CS & IT ENGINEERING

C Programming

Arrays and Pointers

Lec - 09

By- Pankaj Sharma Sir

Arrays and Pointers (Part- 09)

2015 Q1.

2000 , 4 byte

```
unsigned int x[4][3] = { {1,2,3},{4,5,6},{7,8,9}, {10,11,12}};

printf("%u %u %u", x+3, *(x+3), *(x+2)+3);
```

A) 2036, 2036, 2036

B) 2012, 4, 2024

C) 2036, 10, 10

D) 2012, 4, 6

BA - 1000, 4byte

2.

```
int a[2][3] = {1,2,3,4,5,6};
pf("%u %u %u", a, *a, **a);
pf("%u %u %u", a+1, *a+1, **a+1);
```

a → 1000, *a → 1000, **a → 1 → a[0][0]

**a ⇒ **&a[0]
= *a[0]
= *&a[0][0] = a[0][0]

1+1 = 2

*a+1
= *&a[0]+1
= a[0]+1
= &a[0][0]+1
= 1004 (4byte)

A) 1000 1000 1 1012 1004 5

~~B)~~ 1000 1000 1000 1012 1004 1004

~~C)~~ 1000 1000 1 1012 1008 5

D) None

Q3.

```
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]);
```

a[0]	a[1]	a[2]	a[3]
10	20	30	40
100	104	108	112

P[0]	P[1]	P[2]	P[3]
&a[2] ~~&a[3]~~	&a[2]	&a[1]	&a[0]

(i) $P[0] = P[0] - 1$
$= \&a[3] - 1$
$= \&a[2]$

(ii) $y = P[0] - P[1]$
$= \dfrac{\&a[2] - \&a[2]}{4} = \dfrac{0}{4} = 0$

$*P[0]$
$= *\&a[2]$
$= a[2] \Rightarrow 30$

A) 020
B) 010
C) 130
D) 030

2005

Q4 `int (*f)(int*);`

A) A func. that takes a int pointer as argument & returns an integer.

B) A " " " " int as arg. & return a int. pointer.

C) A pointer to a function that takes an int. pointer as arg. and returns an integer.

D) A function that takes an int. pointer as arg & returns a func. pointer.

Q/

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

&a	&b	&c
p[0]	p[1]	p[2]

A) Segmentation fault

B) Compilation Error

C) ✓ 15

D) 10

$$*p[*p[1]-8]$$
$$= *p[\cancel{*}\cancel{\&}b-8]$$
$$= *p[b-8]$$
$$= *p[10-8]$$
$$= *p[2] = \cancel{*}\cancel{\&}c = c$$

Gate - 2015

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

$\Rightarrow$ Ptr = &P[0]

	a[0]	a[1]	a[2]	a[3]	a[4]
a	10	20	30	40	50
	100	104	108	112	116

P	&a[0]	&a[3]	&a[4]	&a[1]	&a[2]
	P[0]	P[1]	P[2]	P[3]	P[4]
	200	204	208	212	216

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

Gate - 2015

```
void main(){
    static int a[] = {10,20,30,40,50};
    static int *p[] = {a, a+3, a+4, a+1, a+2};
    int **ptr = p;
    ptr++;
    pf("%d %d", ptr-p, **ptr);
}
```

$\Rightarrow$ ptr = &p[0]

	a[0]	a[1]	a[2]	a[3]	a[4]
a	10	20	30	40	50
	100	104	108	112	116

p	&a[0]	&a[3]	&a[4]	&a[1]	&a[2]
	p[0]	p[1]	p[2]	p[3]	p[4]
	200	204	208	212	216

ptr &~~p[0]~~ &p[1]

$$\frac{\&p[1] - \&p[0]}{4}$$

$$= \frac{204-200}{4} = \frac{4}{4} = 1$$

Gate - 2015

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

$\Rightarrow$ Ptr = &P[0]

	a[0]	a[1]	a[2]	a[3]	a[4]
a	10	20	30	40	50
	100	104	108	112	116

P	&a[0]	&a[3]	&a[4]	&a[1]	&a[2]
	P[0]	P[1]	P[2]	P[3]	P[4]
	200	204	208	212	216

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

$$\frac{\&P[1] - \&P[0]}{4} = \frac{204-200}{4} = \frac{4}{4} = 1$$

$$**Ptr = **\&P[1] = *P[1] = *\&a[3] = a[3] = 40$$

Gate 2003

```
#define print(x) printf("%d",x)
int x;
void Q(int z){ z = z+x;
                print(z);
               }
void P(int *y) { int x = *y+2;
                 Q(x);
                 *y = x-1;
                 print(x);
                }
Void main(){
            x = 5;
            P(&x);
            print(x);
           }
```

A)	12	7	6
B)	22	12	11
C)	14	6	6
D)	7	6	6

Gate 2003

```
#define print(x) printf("%d", x)
int x;
void Q(int z){ z = z + x;
                print(z);
}
void P(int *y){ int x = *y + 2;
                Q(x);
                *y = x - 1;
                print(x);
}
Void main(){
        x = 5;
        P(&x);
        print(x);
}
```

$\rightarrow$ global

$\Rightarrow z_Q = z_Q + x_g$

call by value $\Rightarrow Q(7)$

$P(1000)$

x_g	~~0~~ 5	1000
y_p	1000	
x_p	7	2000
z_Q	~~7~~ 12	

12

Gate 2003

```
#define print(x) printf("%d",x)
int x;
void Q(int z){ z = z + x;
                print(z);
}
void P(int *y) { int x = *y + 2;
                 Q(x);
                 *y = x - 1;
                 print(x);
}
void main(){
        x = 5;
        P(&x);
        print(x);
}
```

global

$\Rightarrow z_Q = z_Q + x_g$

local

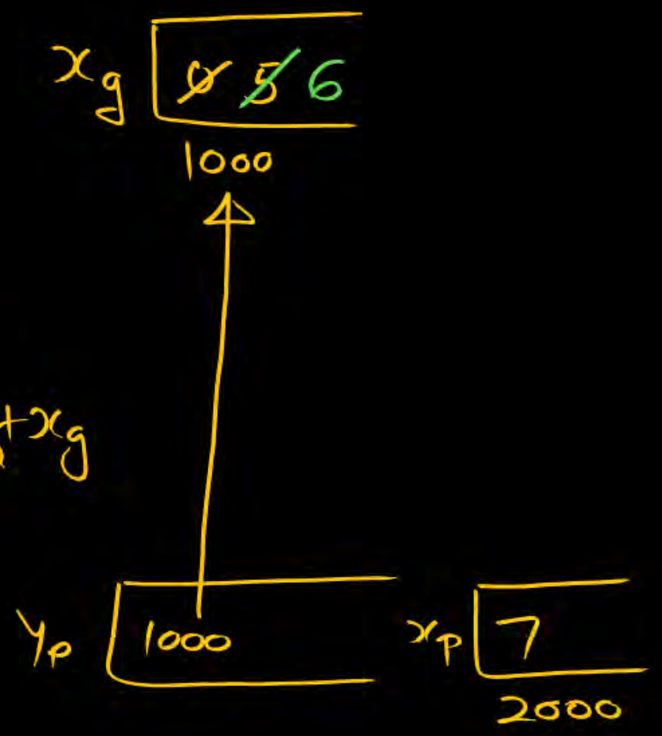
local

7

6

global

P(1000)



12 7 6

Gate 2003

```
int *A[10];
int B[10][10];
```

Of the following exp. which will not give C.T.E. if used as Lvalue of an assignment statement.

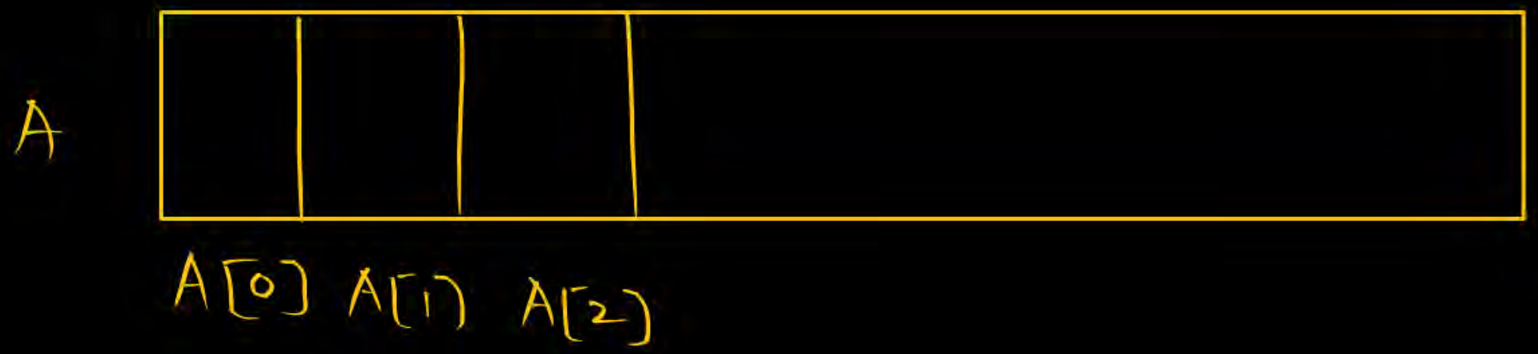
1) A[2] ✓
2) A[2][2] ✓
3) B[1] ✗
4) B[2][2] ✓

A) 1, 2 and 4
B) 2, 3 and 4
C) 2 and 4
D) 4 only

Gate 2003

```
int *A[10];
int B[10][10];
```

1) A[2]
2) A[2][2]
3) B[1]
4) B[2][2]



A[2] = &a;

Gate 2003

```
int *A[10];
int B[10][10];
```

A[2] + 2

*(A[2]+2) =

1) A[2]
2) A[2][2]
3) B[1]
4) B[2][2]

A[2] = &a;

B[10][10]

Address

B[1]

Address =

B[1][1] → element

✓ B[1][1] = 2;

Gate 2020

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

$**a = a[0][0]$

```
pf("%d", *( *(a + **a + 2) + 3));
```

$\downarrow$

1

Gate 2020

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

$**a = a[0][0]$

```
pf("%d", *( *(a + **a + 2) + 3));
```

$\downarrow$

1

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

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

19

Q/

```
int a[4] = {10, 20, 30, 40};
int *p;
p = (int*)(&a+1);
pf("%d %d", *(a+1), *(p-1));
```

$\&a[0]+1 = \&a[1]$ → 4 byte

$\&a+1 \times 16 \Rightarrow 116$

*(a+1) → a[1] → 20

*(p-1) → 40

a[0]	a[1]	a[2]	a[3]
10	20	30	40
100	104	108	112

116

p-1

p

*(p-1)

28 lecture

PYQs

09 : 00 PM

strings – 1
strings – 2
structure & union
Last class –

4 more lecture


THANK YOU GW
SOLDIERS !