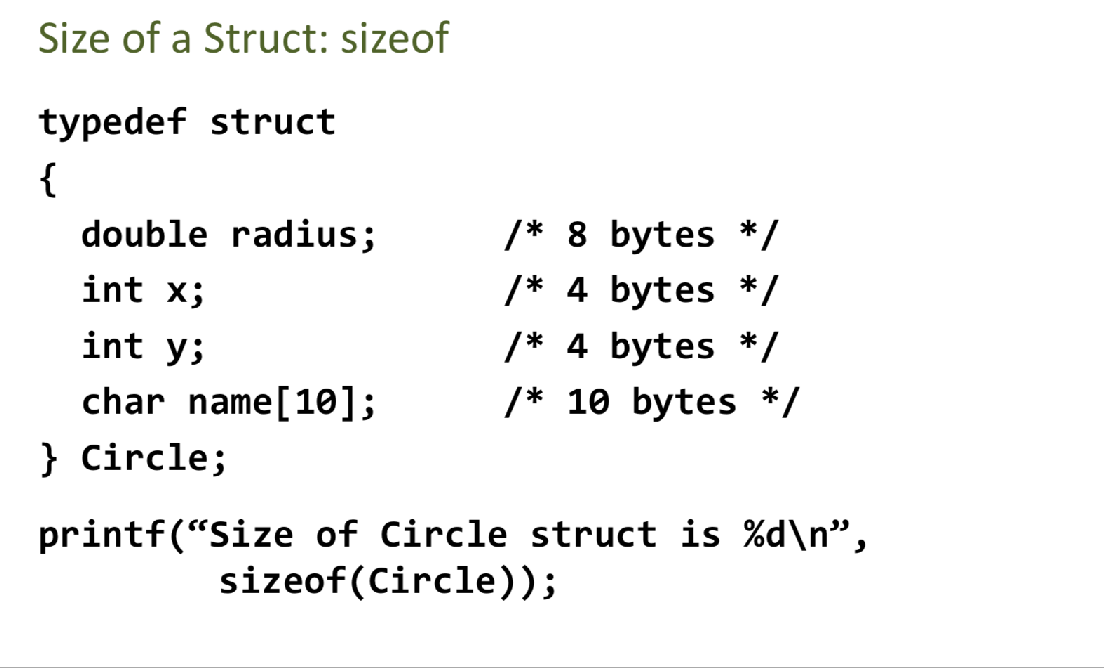
printf( "Book 2 title : %s\n", Book2.title); printf( "Book 2 author : %s\n", Book2.author); printf( "Book 2 subject : %s\n", Book2.subject);

printf( "Book 2 book\_id : %d\n", Book2.book\_id);

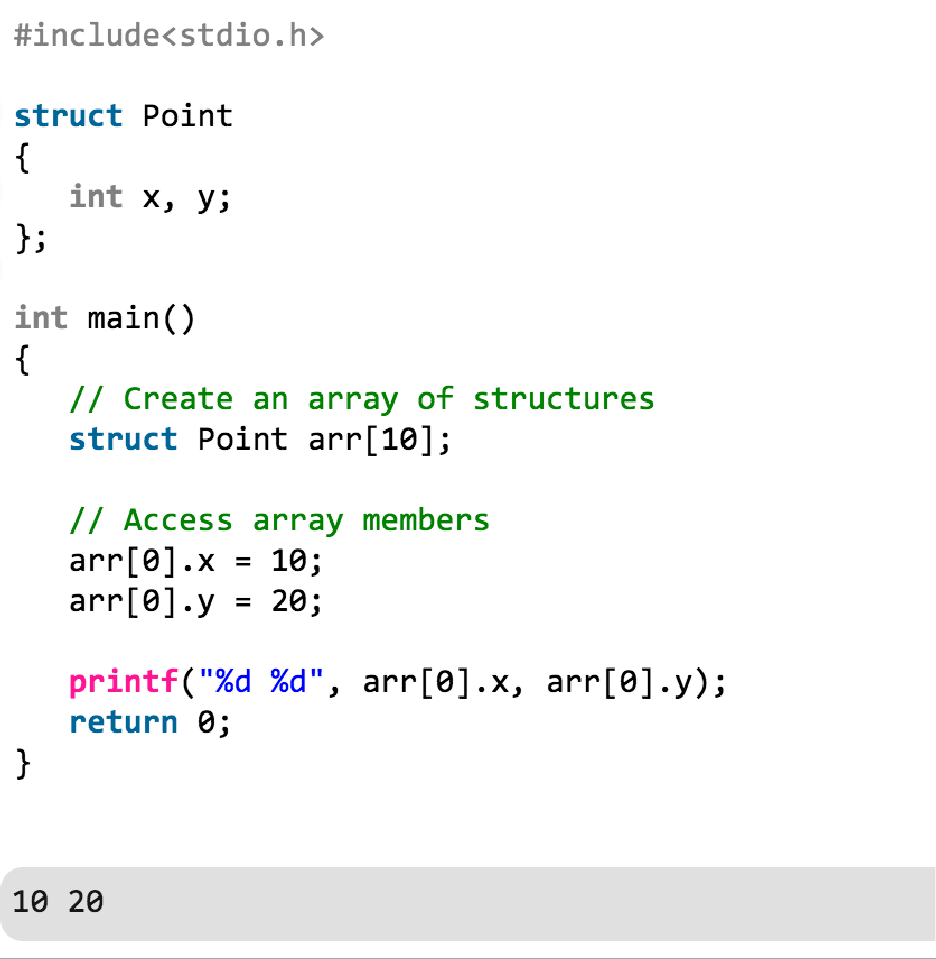
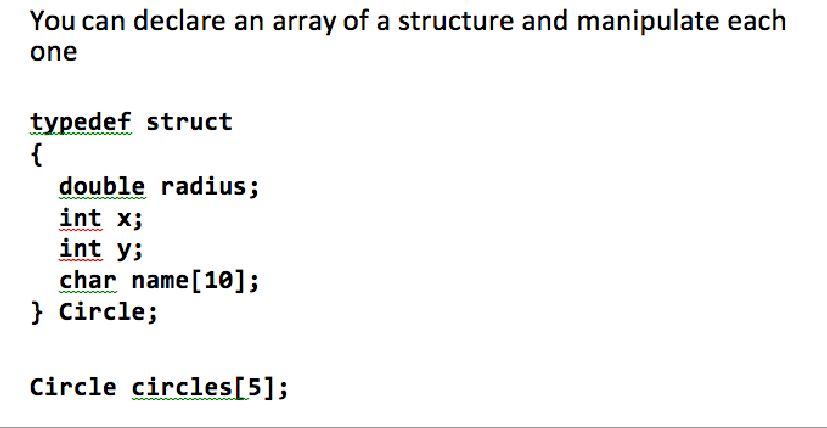
}

return 0;





**Array of Structures**



**LAB TASK**

* 1. Write a program in C that creates structure STUDENT. The STUDENT structure contains the following members: Name, Roll number, Attendance Marks, Test1 marks, Test2 marks and Test3 marks. Initialize two variables to store the record of two students. The program should display the name; roll number and total sessional marks of both the students.
  2. Write a program in C that creates structure TIME. Initialize two variables to store the starting and ending time of the race. The program should calculate the elapsed time in the third TIME variable and display it.
  3. Calculate the average Salary of the 10 employees by adding a function in the TASK 3.
  4. Calculate the Sizeof() Struct EMPLOYEE.
  5. Write a program to compare two dates entered by user. Make a structure named Date to store the elements day, month and year to store the dates. If the dates are equal, display "Dates are equal" otherwise display "Dates are not equal".