LECTURE - 40

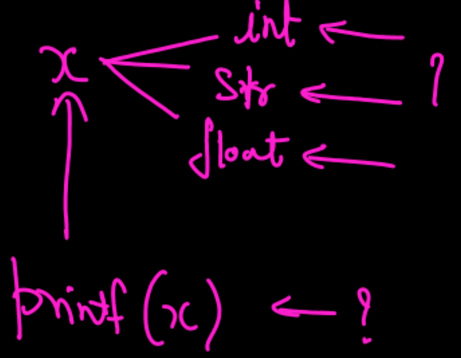
STRUCTURES IN C (PART - 02)

PROGRAMMING IN 'C'

Structure:

```
Struct Student {
        int R_no;
        Char[30] s_name;
        float marks;
    };

main ( ) {
    Struct student x, y ;
    x.R_no = 1 ;
    strcpy ( x.s_name, "Aditi" );
    x.marks = 97.8;
    printf ("%d", x);
}
```



Correction

→ Garbage Value

```
Struct employee {
    int empid;
    char empname[30];
    float empsalary;
};
```

Compiler → No memory Allocation

members

* Structures are Basically defined globally, but they may have a local Scope

```
Struct employee Raju;
```

Datatype

Memory Allocated

```
Raju.empid = 1007;
Raju.empname = "Raju Srivastava";
Raju.empsalary = 17500.00;
```

```
int x;          x ← 4Byte
Char y;         y ← 1Byte
float z;        z ← 4Byte
double m;       m ← 8Byter
int *ptr;       ptr ← 8Byte
```

Raju

100 →	1007	empid ← 4Bytes
150 →	"Raju Srivastava"	empname ← 30Bytes
200 →	17500.000000	empsalary ← 4Bytes
		38Bytes

printf ("%d", sizeof (Raju)); ← 38 Bytes

⇒ Structure can hold multiple data elements of different datatype. ✓

↳ A Structure can hold another Structure.

⇓ Nested Structure

dob → Year, MM, DD

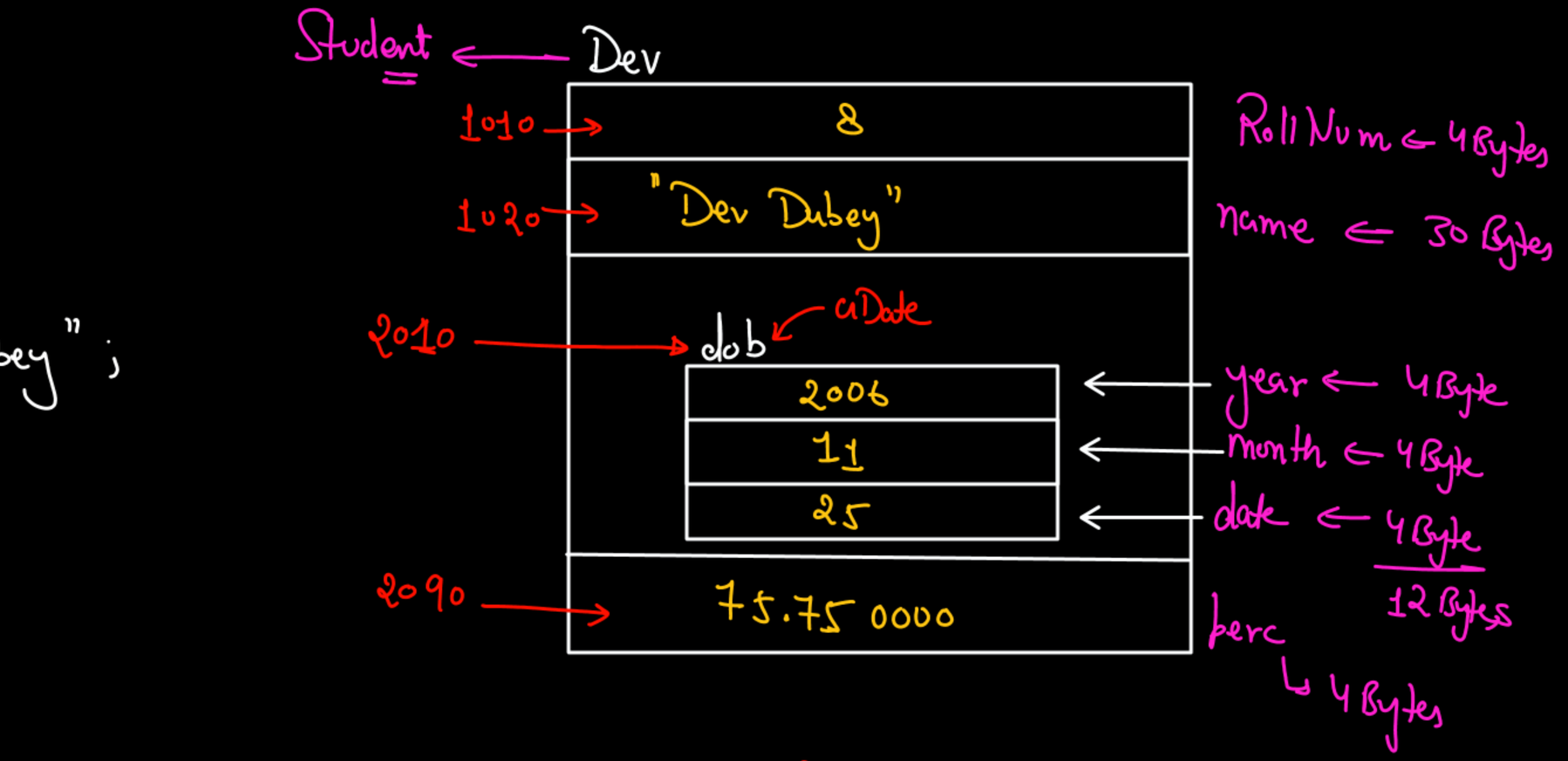
```
struct student {
    int RollNum;
    char name[30];
    struct aDate dob;
    float perc;
};
```

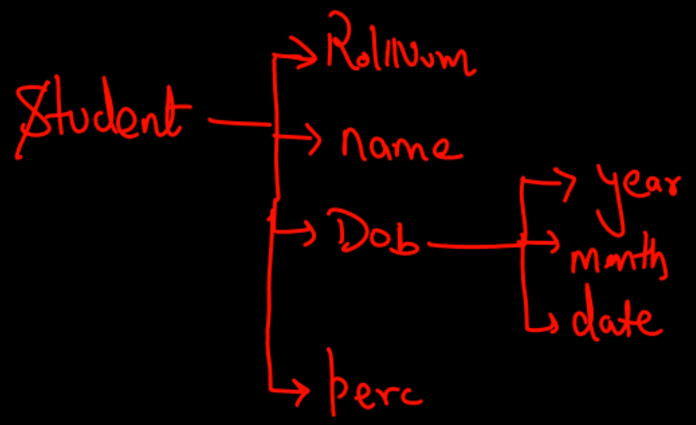
```
struct aDate {
    int year;
    int month;
    int date;
};
```

```
main ( ) {
    struct student Dev;
```

```c
int main ( ) {
    struct student Dev;
    Dev. RollNum = 8;
    Dev. name = "Dev Dubey";
    Dev. dob. year = 2006;
    Dev. dob. month = 11;
    Dev. dob. day = 25;
    Dev. perc = 75.75;
    printf ("%d", sizeof (Dev));
    printf ("%d", Dev. dob. year);
    return 0;
}
```

→ 50 Bytes





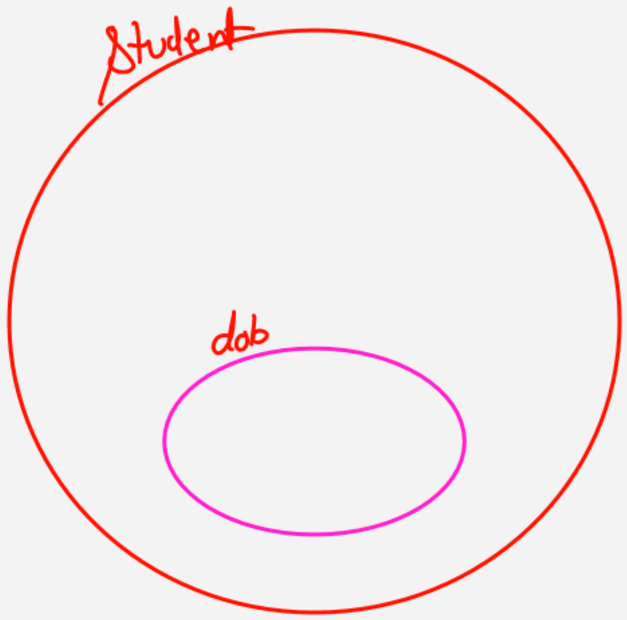
File Edit Selection View Go Run ··· Code

Welcome | structures.c | NESTEDSTRUCT.c

NESTEDSTRUCT.c > student

```c
#include<stdio.h>
#include<string.h>
struct dob{
    int year;
    int month;
    int day;
};
struct student{
    int rollNum; //4
    char name[20]; // 20
    struct dob date; // 12
    float perc; //4
};
int main(){
    struct student s;
    printf("The size of student is:%d bytes\n", sizeof(s));
    s.rollNum = 10;
    strcpy(s.name, "Awantika");
    s.date.year = 2006;
    s.date.month = 11;
    s.date.day = 25;
    printf("The roll No of student is:%d\n", s.rollNum);
    printf("The DOB of student is %d/%d/%d.", s.date.day, s.
    date.month, s.date.year);
    return 0;
```

```
PS C:\Users\sagar\OneDrive\Desktop\Daily Notes\Aditi Chand\Programming in C\Code> cd "c:\Users\sagar\OneDrive\Desktop\Daily Notes\Aditi Chand\Programming in C\Code\" ; if ($?) { g++ NESTEDSTRUCT.C -o NESTEDSTRUCT } ; if ($?) { .\NESTEDSTRUCT }
The size of student is:40 bytes
The roll No of student is:10
The DOB of student is 25/11/2006.
PS C:\Users\sagar\OneDrive\Desktop\Daily Notes\Aditi Chand\Programming in C\Code>
```



Ln 8, Col 15 Spaces: 4 UTF-8 CRLF C++ Win32 Prettier

ENG 9:46 PM 25-Dec-24

Array of Structures :

⇓

Store the Structures in Succesive memory location.

⇓

Datatype

int array	float array	Char array	double array	← Built in type

Structure array ← User defined ← allowed

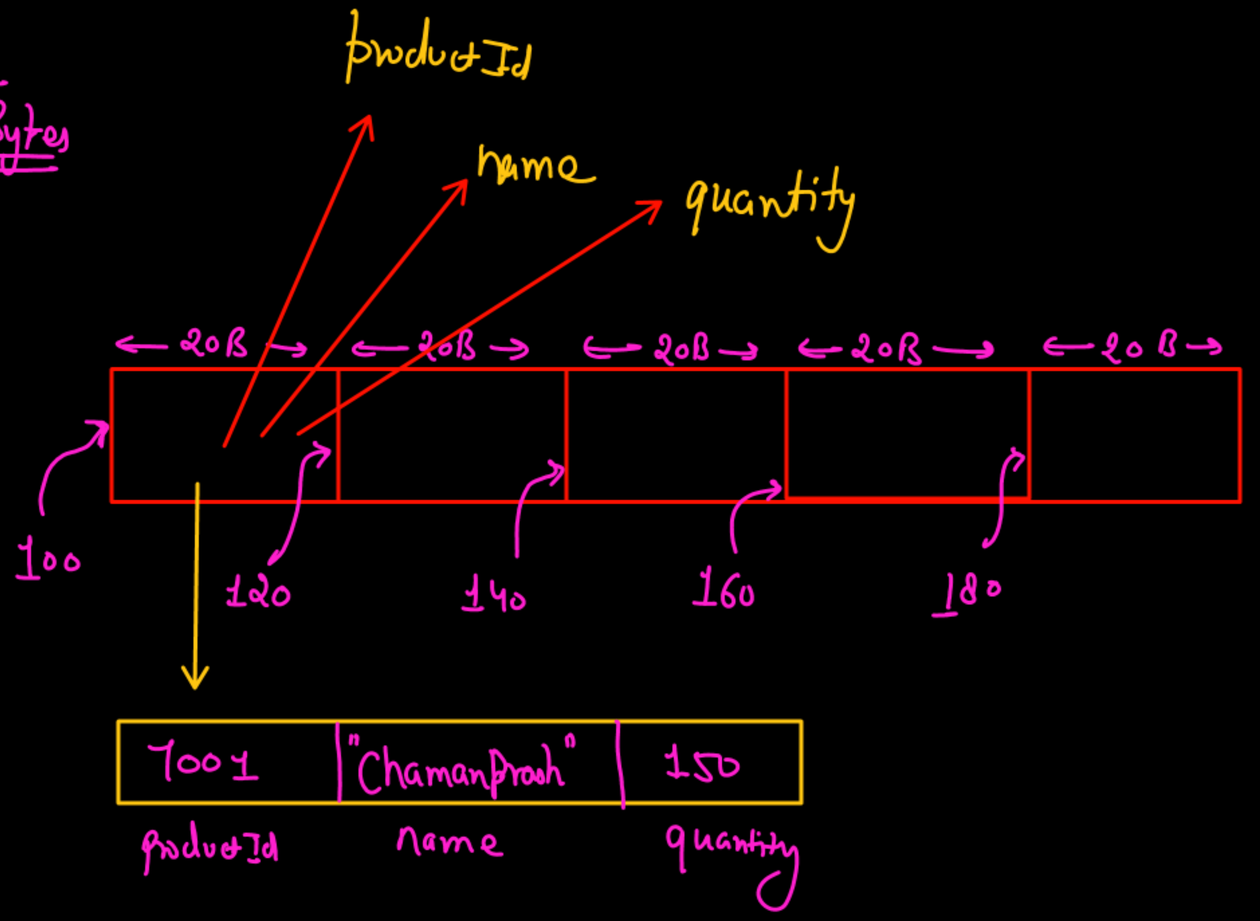
array of array ← derived data type

⇒ All theory is same as other arrays ←

```
struct Product {
    int ProductId;
    char name[12];
    int quantity;
};

int main() {
    int arr[10];
    struct Product p[5];
    p[0].productId = 7001;
    p[0].name = "ChamanPrash";
    p[0].quantity = 150;
    printf("%d", sizeof(p));
```

int - 4
Char → 12
int ⇒ 4
20 Bytes



← no. of element x (Size of one element) → 5 x 20 = 100 Bytes.

Welcome | structures.c | NESTEDSTRUCT.c | arrayofstruct.c

arrayofstruct.c > main()

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

struct product
{
    int productID;
    int quantity;
};

int main(){
    struct product p1 = {101, 25};
    printf("ID of prod = %d\n", p1.productID);
    printf("quantity of prod = %d\n", p1.quantity);
    return 0;
}
```

char name[30]

Implicit definition

printf("%s", p1.name);

{101, "Dabur", 25}

```
ID of prod = 101
quantity of prod = 25
PS>
```

Welcome | structures.c | NESTEDSTRUCT.c | arrayofstruct.c

arrayofstruct.c > main()

```c
#include<stdio.h>
#include<string.h>
struct product
{
    int productID;
    char pname[12];
    int quantity;
};
int main(){
    struct product p[5];
    p[0].productID = 7001;
    strcpy(p[0].pname, "Dabur");
    p[0].quantity = 15;
    printf("%d\n", p[0]); //address of first element
    printf("Index = 0, Product Name = %s\n", p[0].pname);
    printf("Size of array P = %d bytes", sizeof(p));
    return 0;
}
```

Addres →

```
6421920
Index = 0, Product Name = Dabur
Size of array P = 100 bytes
PS>
```

Pointers to the Structures

↳ Tomorrow → Linked List ← DS

[(Arrays + Structure) + Pointers] ← Arithmetic