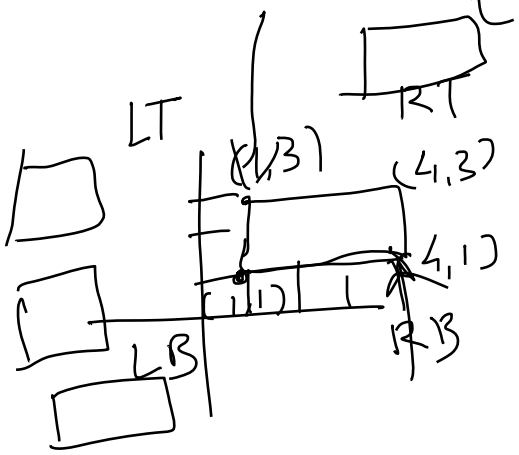
Let’s solve it

app ~ ex

```
struct point
{
    int x, y;
};
```



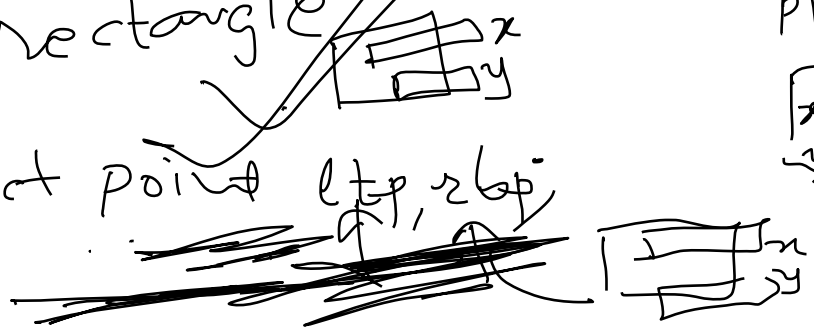

```
int main()
{ struct point result;
  struct point P1, P2;
  printf("\n Enter point 1 ");
  scanf("%d %d", &P1.x, &P1.y);
  printf("\n Enter P-2");
  scanf("%d %d", &P2.x, &P2.y);
  result.x = P1.x + P2.x;
  result.y = P1.y + P2.y;
  printf("%d %d", result.x, result.y);
```

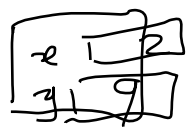
variable

```
struct rectangle
{
    int ltx, lty;
    int rbx, rby;
};
```

① ② ③

```
struct rectangle
{
    struct point ltp, rbp;
};
```





datatype

struct rectangle

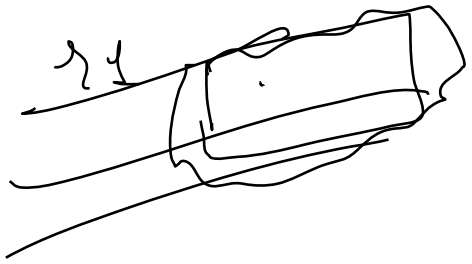
{

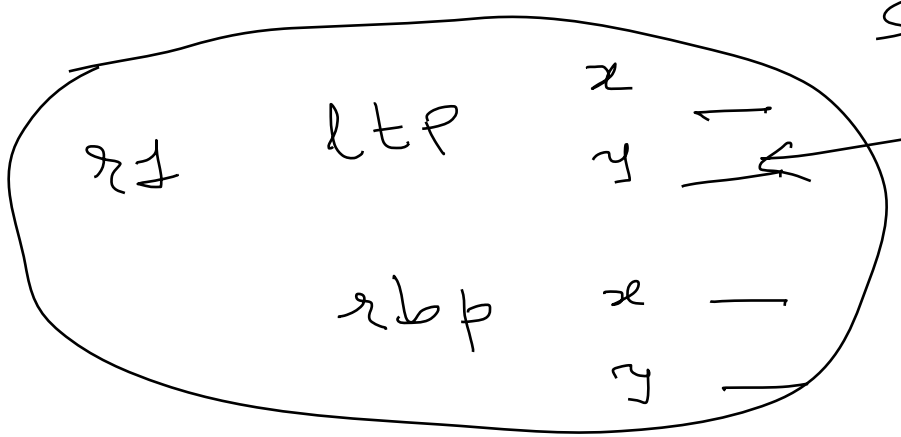
int l1, l2;

};

r1, r2, r3, r4, r5;

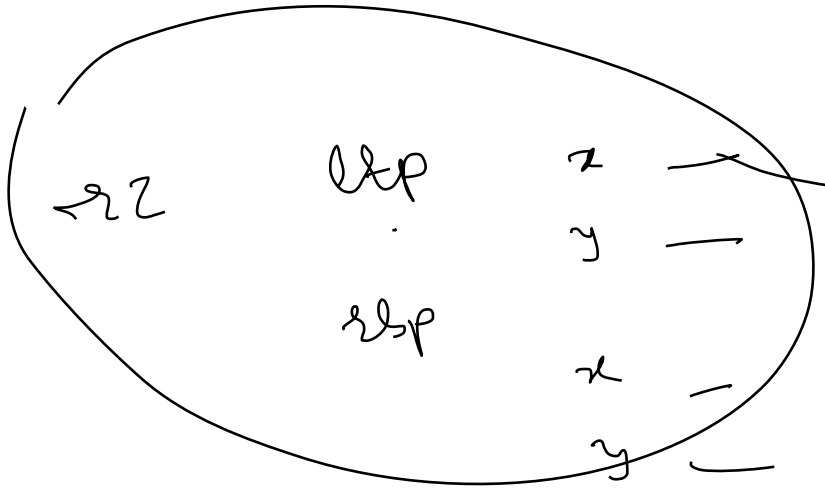
variables





scanf("%d", &r1.ltp.y);

structure variable within structure



printf("%d", r2.ltp.x);

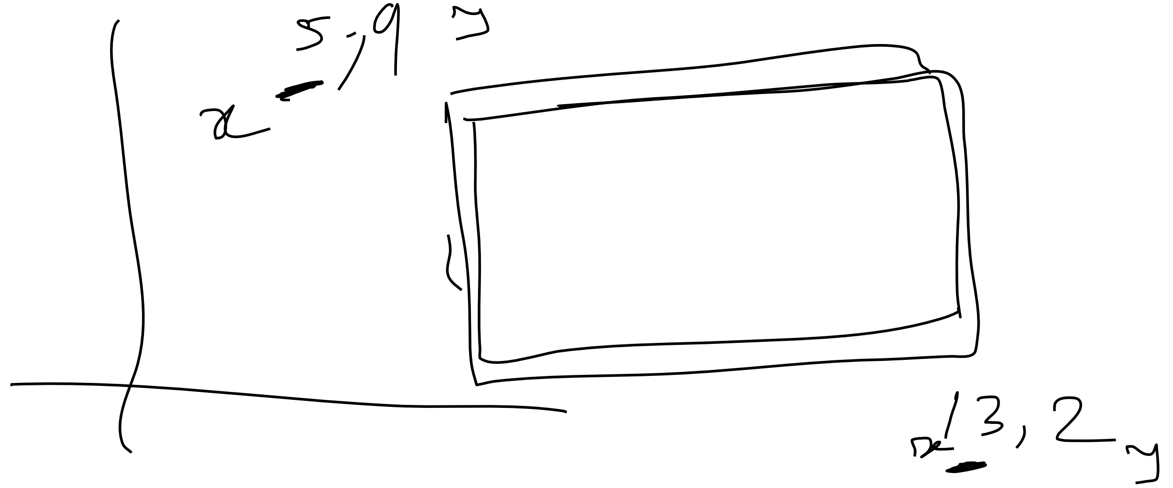
approach (B)

```
struct rectangle
{
    struct point
    {
        int x;
        int y;
    } ltp, rbp;
};
```

```
int main()
{
    struct rectangle r;
    p = r.ltp.x;
    s = r.rbp.y;
    struct point p1;   X
}
```

Hidden

Embedded/nested



$a \le y \le z$

a <= y && y <= z

(6, 7)

```
struct rectangle r;
    r.ltp.x = 5;
    r.ltp.y = 9;
    r.rbp.x = 13;
    r.rbp.y = 2;

struct point p;
    p.x = 6;
    p.y = 7;

if ((r.ltp.x <= p.x && p.x <= r.rbp.x) &&
    (r.rbp.y <= p.y && p.y <= r.ltp.y))
{
    printf("Within");
}
else {
    printf("Not within");
}
```

5 6 6 13

Structure Initialization

```
struct student s1 = { 1, "James", 99 };
```

id name marks

```
struct student
{
    int id;
    char name[10];
    int marks;
};
```

Array of structures

```
struct rectangle r1, r2, ..., r10;

struct rectangle r[10] = { {...}, ... };
```

Assignment of struct variables

Sl__ rect __ r1, r2;

r1.____ = __;

r1 = __;

~~//~~ r2 = r1 ; ?

r2.ltp.x = r1.ltp.x;

⋮

Copies the whole ~~block~~ memory area