CS & IT ENGINEERING

Programming in C

Structure and Union

(In One Shot)

By- Pankaj Sharma sir

TOPICS TO BE COVERED

- Structure and Union

Structure

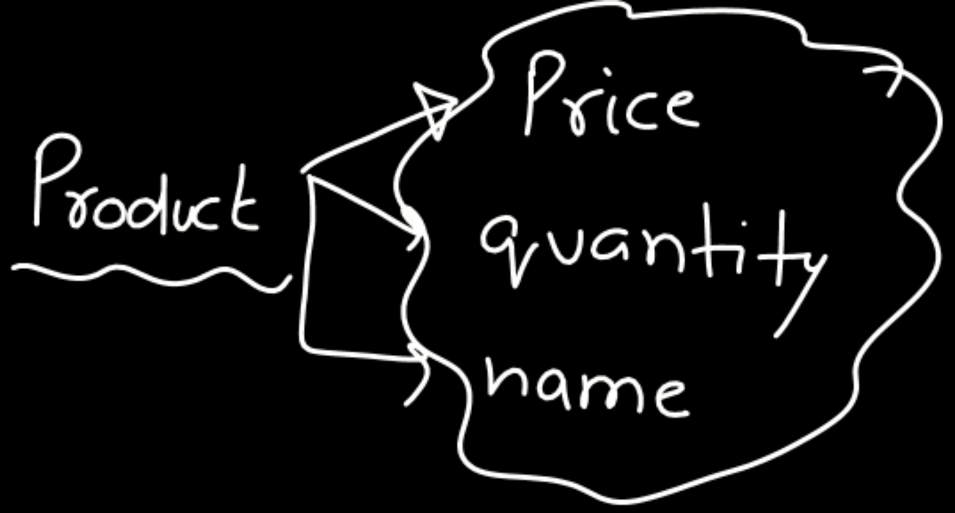
Hospital

Docter

* user defined data type :

stu
- name
- Roll
- Age
- Address

1 Stu
- name : char arr[20]
- Roll : int Roll
- Age : int Age
- Marks : float marks


Product
Price
quantity
name

Structure

* User defined data type
* struct is the keyword used to create user defined data type

```
[struct] student {
        int Roll;
        char name[20];
    };
```

$\boxed{\text{struct}} \rightarrow$ Keyword

```
[struct] Patient{
    int Age;
    char name[20];
    char disease[40];
    ===
    };
```

```
struct student{
        int Roll;
        char name[20];
      };

void main(){

        ___
        ___
        ___

}
```

info → compiler

Just like primitive data type int, char, float a new data type also exist

↳ struct student

→ No memory allocated

→ Bluprint/template

```
struct student{
        int Roll;
        Char name[20];
            };
        void main(){
            int    ;      X
        struct student  ;   X
```

primitive d.t ← int

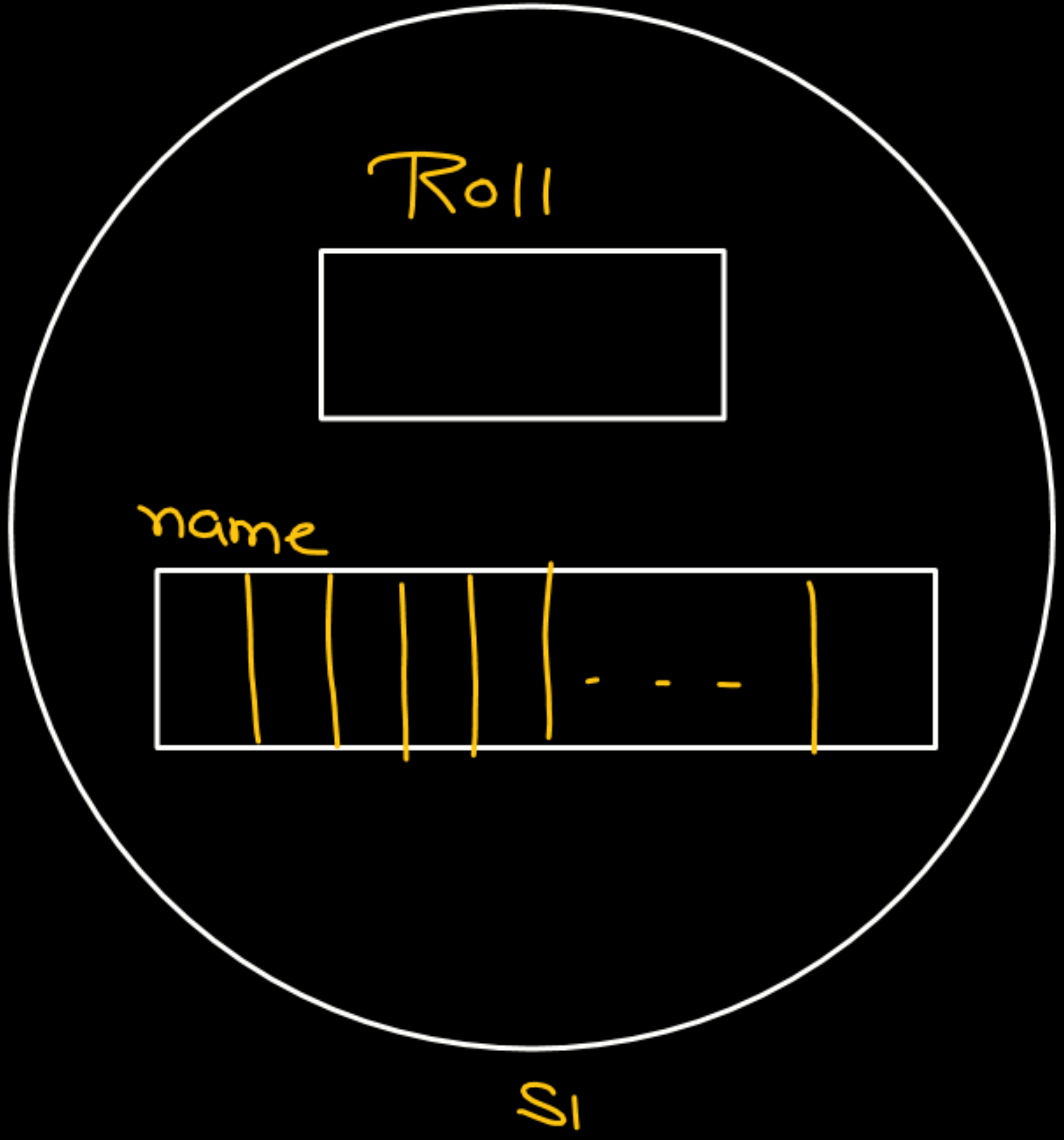
user -defined d.t ← struct student

```
struct student{
        int Roll;
        char name[20];
        };
void main(){
        int a ; //
  struct student s1 ; //
```

Variable

```
struct student {
        int Roll;
        char name[20];
        };

void main(){
     struct student s1;
```

$\rightarrow$ group of 2 members



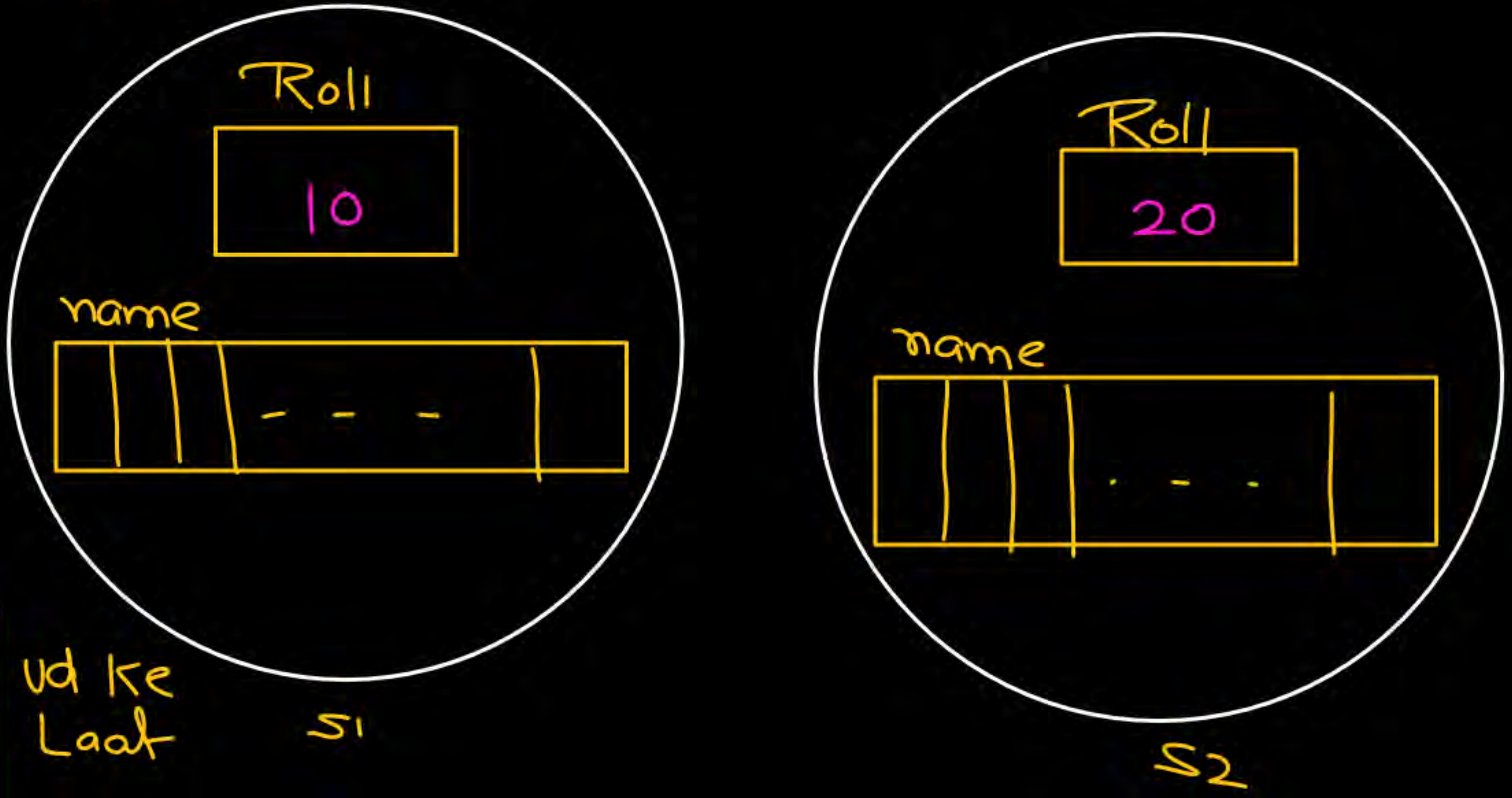
```
struct student {
        int Roll;
        char name[20];
        };
void main(){
      struct student s1, s2;
      s1.Roll = 10; ✓
      s2.Roll = 20; ✓
      s1.name = "Pankaj";
```

ud ke Laat

solution

```
strcpy(s1.name, "Pankaj"); ✓
```

• ⟹ Membership operator



```
struct student {
        int Roll = 10 ;
        char name[20] = "Pankaj" ;
     };

void main() {
        struct student s1, s2 ;

            ___
            ___
            ___
            ___

     }
```

No
Memory
is
allocated

→ Tag of structure

```
struct student{
        int Roll;
       char name[20];
          };

void main(){
  struct student s1,s2;

        =
        —

          }
void fun( ) {
   struct student s1,s2;
          =
          ≡
         }
```

```
struct { int Roll;
        char name[20];
        } s1,s2,s3;  ✓

void main(){
Not able to create variable
    Of type student

        }

  void fun(){
   Not able ______

        }
```

→ गंगाधर

```
typedef struct student{
          int Roll;
        char name[20];
                 } Pankaj;
                 शक्तिमान
 void main(){
   Pankaj s1, s2;  ✓

        ≡

        }
```

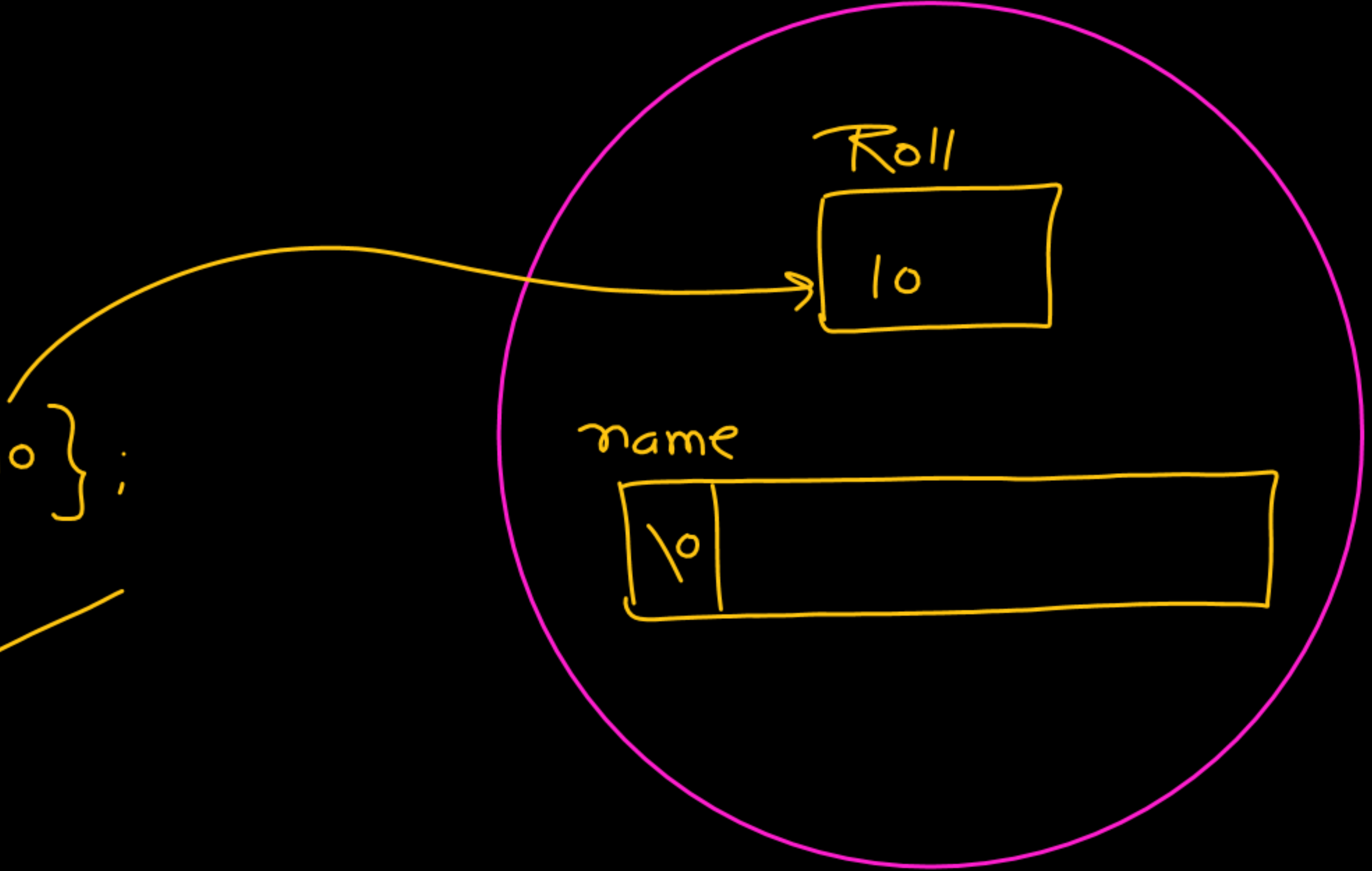
① int a = "Pankaj"; ud ke laat

```
struct student {
        int Roll;        → 1st member
        char name[20];   → 2nd member
    };                   ↓
void main(){

 struct student s1 = { 10, "Pankaj" }; ✓
struct student {            1st member
        int Roll;
                            2nd member
      char name[20];
              };
void main(){

struct student s1 =  { "Pankaj" };   X      int a[3] = {1,2};

   { s1.Roll, s1.name } = { "Pankaj" }
```

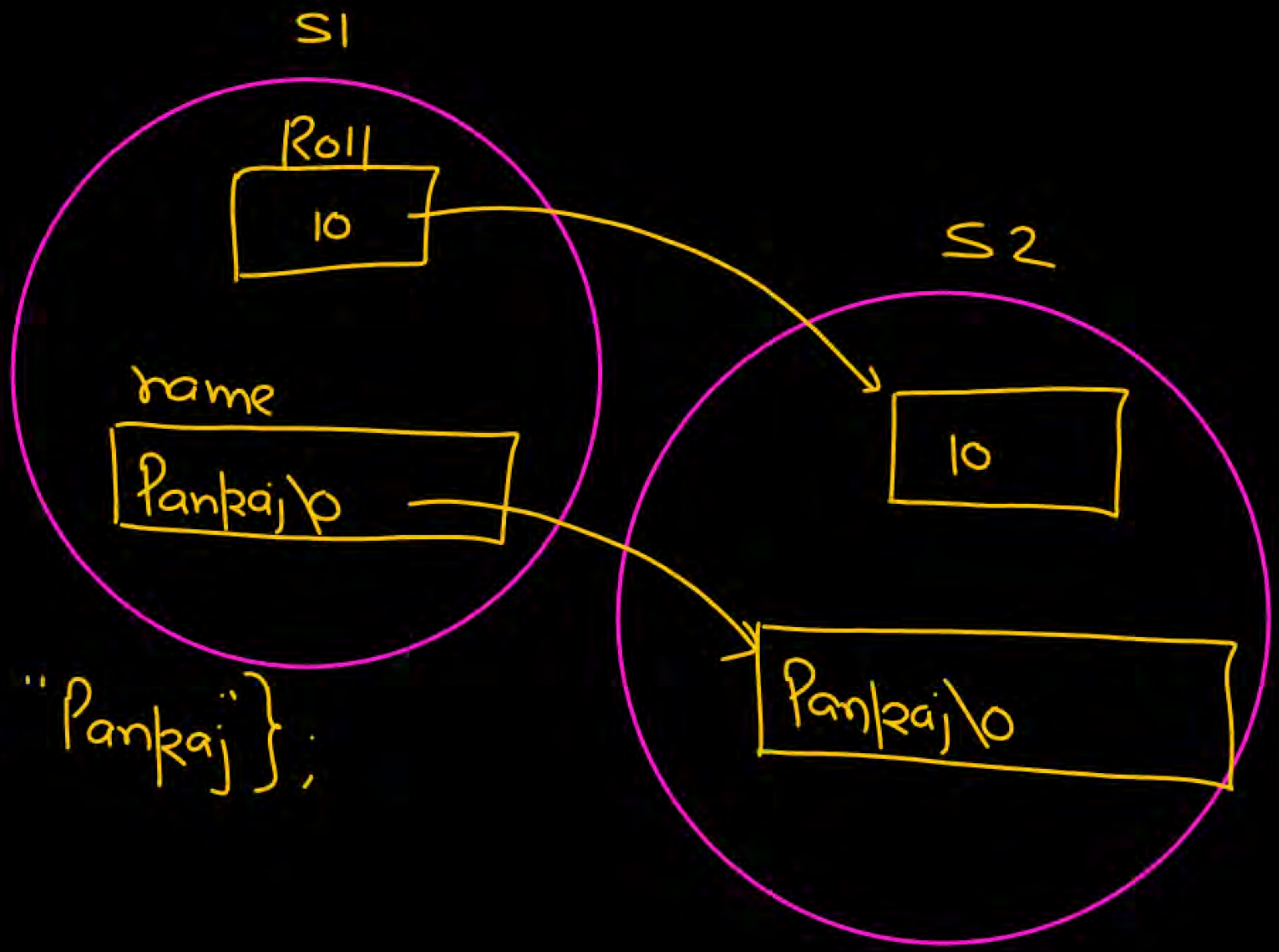
Partial Initialization

```
struct student {
        int Roll;          1st member
      char name[20];       2nd member
         };
void main(){

 struct student s1 = { 10 };
```

✓



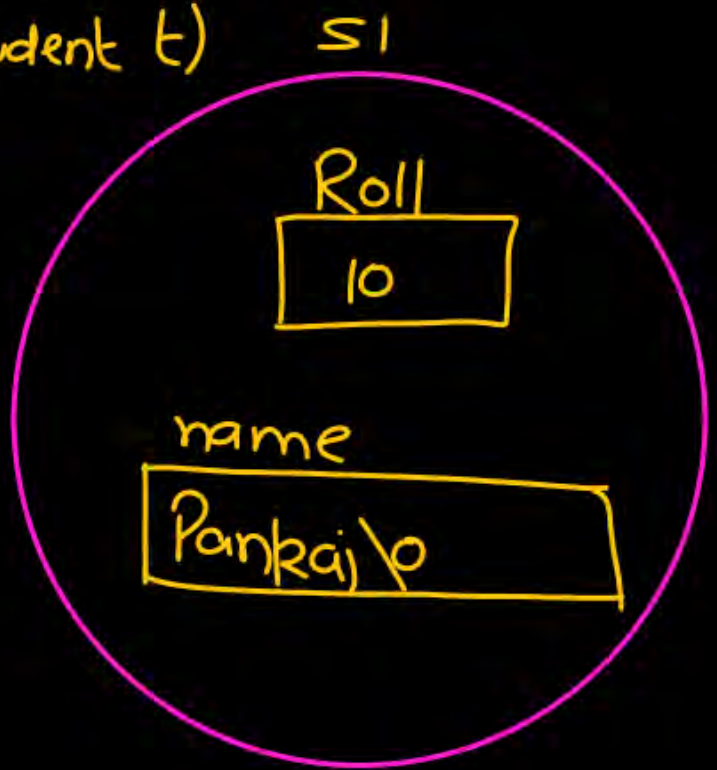
```
struct student {
        int Roll;
      char name[20];
        };
void main(){

struct student s1 = {10, "Pankaj"};
struct student s2 = s1;
```

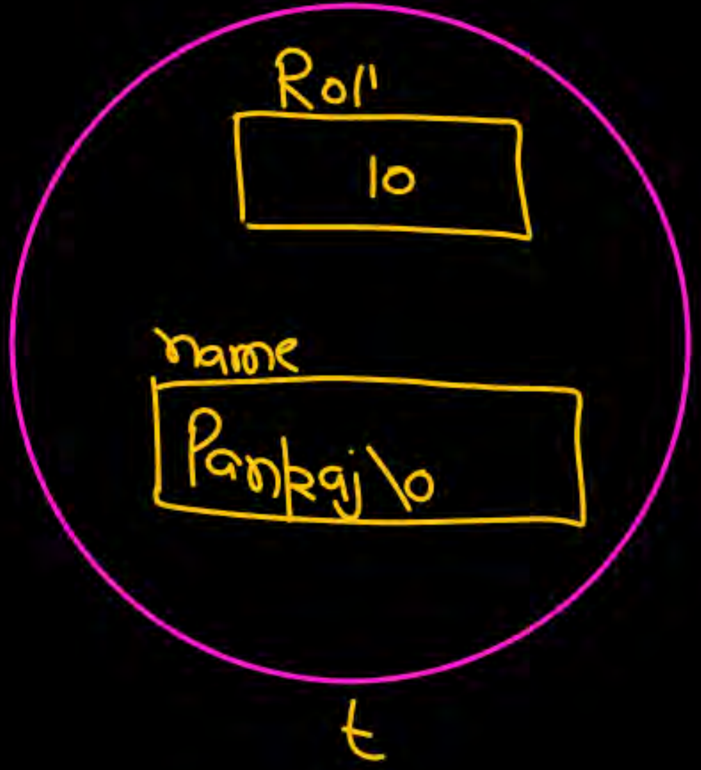


```
struct student {
        int Roll;
      char name[20];
        };

void display(struct student);
 void main(){
    struct student s1 = {10,"Pankaj"};

    display(s1);

      }
```

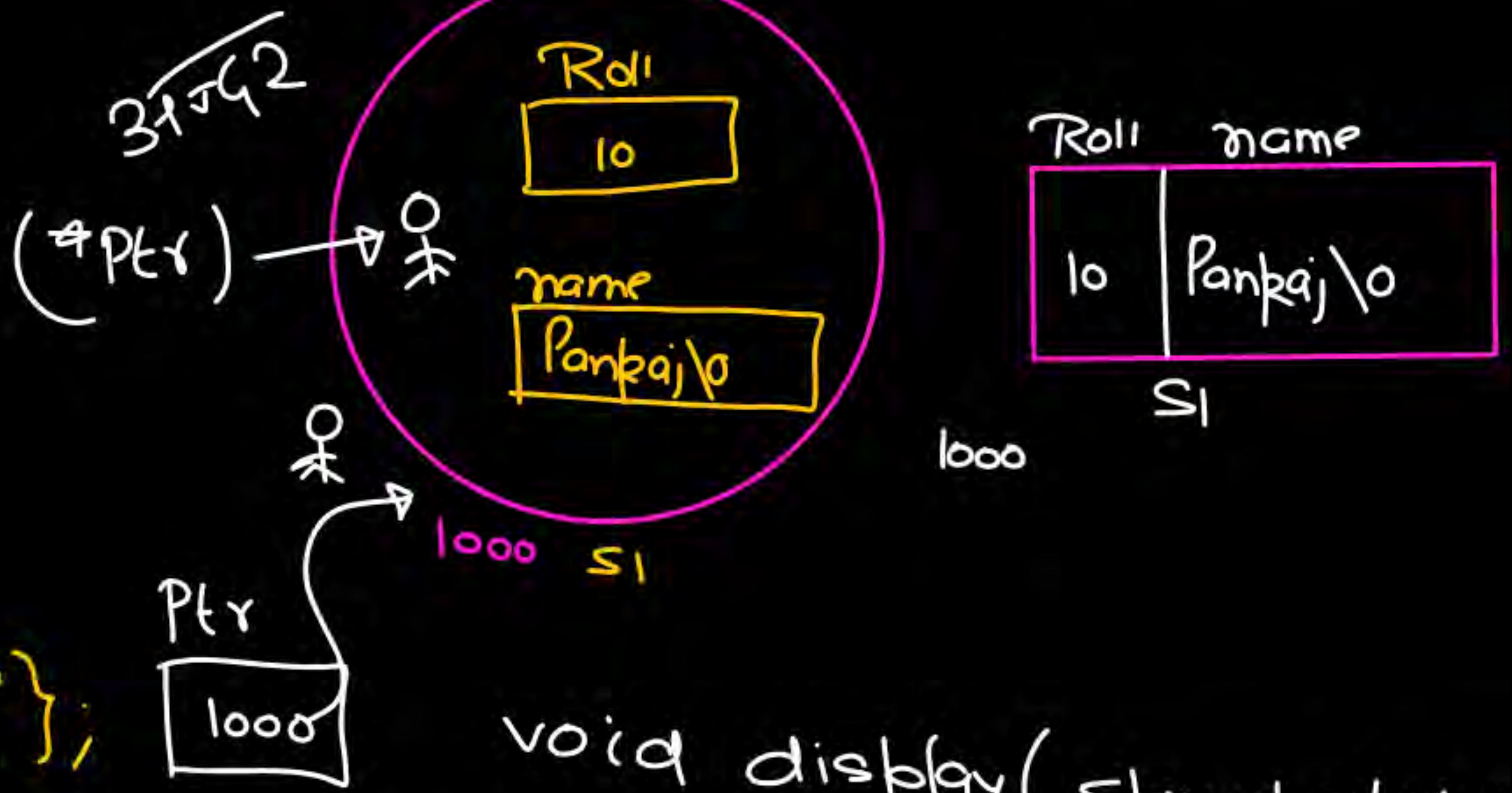
Call by value

```
void display(struct student t)
  {

printf("%d", t.Roll);
printf("%s", t.name);

  }
```



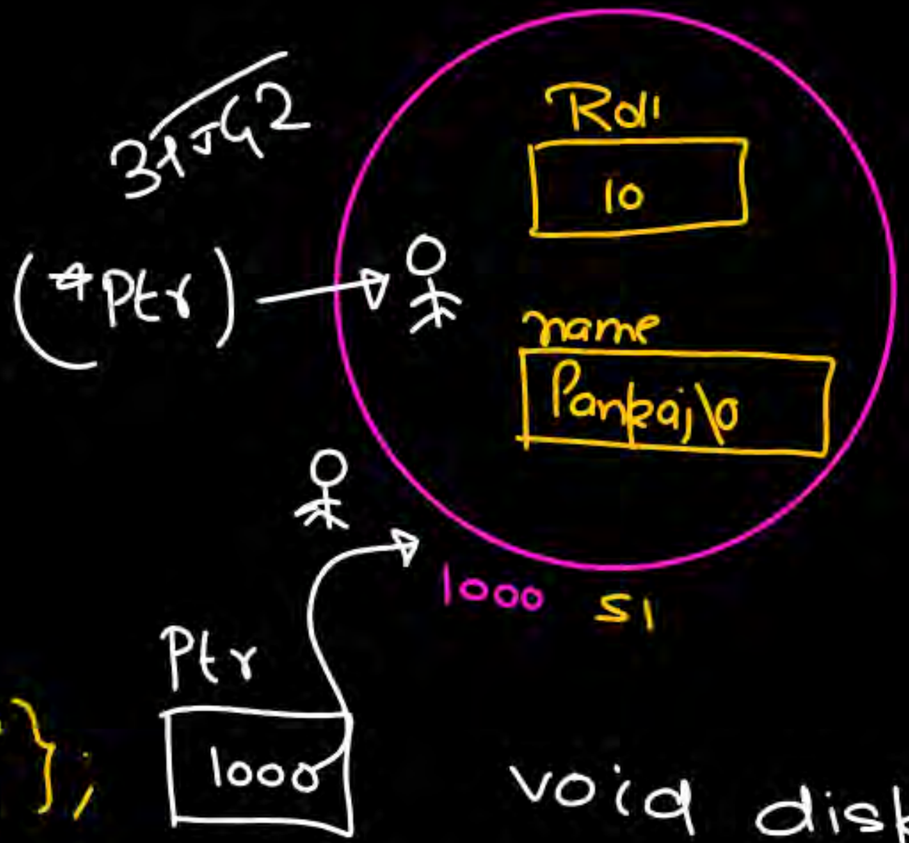


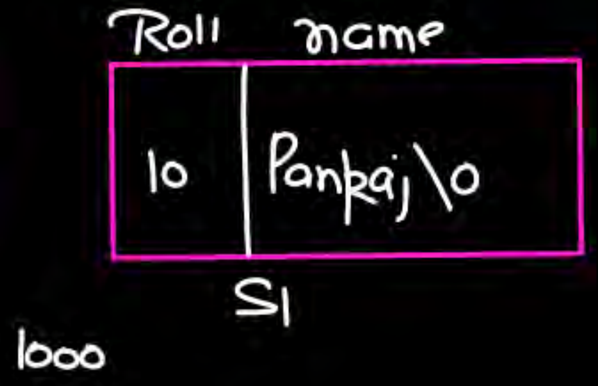
```
struct student {
    int Roll;
    char name[20]; };

void display(                    );

void main(){
    struct student s1 = {10,"Pankaj"};
    display(&s1);
```



```
void display(struct student *ptr)
{
    pf("%d", (*ptr).Roll);
    pf("%s", (*ptr).name);
}
```

```
typedef struct student {
        int Roll;
      char name[20]; } test ;

void display(struct student * );

void main(){
      struct student s1 = {10,"Pankaj"};
      display (&s1);
            pf("%d", s1.Roll);
            pf("%s", s1.name);

        }
```





```
void display( test *ptr)
     {
  pf("%d", (*ptr).Roll);
     pf("%s", (*ptr).name);

     }
```

* static variable can't be member of a structure.

1.

Ptr : Pointer to structure

(*Ptr).member1

(*Ptr).member2

$\Rightarrow$

Ptr → Roll ;

Ptr → name ;

Ptr → member1 ;

Ptr → name ;

2. (   ) → member

$\downarrow$

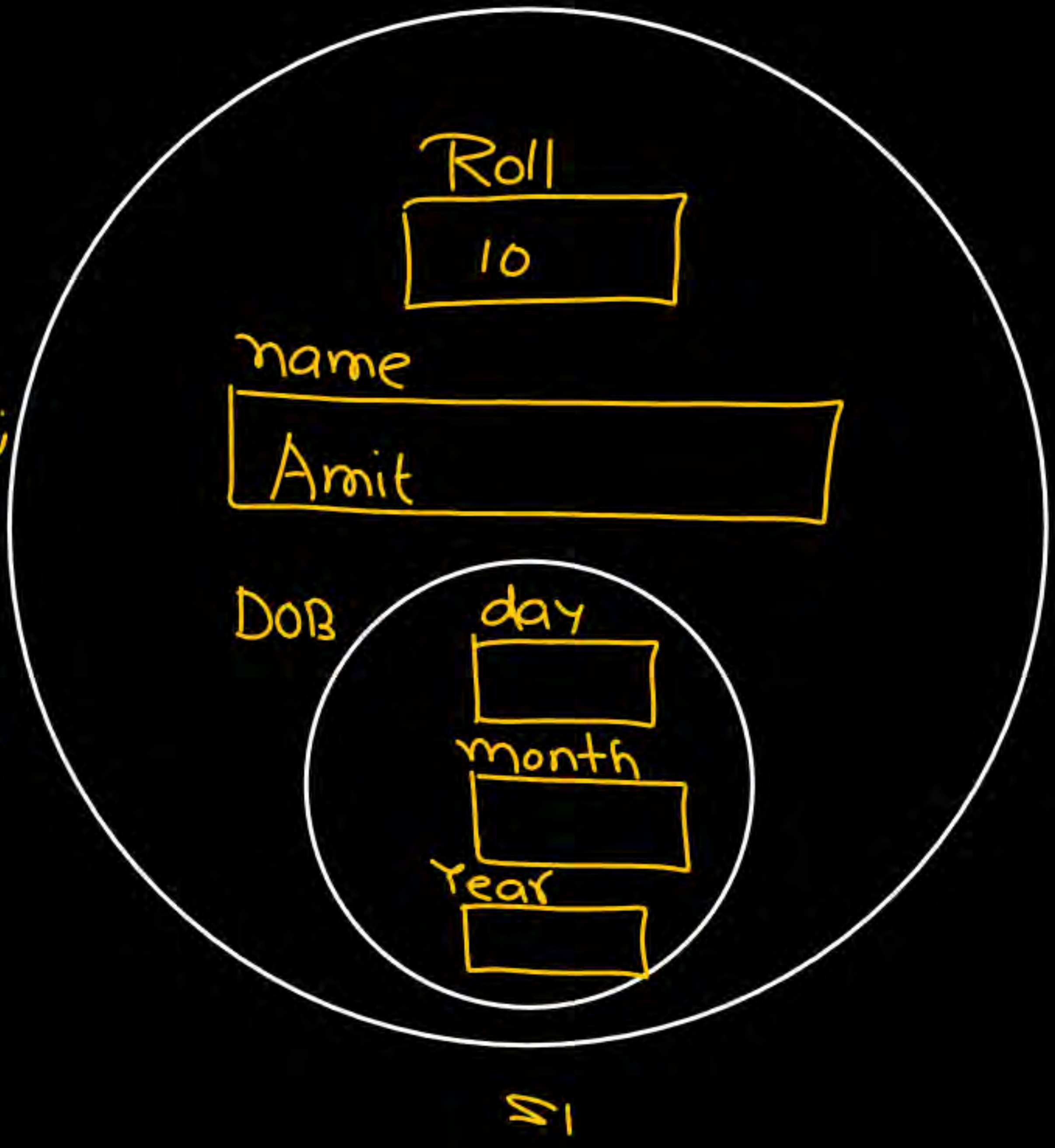
Pointer to structure

Date of Birth $\underbrace{02}_{\text{day}}|\underbrace{03}_{\text{month}}|\underbrace{1982}_{\text{year}}$

```
struct Date_of_Birth{
        int day;
        int month;
        int Year;};

struct student{
      int Roll;                    → Primitive data type
      char name[20];               → derived data type
   struct Date_of_Birth DOB;       → user-defined

            };
```

```
struct Date_of_Birth{
        int day;
        int month;
        int Year;};

struct student{
      int Roll;
      char name[20];
   struct Date_of_Birth DOB;

            };
```

```
void main(){
   struct student s1;
   s1.Roll=10;
   strcpy(s1.name,"Amit");
   s1.DOB.day = 2;
   s1.DOB.month = 3;
   s1.DOB.year = 1982;
```



```
struct Date_of_Birth{
        int day;
        int month;
        int Year;};

struct student{
      int Roll;
      char name[20];
   struct Date_of_Birth DOB;

           };
```

```
    void fun(){
Valid  struct Date_of_Birth a;
✓
```

$\Longrightarrow$

```
struct student{
      int Roll;
      char name[20];
      struct Date_of_Birth{
            int day;
            int month;
            int Year;} DOB;

      }
```

```
void fun(){
✗  struct Date_of_Birth d;
```

```
struct Date_of_Birth{
        int day;
        int month;
        int Year;};

struct student{
      int Roll;
      char name[20];
   struct Date_of_Birth DOB;

          };
```

```
   void fun(){
Valid    struct Date_of_Birth a;
 ✓
```

$\Longrightarrow$

```
struct student{

      int Roll;
      char name[20];
      struct{

          int day;
          int month;
          int Year;} DOB;

      }
void fun(){
✗  struct Date_of_Birth d;
```

```
void main(){
    char name[20];   // Memory

    name = "Pankaj"   ✗
```

✓

```
char name[20] = "Pankaj";
```

THANK YOU GW SOLDIERS !