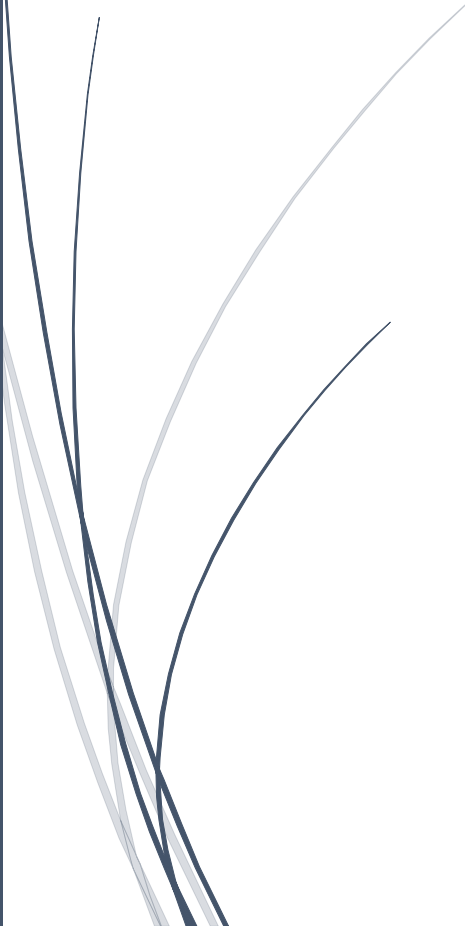


Course No. : EE207



EE207-Computer Programming

Module 5 note



Module 5: Structures & Pointers

Syllabus :-

Structures

- ❖ Declaration
- ❖ Definition
- ❖ Initialization of structures
- ❖ Unions

Pointers

- ❖ Concepts
- ❖ Declaration
- ❖ Initialization of pointer variables
- ❖ Accessing a Variable through its Pointer Chain of Pointers
- ❖ Pointer Expressions
- ❖ Pointer Increments
- ❖ Scale Factor
- ❖ Pointers and Arrays
- ❖ Examples

- ✓ 8 Hours for this module(for 1st teaching)
- ✓ In semester exam 20% marks from this module
- ✓ In this module we studying Structures & Pointers in C
- ✓ Reference ANSI C- Second edition

Structure

Structure can be used to represent a collection of data items of different types using a single name ie,structure is a mechanism for packing data of different types.

Defining a structure

The general format of a structure definition is as follow:-

```
struct tag-name
{
    datatype number_1;
    datatype number_2;
    .....
    datatype number_n;
};
```

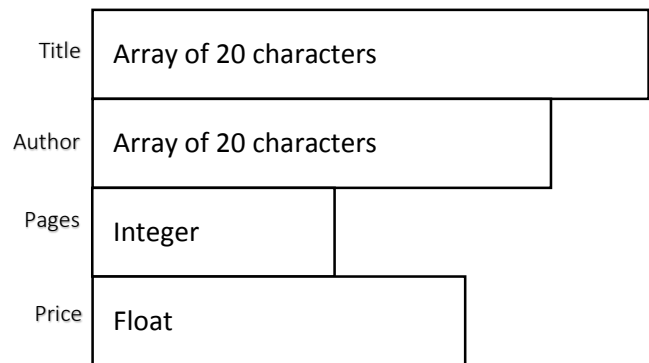
Eg:-

```

struct book
{
    char title[50];
    char author[15];
    int pages;
    float price;
};

```

Template



Declaration

A structure variable declaration includes the following elements

1. The keyword 'struct'
2. The structure tag-name
3. List of variables name separated by commas
4. A terminating semicolon

Eg:-

```
struct book
```

```

{
    char title[50];
    char author[15];
    int pages;
    float price;
}book1,book2;

```

```
struct book
```

```

{
    char title[50];
    char author[15];
    int pages;
    float price;
};

```

```
struct book book1,book2;
```

Structure and union

Structure	Union
<ul style="list-style-type: none"> • Keyword : struct • In structure, each member get specific space in memory <p>Eg:- struct student { int rollno; (int – 2 bytes)</p>	<ul style="list-style-type: none"> • Keyword : union • In union, the total memory space allocated is equal to the member of largest size <p>Eg:- union student</p>

<pre>char gender; (char – 1 bytes) float mark; (float – 4 bytes) }s1;</pre> <p>Total memory= 2+1+4=7</p> <ul style="list-style-type: none"> Access any member in any sequence s1.rollno=20 s1.mark=80.0 printf(“%d%f”,s1.rollno,s1.mark); All the members of a structure can be initialized 	<pre>{ int rollno; (int – 2 bytes) char gender; (char – 1 bytes) float mark; (float – 4 bytes) }s1;</pre> <p>Total memory= 4</p> <ul style="list-style-type: none"> We can only access that variable whose value is recently stored s1.rollno=20 s1.mark=80.0 printf(“%d%f”,s1.rollno); it gives erroneous output Only the first member can be initialized
--	--

Pointers

A pointer is derived data types in C. it is built from one of the fundamental data types available in C. Pointers contain memory address as their values. Since these address are the locations in the computer memory where program instruction and data are stored, pointers can be used to access and manipulate data stored in the memory

Benefits of using pointer

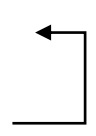
- ❖ Pointers are more efficient in handling arrays and data table
- ❖ Can be used to return multiple values from a function via function arguments
- ❖ Permit reference to functions
- ❖ Use of pointer arrays results in saving of data storage space in memory
- ❖ Allows C to support dynamic memory management
- ❖ Reduce length and complexity of programs
- ❖ Increase execution speed

Pointer variable

A variable that contains an address, which is location of another variable in memory. Since memory address are simply number, they can be assigned to some variable. Such variable that hold the memory address are called pointer variable

Suppose, we assign the address of ‘quantity’ to variable P. the link between the variable P and quantity can be visualized as follows:-

Variable	Value	Address
Quantity	179	5000
p	5000	5048



Declaration

The declaration of pointer variable as in the form

```
datatype *pointer_name;
```

Eg:- int *p;
 float *q;

Initialization of pointer

The process of assigning the address of a variable to a pointer variable is known as initialization.

Eg:-

```
int quantity;  
int *p;           // declaration  
p=&quantity;      // initialization
```

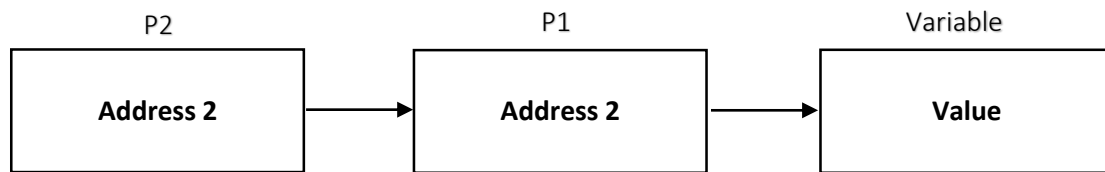
This will assign the address of variable quantity to the variable p.

Note : If p is pointer, q is a integer with value 3, then

- ❖ p=&q (Address=5000)
- ❖ *p=q

Chain pointer

It's possible to make a pointer to point to another pointer thus creating a chain of pointer



Here, the pointer P2 contains the address of the pointer variable P1, which points to the location that contains the desired value. This is known as multiple indirection

A variable that is a pointer to a pointer must be declared using additional indirection operator symbols in front of the name

```
Int **p;
```

Relation

- ❖ $p1 > p2$, $p1 == p2$, $p1 != p2$: these relations are allowed
- ❖ $p1/p2$, $p1 * p2$, $p1/3$: these relations are not allowed

Pointer increment & scale factor

A pointer p1 can be incremented like, **p1++**;

This will cause the pointer p1 to the next value of its type. For example; if p1 is integer pointer with initial address 2800, then after operation p1++;. Then the value of p1 will be 2802 and not 2801

When we increment a pointer, its value is increased by the 'length' of the data type that it points. This length is called the **scale factor**

Pointers and arrays

When an array is declared, the compiler allocates a base address and sufficient amount of storage to contain all the elements of the array in the continuous memory locations. The base address is the location of the first element

Eg:- `int x[5]={1,3,6,7,8}`

So, `a=&x[0]`; (a is a pointer)

`b=&x[1]`; (b is a pointer)

Note prepared by
VISHNU CV KTU live
vishnucvd@gmail.com

For more notes and question papers visit KTU live
All module Notes of Computer programming now available at our site



Now we available at WhatsApp 81-57097-880 (Send ur name reg. no & name to this to connect with us)