Let’s solve it

Unnamed structure

① struct student
{

};

struct student s1;

② struct student
{

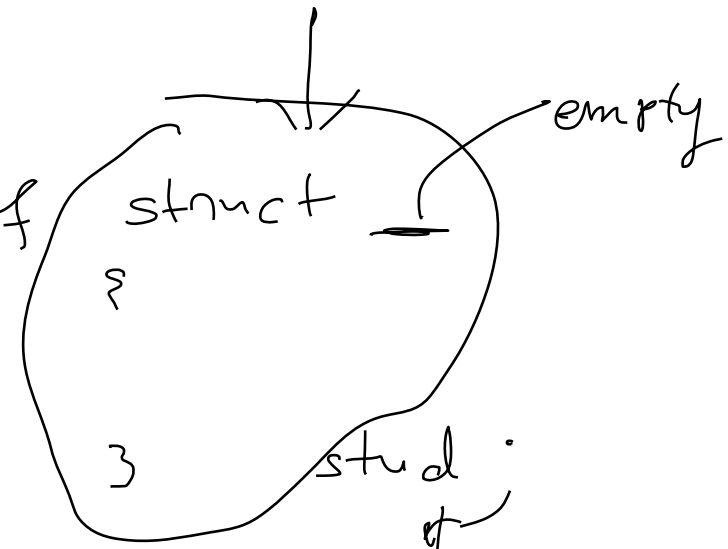
};
typedef struct student stud;

stud s1;

③ typedef struct
{

} stud;

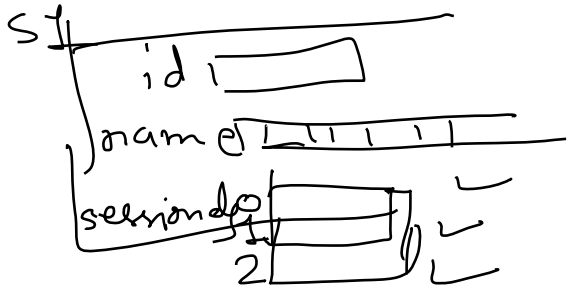
stud s1;



- Array of structure variables stud s[100];

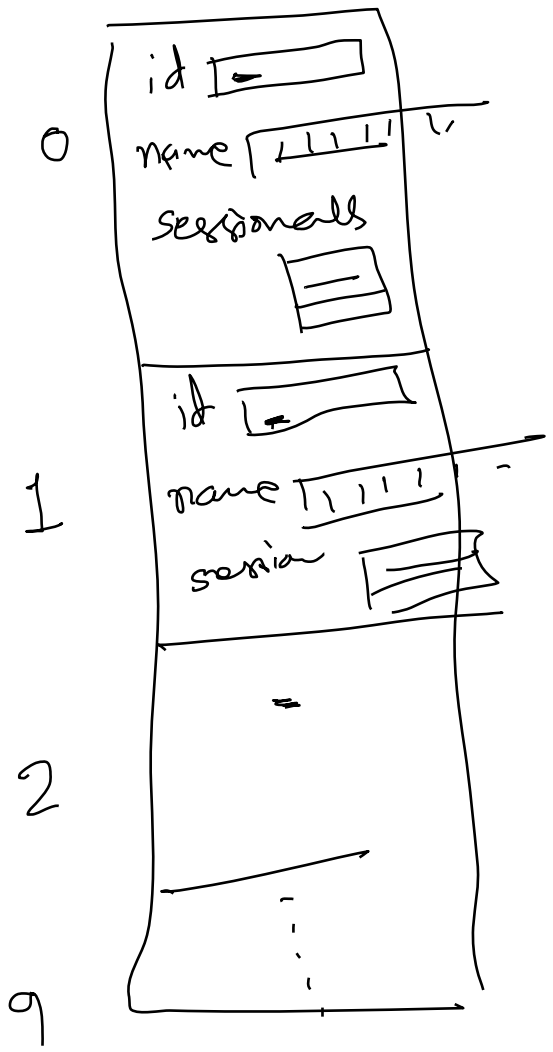
- Array within structure definition

```
struct student
{   int id;
    char name[25]; // string /char array
    int sessionals[3]; // Integer array
};
```




```
struct student st;
scanf("%d", &st.id);
scanf("%s", st.name);
for(i=0; i<3; i++)
{
  scanf("%d", &st.sessionals[i]);
}
```

struct student s[10];



minimum

33 bytes

33 bytes

33 bytes

1

P.S.

slack byte later

```
// Reading information of 10 students
for( si=0; si<10; si++)
{
    scanf("%d", &s[si].id);
    scanf("%s", s[si].name);

    for( mj=0; mj<3; mj++)
    {
        scanf("%d", &s[si].sessionals[mj]);
    }
}
```

structures and Functions (UDF)

```
main()
{
    struct student s1;// -- s1 = {1, "James", {20,30,10}};
    s1.id = 1;
    strcpy(s1.name, "James");//  s1.name = "James"; ?X
    s1.sessionals[0] = 20;
    s1.sessionals[1] = 30;
    s1.sessionals[2] = 10;

    displayV2(s1);//actual
    displayV2(s1.id, s1.name, s1.sessionals[0], s1.sessionals[1], s1.sessionals[2]);
```

struct student

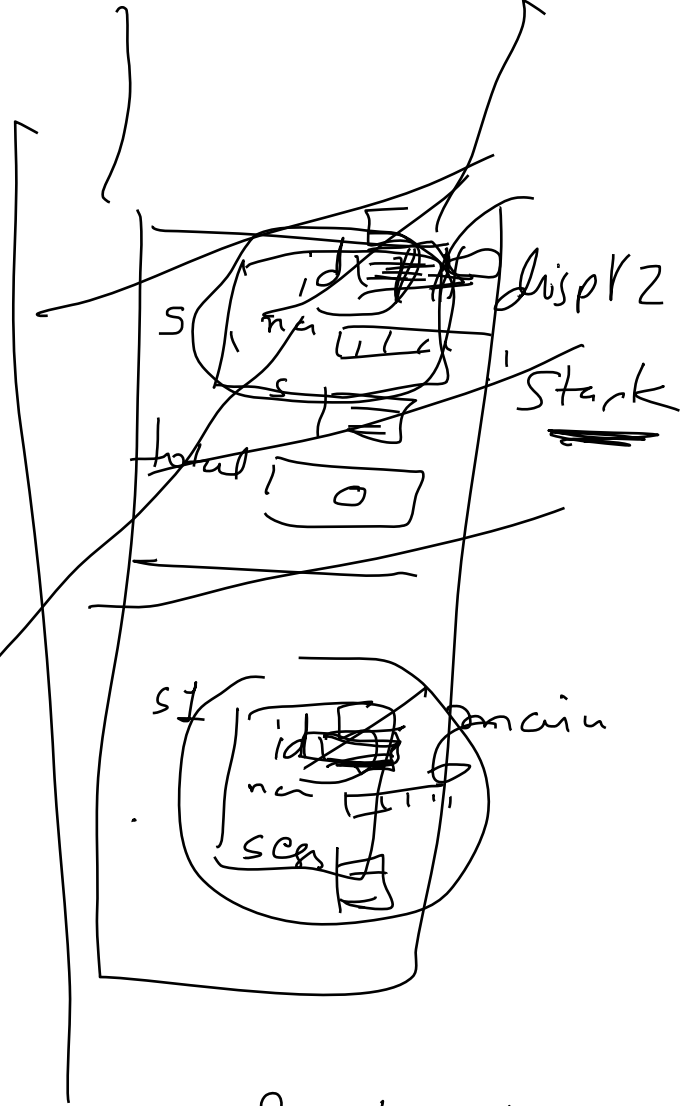
constant parts

int factorial(n);

, s1.sessionals

```
void displayV2(struct student );

void displayV2(stmet student s)
{
    int total=0;
    //total= s.sess[0]+ s.sess[1]
                    + s.sess[2];
    for(mj=0; mj<3; mj++)
    {
        total+= s.sessmod[mj];
    }
    s.id++; //9
}
```



Structure variable are literally copied. Call by value. Pass by value. copy and paste.

Returning structure variable from a function

```
struct student dosomething(struct student s)
{
    s.id++;
    return s;
}
int main()
{   struct student s1 = {10 ____ };s2;
    Before ____ 10
    s2 = dosomething(s1);
    After ____ 11
}
```

Data Stack