

Computer Science & IT

C programming



Array & Pointers

Lecture No. 01



By- Abhishek Sir



Recap of Previous Lecture



Topic

Recursion

Topic

Topic

Topic

Topic

Topics to be Covered



Topic

pointers

Topic

Double pointers

Topic

Call by reference

Topic

Topic



Question



```
int fib(unsigned int n){  
    if (n <= 1)  
        return n;  
    return fib(n-1) + fib(n-2);  
}
```

The total number of times fib function
invoked in calls in fib (8) is _____



Question



Consider the following two functions.

```
void fun1 (int n)
{
    if (n == 0 ) return;
    printf ("%d" , n);
    fun2 (n - 2);
    printf ("%d" , n);
}
```

```
void fun2 (int n)
{
    if (n == 0) return ;
    printf ("%d" , n);
    fun1(++n) ;
    printf ("%d" , n);
}
```

The output printed when `fun1 (5)` is called is

(A) 53423122233445 ✓

(B) 53423120112233 ✗

(C) 53423122132435 ✓

(D) 53423120213243

[A]



Question

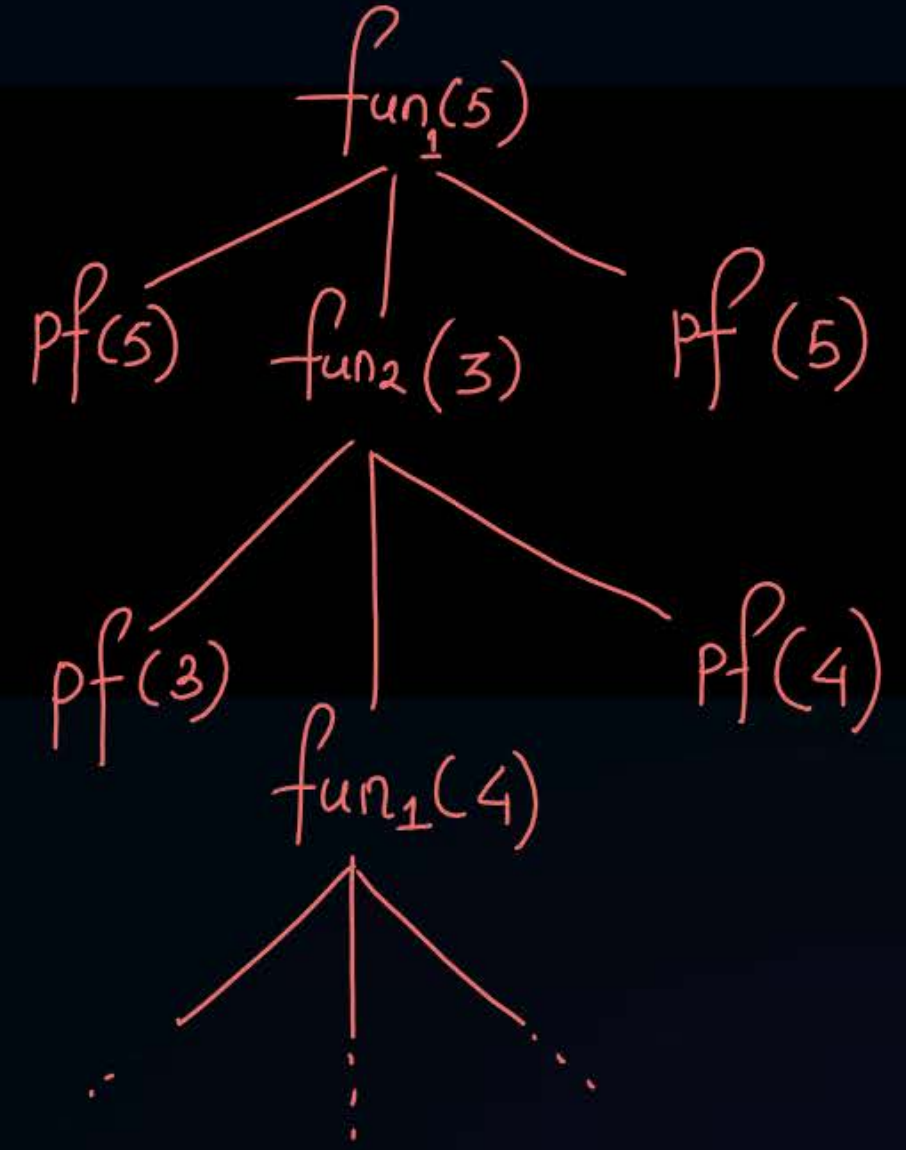


Consider the following two functions.



```
void fun1 (int n)
{
    if (n == 0 ) return;
    printf ("%d" , n);
    fun2 (n - 2);
    printf ("%d" , n);
}
```

```
void fun2 (int n)
{
    if (n == 0) return ;
    printf ("%d" , n);
    fun1(++n) ;
    printf ("%d" , n);
}
```





Question



Consider the following C function.

```
int fun(int n) {  
    int x = 1, k; 1 Mark  
  
    if (n == 1) return x;  
    for (k = 1; k < n; ++k)  
        x = x + fun(k) * fun(n - k);  
  
    Return x;  
}
```

The return value of fun (5) is _____

value calculation Bottom up calculation

Smaller to Larger value

$$\text{fun}(1) = 1$$

$$\begin{aligned} \text{fun}(2) \quad k=1 \quad & 1 + \text{fun}(1) * \text{fun}(1) \\ & 1 + 1 * 1 = 2 \end{aligned}$$

$$\begin{aligned} \text{fun}(3) \quad k=1 \quad & 1 + \text{fun}(1) * \text{fun}(2) \\ & 1 + 1 * 2 = 3 \end{aligned}$$

$$\begin{aligned} k=2 \quad & 3 + \text{fun}(2) * \text{fun}(1) \\ & 3 + 2 * 1 = \textcircled{5} \end{aligned}$$



Question

Consider the following C function.

```
int fun(int n) {  
    int x = 1, k;  
    if (n == 1) return x;  
    for (k = 1; k < n; ++k)  
        x = x + fun(k) * fun(n - k);  
    Return x;  
}
```

The return value of fun (5) is _____

$$\begin{aligned} \text{fun}(4) = & \quad k=1 \quad 1 + \text{fun}(1) * \text{fun}(3) \\ & \quad \quad \quad 1 + 1 * 5 = 6 \\ & \quad k=2 \quad \hookrightarrow 6 + \text{fun}(2) * \text{fun}(2) \\ & \quad \quad \quad 6 + 2 * 2 = 10 \\ & \quad k=3 \quad 10 + \text{fun}(3) * \text{fun}(1) \\ & \quad \quad \quad 10 + 5 * 1 = 15 \end{aligned}$$

$$\begin{aligned} \text{fun}(5) = & \quad k=1 \quad 1 + \text{fun}(1) * \text{fun}(4) \\ & \quad \quad \quad 1 + 1 * 15 = 16 \\ & \quad k=2 \quad 16 + \text{fun}(2) * \text{fun}(3) \\ & \quad \quad \quad 16 + 2 * 5 = 26 \\ & \quad k=3 \quad 26 + \text{fun}(3) * \text{fun}(2) \\ & \quad \quad \quad 26 + 5 * 2 = 36 \\ & \quad k=4 \quad 36 + \text{fun}(4) * \text{fun}(1) \\ & \quad \quad \quad 36 + 15 * 1 = 51 \end{aligned}$$



Question

Consider the following recursive function

$$\text{fun}(n) = \begin{cases} 2 & \text{if } n = 0 \\ x & \text{if } n = 1 \\ 2 \text{ fun}(n-1) + 4 \text{ fun}(n-2) & \text{if } n \geq 2 \end{cases}$$

H/W

If the value of $\text{fun}(4)$ is 88, then the value of 'x' is

(A) -1

(B) 0

(C) 2

(D) 1

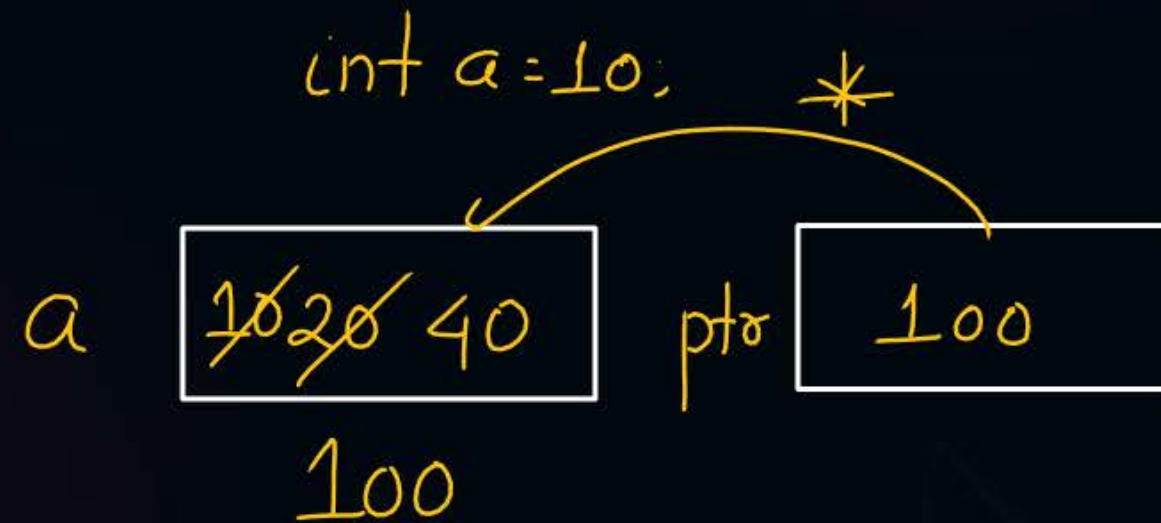


Pointers



pointer is a variable which stores address of another variable of

Same.



```
int *ptr,  
int *ptr = &a,  
ptr = &a;
```

variable a can be updated in two ways

$a = a + 10$

$*ptr = *ptr + 20;$
 $20 + 20$



Pointers



& Address of

45

constant

* dereference operator

expression

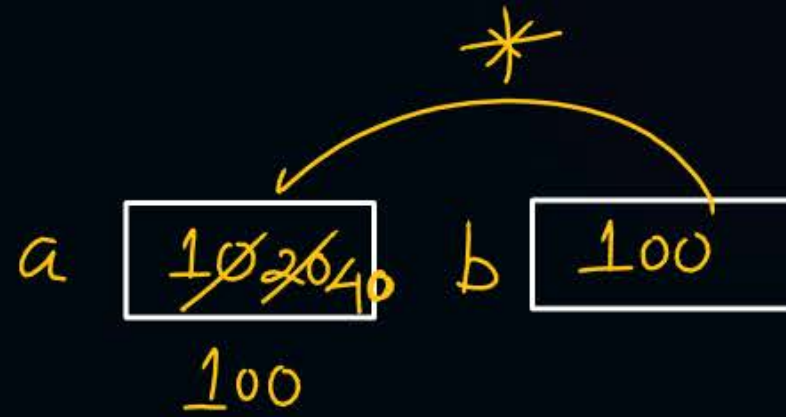
No Memory allocation



Pointers



```
#include<stdio.h>
int main (){
    int a, *b;
    a = 10;
    b = &a;
    a =a+10;
    *b = *b+20;
    printf("%d", a);
    return 0;
}
```

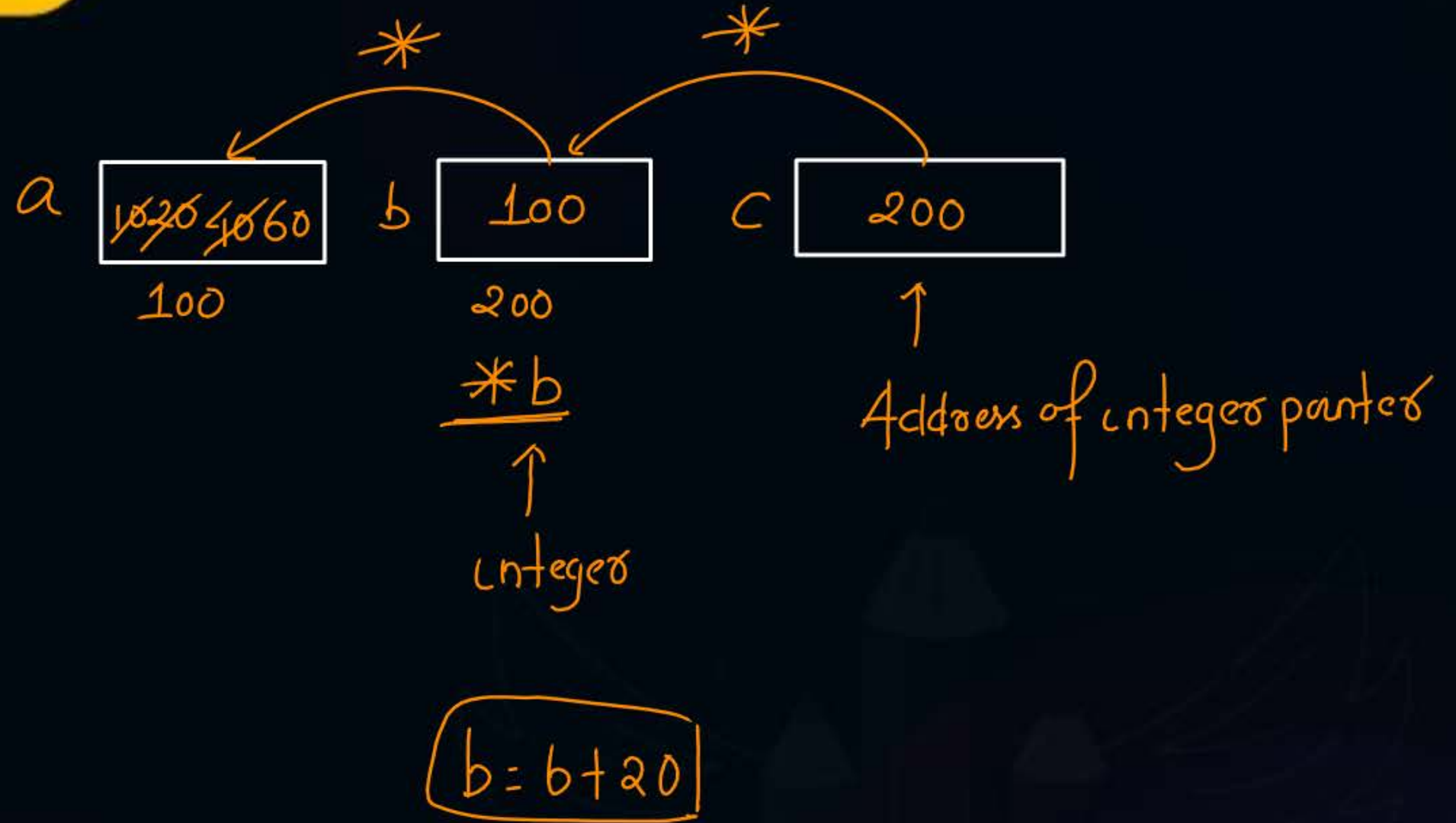




Pointers



```
#include<stdio.h>
int main (){
    int a, *b, **c;
    a = 10;
    b = &a;
    c = &b;
    a = a+10; ✓
    *b = *b+20; ✓
    **c = **c+20;
    printf("%d", a);
    return 0;
}
```





Size of Pointer



```
include <stdio.h>
```

```
int main() {  
    int *p;  
    char *p1;  
    float *p2;  
  
    printf("%lu", sizeof(p));  
    printf("%lu", sizeof(p1));  
    printf("%lu", sizeof(p2));  
  
    return 0;  
}
```

Size of pointer variable

int = 4B

*int = 8B

char = 1B

*char = 8B

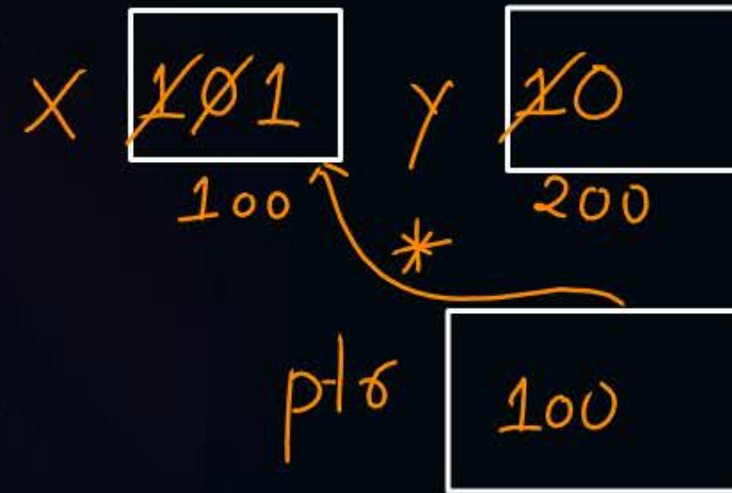
float = 4B

*float = 8B



#Q Consider the following function implemented in C:

```
void printxy (int x, int y) {  
    int *ptr ;  
    x = 0;  
    ptr = &x;  
    y = * ptr;  
    * ptr = 1;  
    printf ("%d, %d," x, y);  
}
```



The output of invoking printxy (1, 1) is

- (A) 0, 0
- (B) 0, 1
- (C) 1, 0
- (D) 1, 1

Handwritten notes:

$y = *ptr$
 $y = 0$



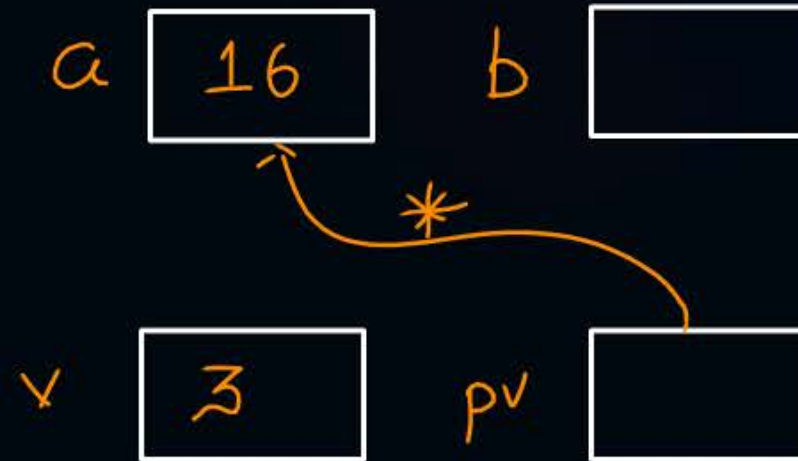
Question

#Q Find the output of the following program.

```
#include <stdio.h>

int main () {
    int a, b;
    int v=3;
    int *pv = &v;
    a = 2 * (*pv+5);
    pv = &a; ✓
    b = 2 * (*pv+5);
    printf ("a=%d b=%d", a,b);
}
```

- (A) a = 16 b = 16 (B) a = 16 b = 42 ✓
(C) a = 16 b = 8 (D) a = 16 b = 64



$$a = 2 * (3 + 5) = 16$$

$$b = 2 * (16 + 5)$$

$$= 2 * 21 = 42$$



Question



#Q Find the output of the following program. 130

```
#include <stdio.h>
```

```
int main () {
```

```
    int a, b;
```

```
    int v=3;
```

```
    int *ptr1 = &v;
```

```
    int **ptr2 = &ptr1;
```

```
    a = 2 * (*ptr1+5);
```

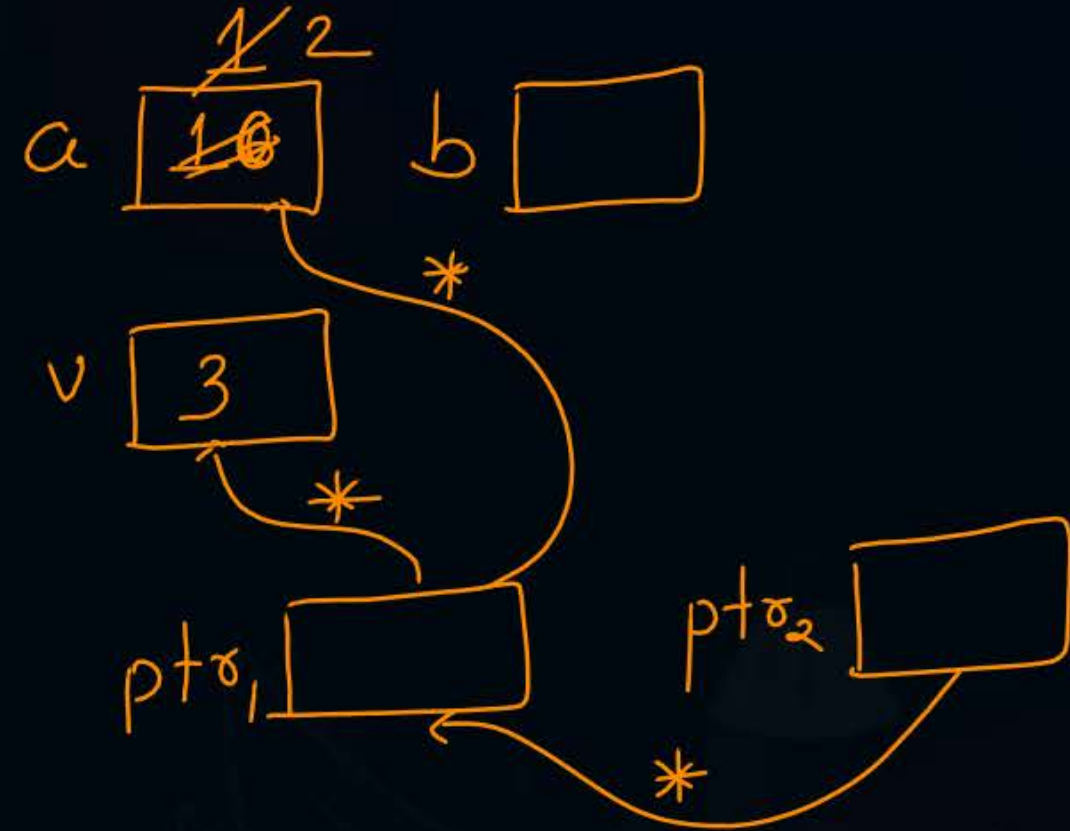
```
    a = **ptr2>>1;
```

```
    ptr1 = &a;
```

```
    (**ptr2)++;
```

```
    b= 2*(*ptr1<<5);
```

```
    printf ("%d", a+b);
```



$$a : 2 * 8 = 16$$

$$b = 2 * (2 \times 2^5) \quad a = 3 \gg 1 = \lfloor \frac{3}{2} \rfloor = 1$$

$$b = 2 * (2 \times 2^5) = 128 + 2 = 130 \quad b = 128$$



Question

#Q What does the following program print?

```
#include<stdio.h>
```

```
void f(int *p, int *q) {
```

```
    p=q;
```

```
    *p=2;
```

```
}
```

```
int i=0, j=1;
```

```
int main() {
```

```
    f(&i, &j);
```

```
    printf("%d %d\n", i,j);
```

```
    return 0;
```



(A) 2, 2

(B) 2, 1

(C) 0, 1

(D) 0, 2





Question



The value printed by the following program is _____.

```
void f (int * p, int m) {  
    m = m + 5;  
    *p = *p + m;  
    return;  
}  
void main () {  
    int i=5, j=10;  
  
    f (&i, j);  
    printf ("%d", i+j);  
}
```

Diagram illustrating memory state and pointer manipulation:

- Variable `p` points to a memory box containing `100`.
- Variable `m` points to a memory box containing `10` (initially `15`, which is crossed out).
- Variable `i` points to a memory box containing `5` (initially `20`, which is crossed out).
- Variable `j` points to a memory box containing `10`.
- Handwritten calculation: $*p = 5 + 15 = 20$.
- Handwritten calculation: $20 + 10 = 30$.



Question



What is printed by the following C program?

```
int f(int x, int *py, int **ppz){
```

```
    int y, z;
```

```
    **ppz += 1; z = **ppz; ✓ z=5
```

```
    *py += 2; y = *py; ✓ y=7
```

```
    x += 3; ✓ x=7
```

```
    return x+y+z;
```

```
}
```

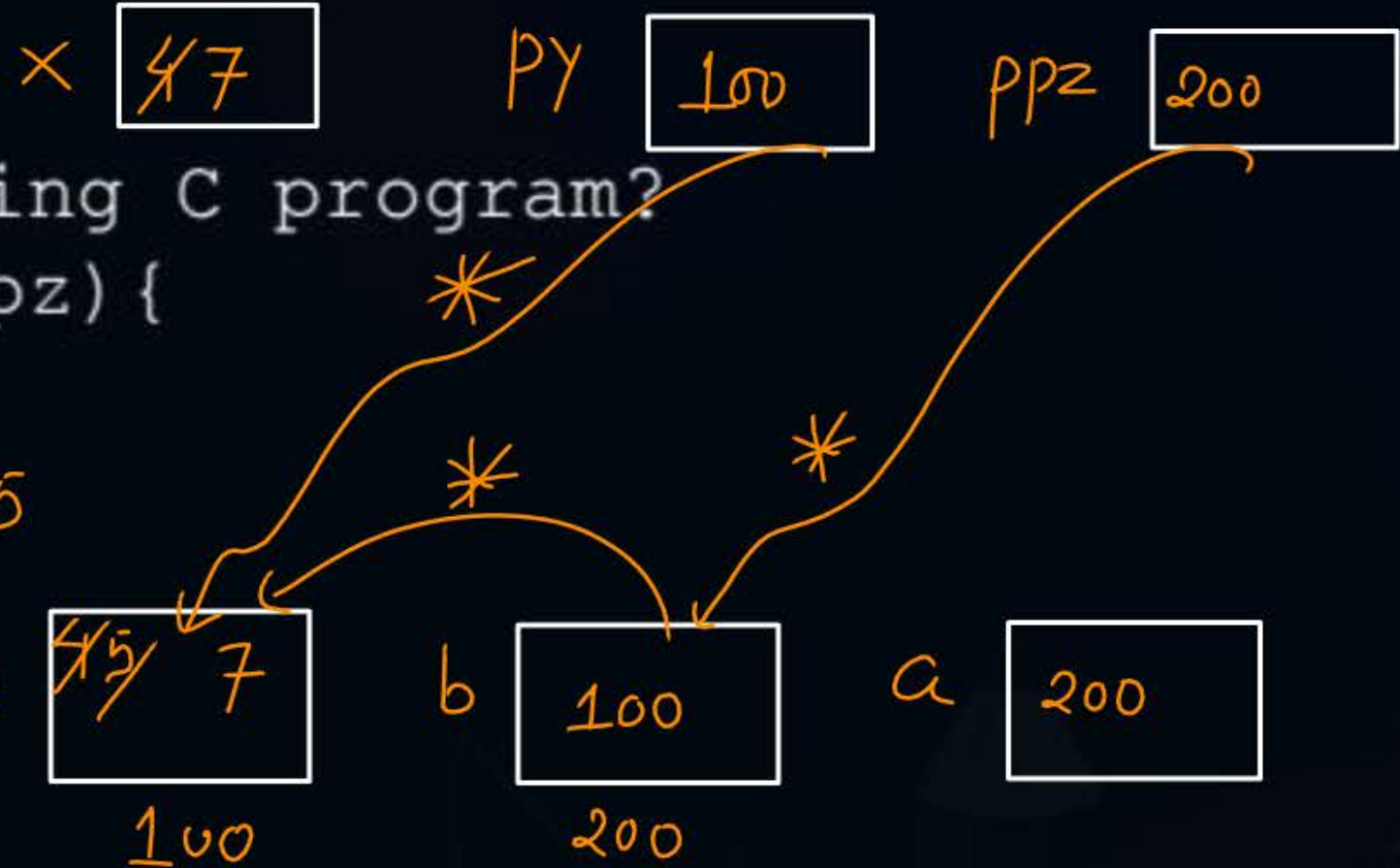
7+7+5 = 19

```
void main(){
```

```
    int c, *b, **a;
```

```
    c = 4; b = &c; a = &b;
```

```
    printf("%d", f(c, b, a));
```





Question

5, 7, 6



#Q Consider the C program shown below.

```
#include <stdio.h>
```

```
#define print(x) printf("%d ", x) *
```

```
int x; ✓ z  $7+5=12$ 
```

```
void Q(int z) {
```

```
    z += x; ✓
```

```
    print(x);
```

```
}
```

```
void P(int *y) {
```

```
    int x = *y+2;
```

```
    Q(x);
```

```
    *y = x-1;
```

```
    print(x);
```

```
}
```

```
main(void) {
```

```
    x = 5; ✓
```

```
    P(&x)
```

```
    print(x);
```

```
}
```

The output of this program is

(A) 12 7 6

(B) 22 12 11

(C) 14 6 6

(D) 7 6 6

X_g $\boxed{5/6}$
100

$X_L - 1$

$7 - 1 = 6$

X_L

$\boxed{7}$

y

$\boxed{100}$

```
void swap1(int a, int b){
```

```
    int temp;
```

```
    temp= b;
```

```
    b= a;
```

```
    a= temp;
```

```
}
```

```
#include <stdio.h>
```

```
int main(){
```

```
    int x = 10, y=20;
```

```
    swap1(x,y);
```

```
    printf("%d %d", x, y);
```

```
    return 0;
```

```
}
```

a=10 b=20 pass by value

a 10

b 20

x 10

y 20

2 set of
variable

Independent

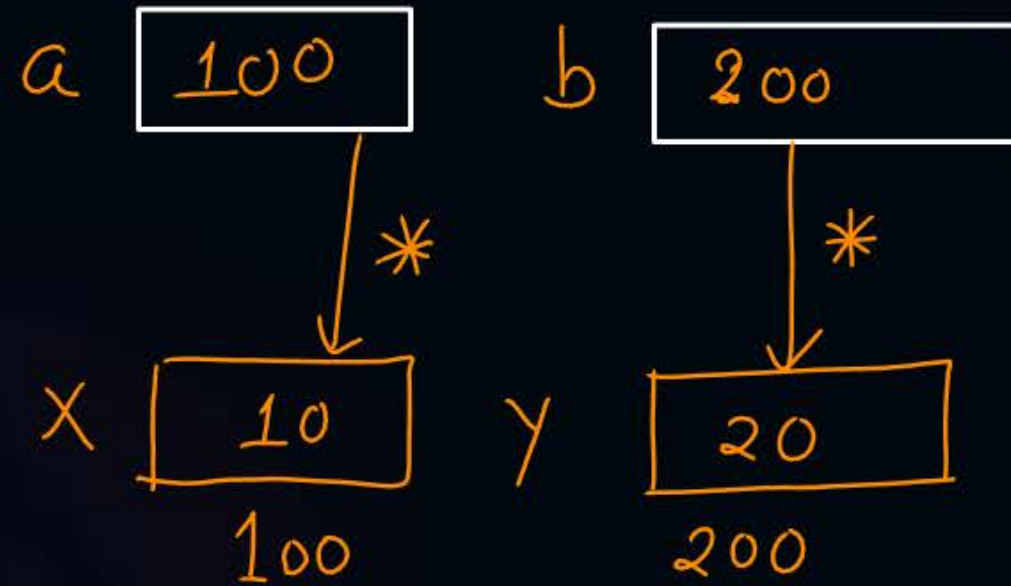
No connection

Call by reference

```
void swap1(int *a, int *b) {
    int temp;
    temp = *b;
    *b = *a;
    *a = temp;
}
```

```
#include <stdio.h>
```

```
int main() {
    int x = 10, y = 20;
    swap1(&x, &y);
    printf("%d %d", x, y);
    return 0;
}
```



Connection
between variable



2 mins Summary



Topic

pantes

Topic

Double pantes

Topic

Call by reference

Topic

Topic

Aray

THANK - YOU

