CS & IT ENGINEERING

Programming in C

Arrays and Pointers

Lec- 06

By- Pankaj Sharma sir

- Arrays and Pointers

1) int (*P) [4]

P is a pointer to array of 4 integer.

2) int*(*P)[5]

P is a pointer to (array of 5 pointer to integer)

3) int *P[4]

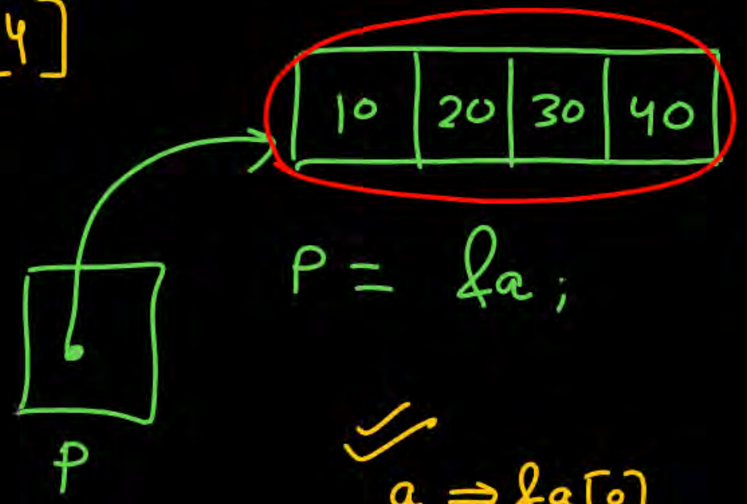
P is an array of 4 pointer to integer

4) int (*P)()

P is a pointer to function that takes no argument and returns an integer.

int a[4] = {10,20,30,40};

1) int (*p)[4]



p = &a;

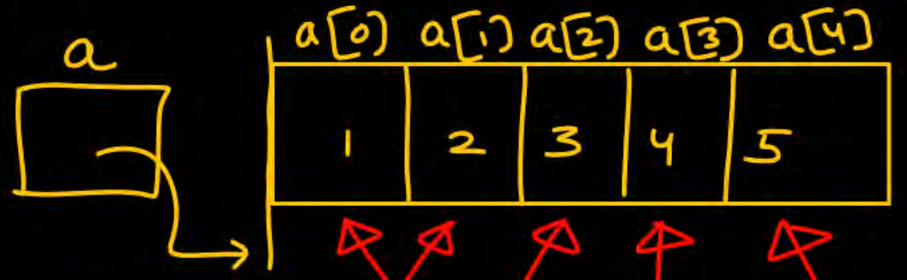
a ⇒ &a[0]
a+1 ⇒ &a[1]
a+2 ⇒ &a[2]
a+3 ⇒ &a[3]
a+4 ⇒ &a[4]

2) int*(*p)[5]

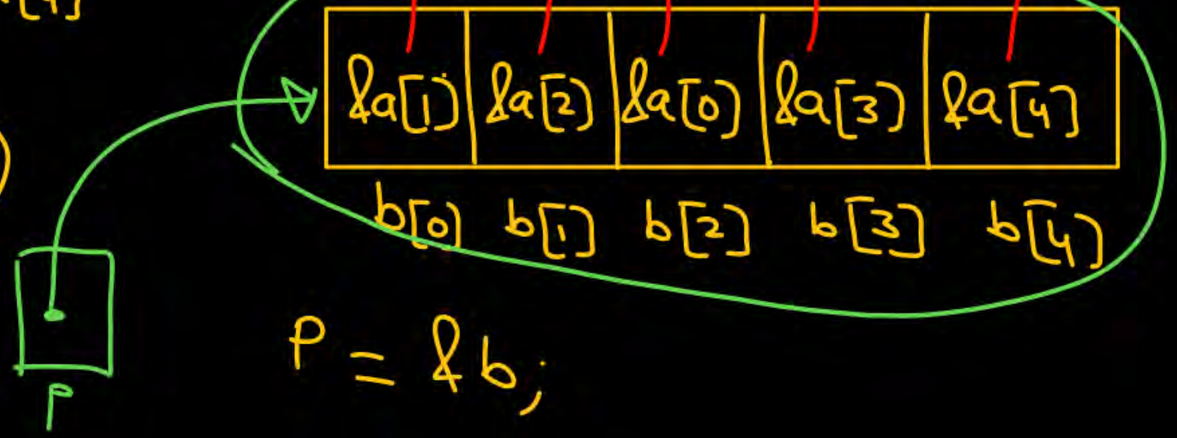
p is a pointer to

array of 5 pointer to integer

int a[5] = {1,2,3,4,5};



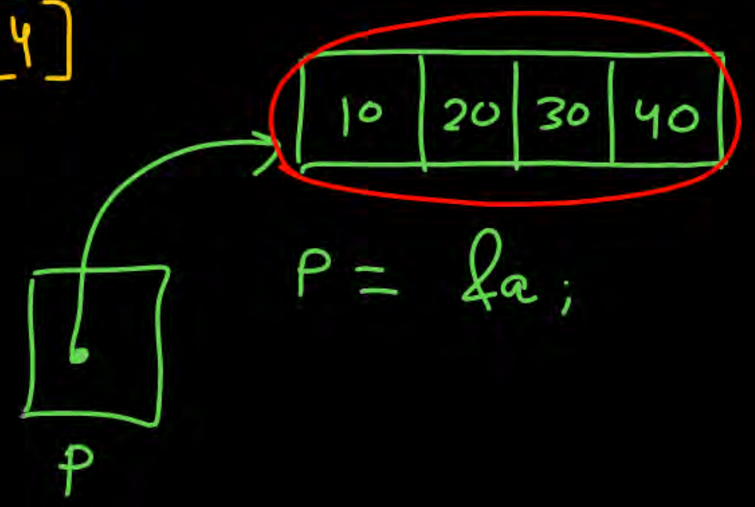
int* b[5] = {a+1, a+2, a, a+3, a+4};



p = &b;

int a[4]={10,20,30,40};

1) int (*p)[4]



p = &a;

3) int *p[4]

2) int*(*p)[5]

4) int (*p)()

function pointer

(int, float) ( ) → funct intAdd(int, int)

5) int (*P)(int, float)

P is a pointer to function that takes 2 arguments, first is integer second is float and it returns a integer value.

6) int (*P)(char *)

arg

P is a pointer to function → that takes a pointer to char as argument and returns integer.

7) int (*f)(int*)

f is a pointer to function that take a pointer to integer as argument and returns integer value.

function pointer
(pointer to function)

Q) Can we pass value to a function

```
int a=10,b=20,result;
result = Add(a,b);
```

valid ✓

Q) Can we pass an address to a function

```
int a[4]={10,20,30,40}; fun(a);
```

```
#include<stdio.h>
 void fun( ){
        printf("Hello");
        }
 void main(){
        fun();
        fun();
        }
```

address

```
void
 fun(){

        }
```

```
main(){

        }
```

$$e^x = 1 + \frac{x}{1!} + \frac{x^2}{2!} + \cdots$$

factorial

int a = 10; float b = 10.38

int *p; float *Ptr;

2 1

int Add (int, int);

P is a pointer to

a function that takes 2 integer arguments and return integer.

int (*p) (int, int);

```
int Add(int, int);
```

```
int (*P)(int, int);
```

```
Add(10,20);
```

Equivalent

```
P = Add;
(*P)(10,20);
```

```
#include<stdio.h>
int Add(int a, int b){
        return a+b;
        }
int sub(int a, int b){
        return a-b;
        }
int Prod(int a, int b){
        return a×b;
        }
```

```
void main(){
    int (*P)(int, int);
    P = Add;
    printf("%d", (*P)(10,20));
    P = sub;
    printf("%d", (*P)(10,20));
    P = Prod;
    printf("%d", (*P)(10,20));
    }
```

```
int a[4] = {10, 20, 30, 40};

int *P[4] = {a+3, a+2, a+1, a};

int y;

y = --P[0] - P[1];

printf("%d", y);

printf("%d", *P[0]);
```

Pre-dec

$\&a[3]$ $\&a[1]$ $\&a[2]$ $\&a[0]$

a[0]	a[1]	a[2]	a[3]
10	20	30	40
100	104	108	

$\&a[2]$

P[0]	P[1]	P[2]	P[3]
~~$\&a[3]$~~	$\&a[2]$	$\&a[1]$	$\&a[0]$
100	104		

(i) $P[0] = P[0] - 1$

$y = P[0] - P[1]$

$\rightarrow 0$

$\rightarrow 30$

$P[0] = \&a[3] - 1$

$P[0] = \&a[2]$

$y = \&a[2] - \&a[2]$

$= \frac{108 - 108}{4} = \frac{0}{4} = 0$

$*P[0]$

$\cancel{*}\cancel{\&}a[2] \Rightarrow a[2]$

```
int a=5, b=10, c=15;
int *p[3]={&a, &b, &c};
printf("%d", *p[*p[1]-8]);
```

15

$p[1] \equiv \&b$

$*p[\cancel{*}\&b-8] \Rightarrow \cancel{*}\&c$

$*p[b-8]$

$*p[10-8] \Rightarrow *p[2]$

$\Rightarrow c$

```
static int a[] = {10, 20, 30, 40, 50};
static int *P[] = {a, a+3, a+4, a+1, a+2};
int **Ptr = P;
Ptr++;
printf("%d %d", Ptr-P, **Ptr);
```

&a[0]	&a[3]	&a[4]	&a[1]	&a[2]
P[0]	P[1]	P[2]	P[3]	P[4]
100	104	108	112	116

Ptr: ~~&P[0]~~ &P[1]

(i) Ptr − P ⇒ &P[1] − &P[0] = (104 − 100)/4 = 4/4 = 1

(ii) **Ptr ⇒ **&P[1] ⇒ *P[1] ⇒ *&a[3] ⇒ a[3]

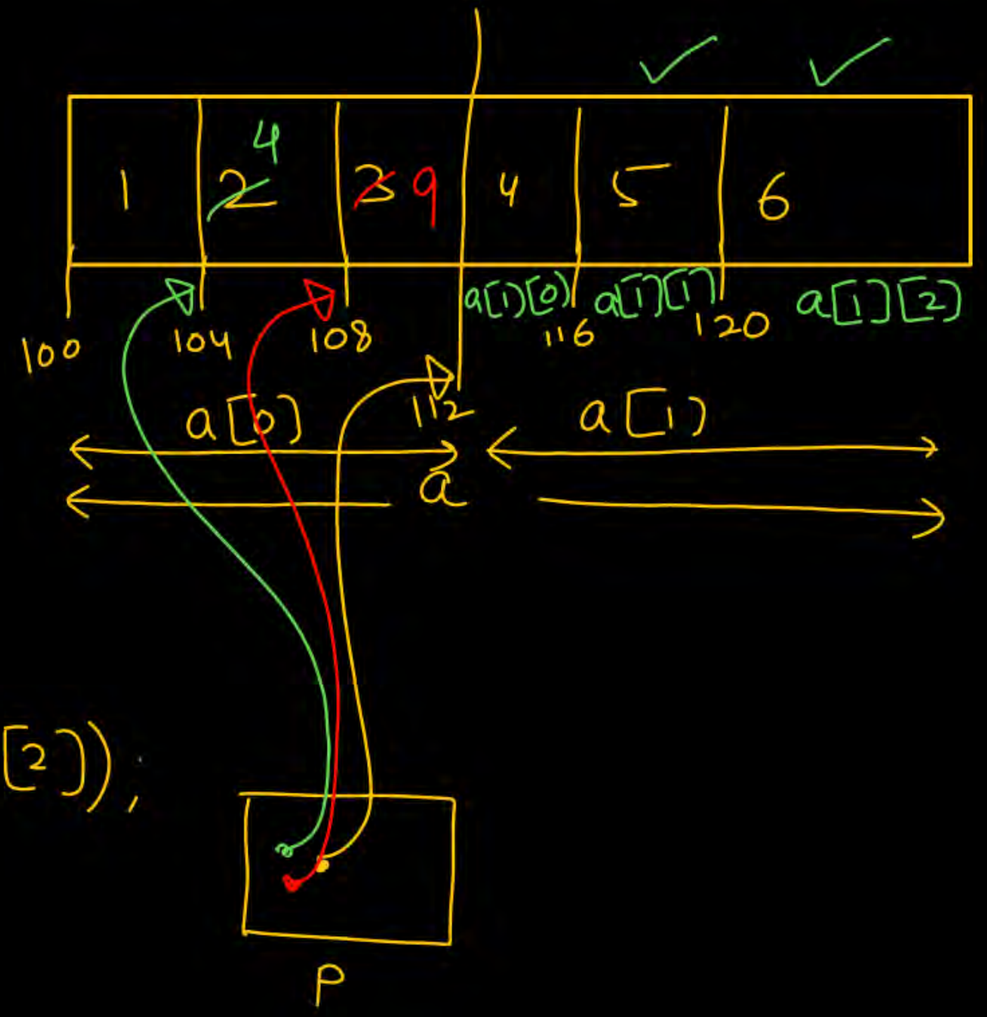
Q

```
void f(int *);

int main(){

    int a[2][3] = {1,2,3,4,5,6};

    f(a[1]);

    printf("%d %d", a[1][1], a[1][2]);
}

void f(int *p){

    p--;
    *p = *p * *p;
    p--;
    *p = *p * *p;
}
```

$\&a[1][0]$

$*p = 3 \times 3$

$*p = 2 \times 2$



```
int a[4][5] = { { 1,2,3,4,5},
                {6,7,8,9,10},
                {11,12,13,14,15},
                {16,17,18,19,20} };
```

```
printf("%d\n", *(*(a + **a + 2)+3));
```

19

$*(*(a + 1 + 2)+3)$

$*(*(a+3)+3)$

$*(a[3]+3) \Rightarrow a[3][3]$

① $a[i] = *(a+i)$

$*(a+3) = a[3]$

② $*(a[i]+j) \Rightarrow a[i][j]$

$\cancel{**\&a[0]}$

$*a[0]$

$\cancel{*\&a[0][0]}$

$a[0][0]$

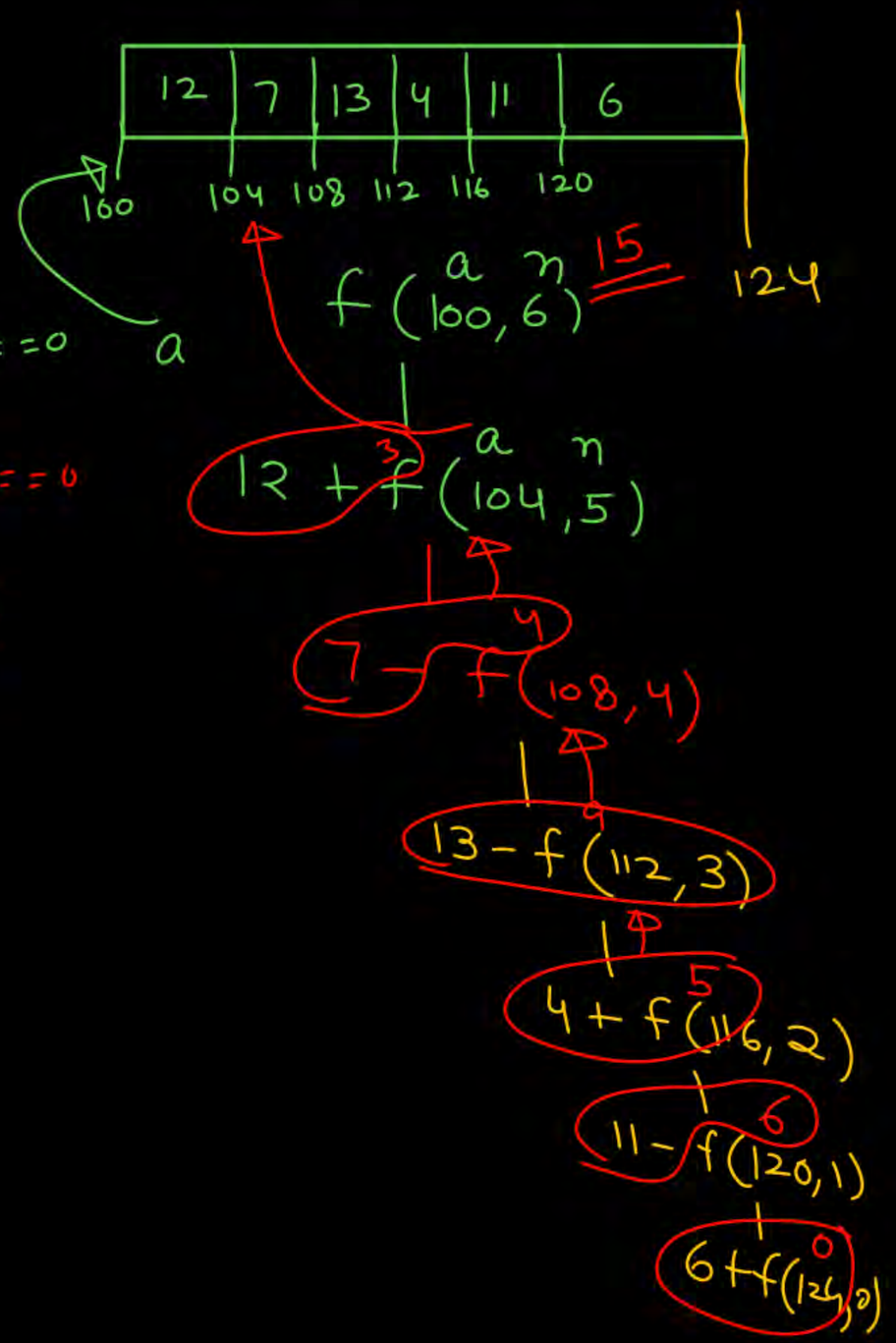
```
int f(int *a, int n){
    if (n<=0) return 0;
    else if(*a % 2==0)
        return *a + f(a+1,n-1);
    else
        return *a - f(a+1,n-1);
}

void main(){
    int a[]={12,7,13,4,11,6};
    printf("%d", f(a,6));
}
```

(i) 12 % 2 == 0

even (ii) 7 % 2 == 0

odd

15



```
int f(int *P, int n)
{
    if(n<=1) return 0;
    else
        return max(f(P+1,n-1), P[0]-P[1]);
}

void main(){
    int a[]={3,5,2,6,4};
    printf("%d", f(a,5));
}
```

$f(100,5)$

$P[0] = *(P+0) = 3$

$P[1] = *(P+1) = 5$

3	5	2	6	4
100	104	108	112	115

$f(\overset{P}{100}, \overset{n}{5})$ 3

$\max(f(\overset{P}{104}, \overset{n}{4}), 3-5)$ 3, -2

$\max(f(\overset{P}{108}, \overset{n}{3}), 5-2)$ (2, 3)

$\max(f(\overset{P}{112}, \overset{n}{2}), 2-6)$ 2, -4

$\max(\cancel{f(\overset{P}{116}, \overset{n}{1})}, 6-4)$ 0, 2


THANK YOU GW
SOLDIERS !