```
for( int i=0; i<5; i++)
    Printf ("%d", ptr[i]);

//free
    free(ptr);

set
    ptr = NULL;
```

→ good practice to set the pointer to NULL, to avoid "dangling pointer".

5) Structures

Arrays, strings → handle similar data

To handle dissimilar data } → structures

eg) 'book' contains collection of items like title, author, publisher, no. of pages, date of pub etc.

Structure — A collection of variables of different datatypes.

It is a user defined datatype.

that holds collection of elements of different type, under a single name.

Array	Structures
same type of ele.	different type of ele,
homogenous data	contains heterogenous data.

Suppose, if i wish to store in memory (name, price, no. of pages)

↓ i need 3 arrays

(or)

Structure variable.

Preferred.

ex) char name[3];
float price[3];
int pages[3];

1) It omits the fact, that we are dealing with the chars of a single entity book.

2) If we need to add more chars we need to increase the arrays.

Syntax of a structure

```
struct structure_tagname   optional
{
    datatype mem1;
    datatype mem2;
    .
    .
    datatype memN;
} [structure var1, structure var2, ....];
```

optional.

ex

```
struct book
{
    char title[50];
    char author[50];
```

float price;
int pages;

} book1;

↓
optional - can do it separately

Accessing the structure:-

To access the members of a structure, I need to declare structure variable.

ex) above ex: Struct book book1;

struct structure name var name ;

↗
general syntax.

book1.title
book1.author
book1.price
book1.pages

} create any no: of vars, book2, book3, ... Access the structure member, using the dot operator. membership

Initialising the structure:-

ex)
```
struct mystruct
{
    int a;
    float b;
    char c;
} s1 = { 5, 3.6, 'a' };
```

1st way

If more than 1 variable,

struct mystruct s2 = { 6, 4.6, 'b' };

" " s3 = { 0 };

like var, pointers, arrays & strings, struct vars too can be initialized where they are declared

2nd way

```
#include <stdio.h>
  Struct book
  {
     char title[10];
     char author[20];
     double Price;
     int pages;
  }
Void main()
{
  struct book book1 = { "Learn c", "Dennis Ritchie",
                        675.50, 325 };

  Printf (" %.s", book1.title);
  Printf (" %.s", book1.author);
  Printf (" %f", book1.price);
  Printf (" %.d", book1.pages);

}.
```

→ structure can be defined before the main function.

Def. value of structure } : Garbage (decl. inside main)
zero (decl. outside main).

Partial initialisation of strct is possible, but in order only

3rd way: get value from scanf().

→ If a structure variable is initated to a value {0}, then all its elements are set to 0.

Storage of structures

↳ always stored in contiguous memory locations.

ex:

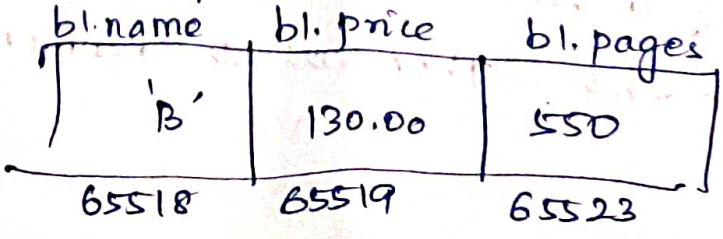
```
#include <stdio.h>
Struct book
{
    char name;
    float price;
    int pages;
};
main()
{
    Struct book b1 = {'B', 130.00, 550};
printf ("Add. of name = %u", &b1.name);
    "       "          %u", &b1.price);
    "       "          %u", &b1.pages);
}
```

O/P:

Addr. of name = 65518

" " price = 65519

" " pages = 65523



copying the structure

can be copied either one/one or all at one shot.

```
#include <stdio.h>
  struct employee
  {
      char name[10];
      int age;
      float salary;
  } e1={"Sanjay", 30, 50,000.00};
void main()
{   struct employee e2, e3;
```

Piece-meal copying.

```
    strcpy (e2.name, e1.name);
    e2.age = e1.age;
    e2.salary = e1.salary;
```

copying at a short

```
    e3 = e2;
Printf(" %s %d %f", e1.name, e1.age, e1.salary);
    "   (    "      , e2.name,  "      " );
    "   (    "      , e3.name,  "      " );
}
```

o/p

Sanjay 30 50,000.00

Sanjay 30 50000.00

Sanjay 30 50000.00

Typedef

↓ redefine the name of an existing variable type.

Syntax :-

typedef existing datatype newdatatype ;

typedef unsigned long int UTI ;

Now declare variables of unsigned long int by

UTI var1, var2 ;

instead of

unsigned long int var1, var2 ;

Provides Shortcut

usually uppercase letters used for } typedef

ex2 ;

```
Struct employee
{
    char name[30];
    int age;
    float bs;
};
```

typedef Struct employee EMP;

EMP e1, e2 ;

reducing the length & complexity of

otherway;

```
typedef struct employee
{
    char name[30];
    int age;
    float bs;
} EMP;
```

→ create one name

```
EMP e1,e2;
```

→ use that in main.

Array of structures

If the no:of books increase, create an array of structures.

```
#include <stdio.h>
struct book
{
    char name;
    float price;
    int pages;
};
struct book b[10];
void main()
{
    int i;
    int dh;
    for(i=0; i<=9; i++)
    {
        Printf("Enter name, price & pages");
        scanf("%c%f%d", &b[i].name,
                        &b[i].price, &b[i].pages);
```