CS & IT ENGINEERING

Programming in C

Arrays and Pointer

DPP 03 Discussion Notes

By- Pankaj Sharma sir

01 Question

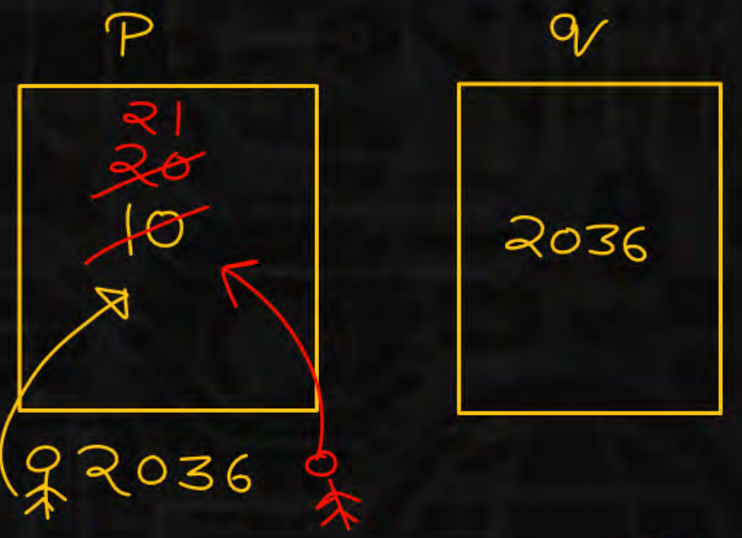
02 Discussion

Q.1

[NAT]

Consider the following program:

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



*q = (P++) + (*q)

(i) *q = P + *q

(ii) P = P + 1

*q = 10 + 10 => *q = 20

The output is 21.

Q.2

[MCQ]

Consider the following program:

```
#include<stdio.h>
int * f(){
static int a[4]={1, 2, 3, 4};
return a;
}
int main(){
 int *p, i;
 p=f();
 for(i=0;i<3;i++){
printf("%d\t", p[i]+p[i+1]);
 }
return 0;
}
```

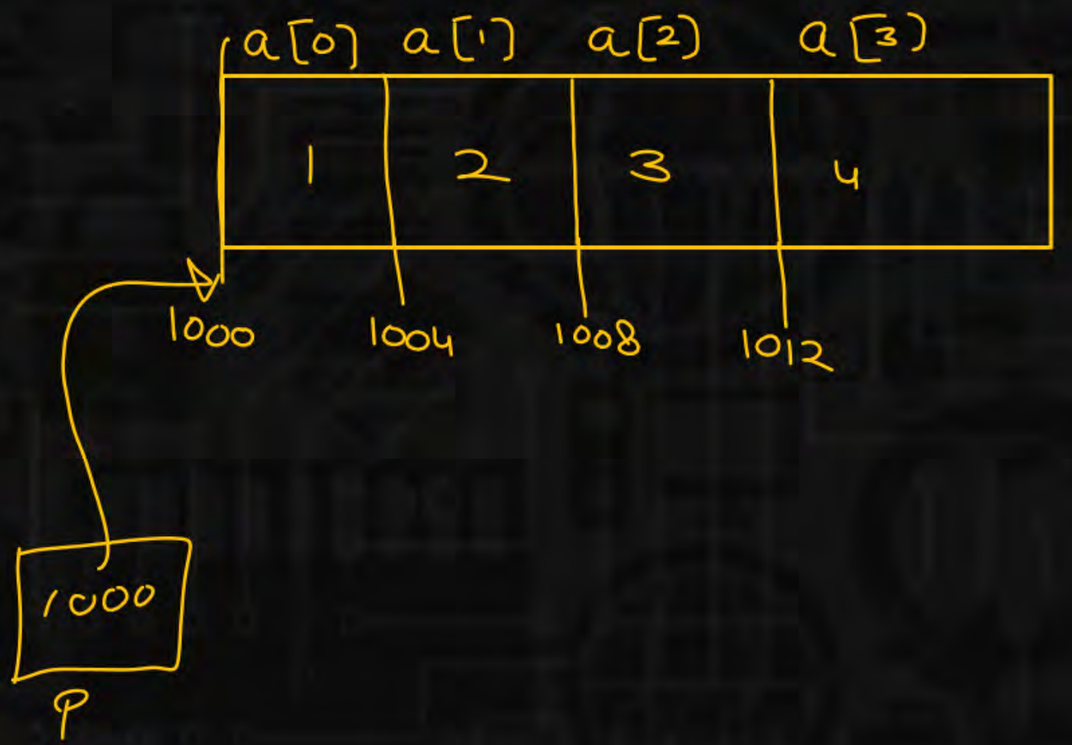
The output is-

A. Compilation Error

B. Runtime Error

C. 3 5 7

D. None



Q.2

[MCQ]

Consider the following program:

```
#include<stdio.h>
int * f(){
static int a[4]={1, 2, 3, 4};
return a;
}
int main(){
 int *p, i;
 p=f();
 for(i=0;i<3;i++){
printf("%d\t", p[i]+p[i+1]);
 }
return 0;
}
```

The output is-

(i) i = 0
P[0] + P[1]
*(P+0) + *(P+1)
1 + 2 ⇒ (3)

(ii) i = 1
P[1] + P[2]
*(P+1) + *(P+2)
2 + 3 ⇒ (5)

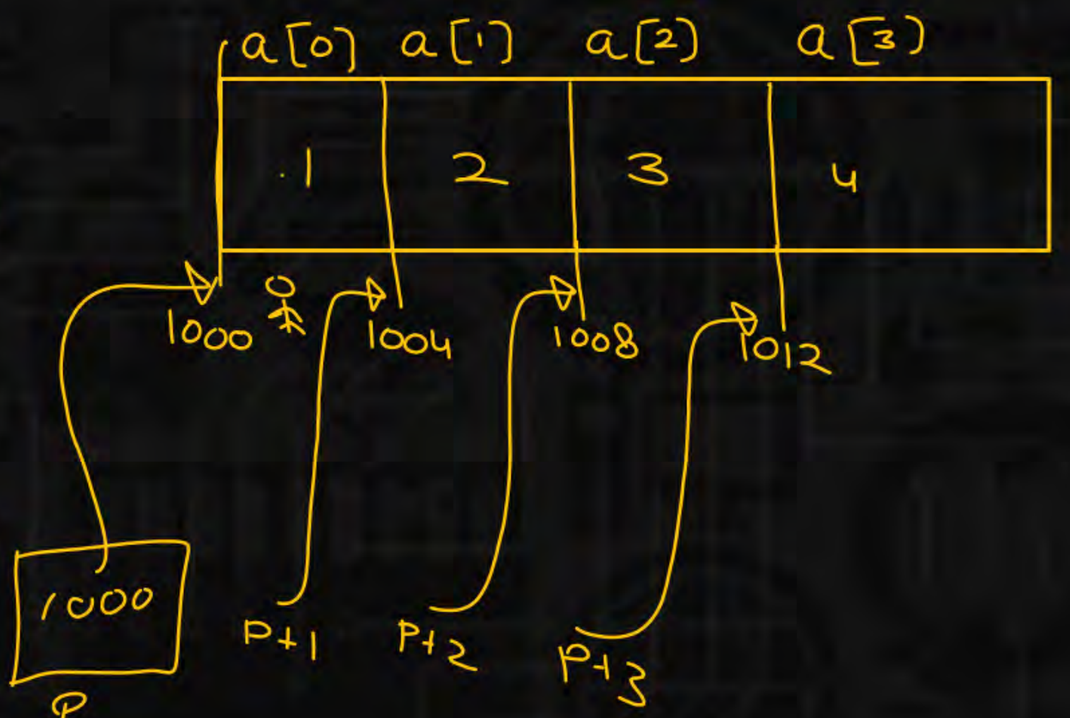
(iii) i = 2
P[2] + P[3]
3 + *(P+3) = (7)
3+4

A. Compilation Error

B. Runtime Error

C. 3 5 7

D. None



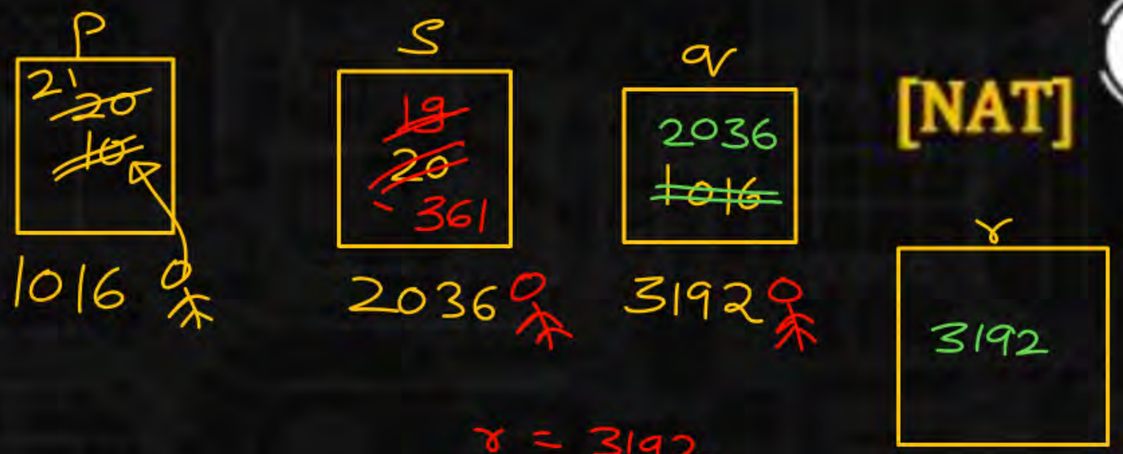
Q.3

[NAT]

Consider the following program:

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

The output is 382.



*q = (p++) + *q

(i) *q = p + *q
(ii) p = p+1

*q = 10 + 10 = 20

21 + 361 = 382

r = 3192
*r = 2036
**r = val at (M. 2036)
***r = --(*q) * (**r)

(i) *q = *q - 1 => *q = 20 - 1 = 19
(ii) **r = *q x (**r)
**r = 19 X 19
**r = 361

Q.4

[MCQ]

Consider the following program:

```
#include<stdio.h>
int * f()
{
    int a[4]={1, 2, 3, 4};
    return a;
}
    int main()
{
    int *p, i;
    p=f();
    for(i=0;i<3;i++){
    printf("%d\t", p[i]+p[i+1]);
 }
  return 0;
}
```

The output is-

A. Compilation Error

B. Runtime Error

C. 3 5 7

D. None

Q.5

[MCQ]

Consider the following statements:

P: int * p(int *) - p is a function that takes an integer pointer as argument and returns an nteger pointer.

Q: int (*p(int *))[] - p is a function that takes an integer pointer as argument and returns a pointer to an array of integers.

Which of the following is INCORRECT?

Correct

Correct

int * P (int *)

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

A. P only

B. Q only

C. Both P and Q

D. Neither P nor Q

int (*p(int *))[]

P is a function that takes integer pointer as argument and returns pointer to array of integers

Q.6

[MCQ]

Consider the following program:

```
#include<stdio.h>
void f(int (*q)[2]){
printf("%d\t",(*q)[1]);
q+=2;
printf("%d",(*q)[1]);
}
int main()
{
int a[][2]={2,4,6,8,10,12};
int (*ptr)[2]=a;
f(ptr);
return 0;
}
```

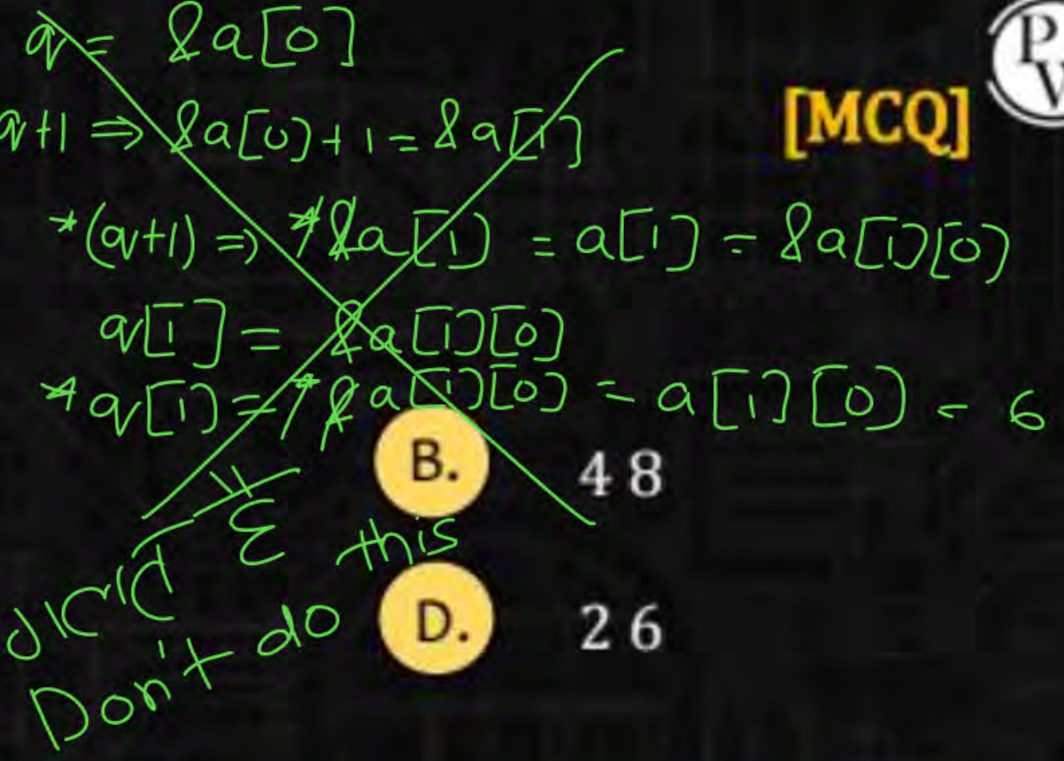
A. 4 12

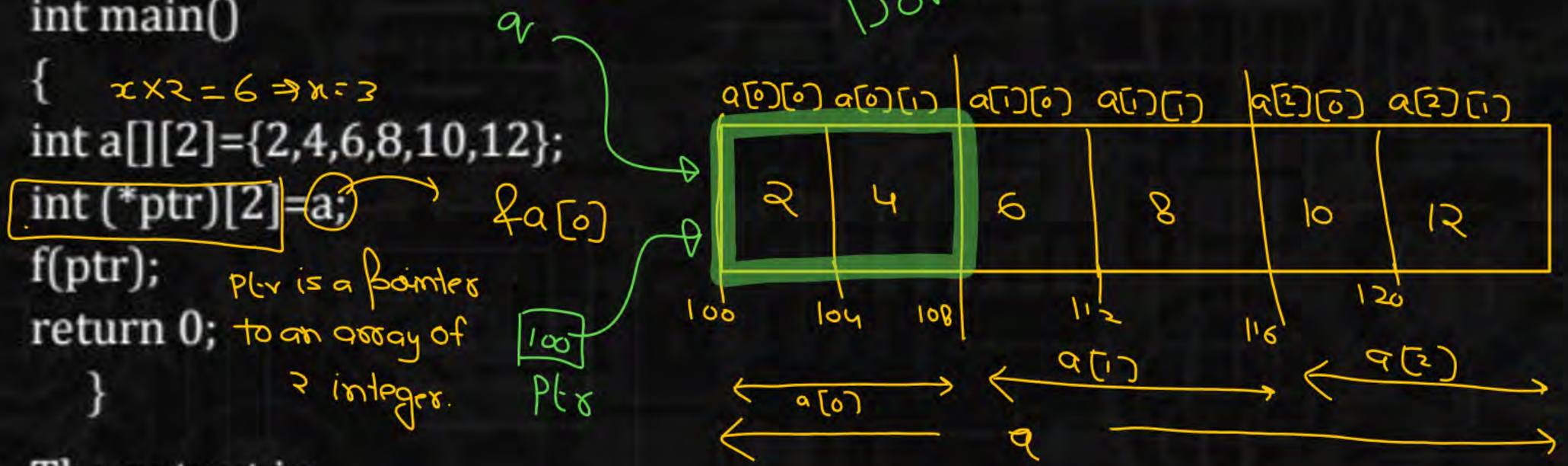
B. 4 8

C. 2 10

D. 2 6

The output is:





Q.6

[MCQ]

Consider the following program:

```
#include<stdio.h>
void f(int (*q)[2]){
printf("%d\t",(*q)[1]);
q+=2;
printf("%d",(*q)[1]);
}
int main()
{
int a[][2]={2,4,6,8,10,12};
int (*ptr)[2]=a;
f(ptr);
return 0;
}
```

(4)

q = &a[0]

(*q) = *&a[0] = a[0] = &a[0][0]

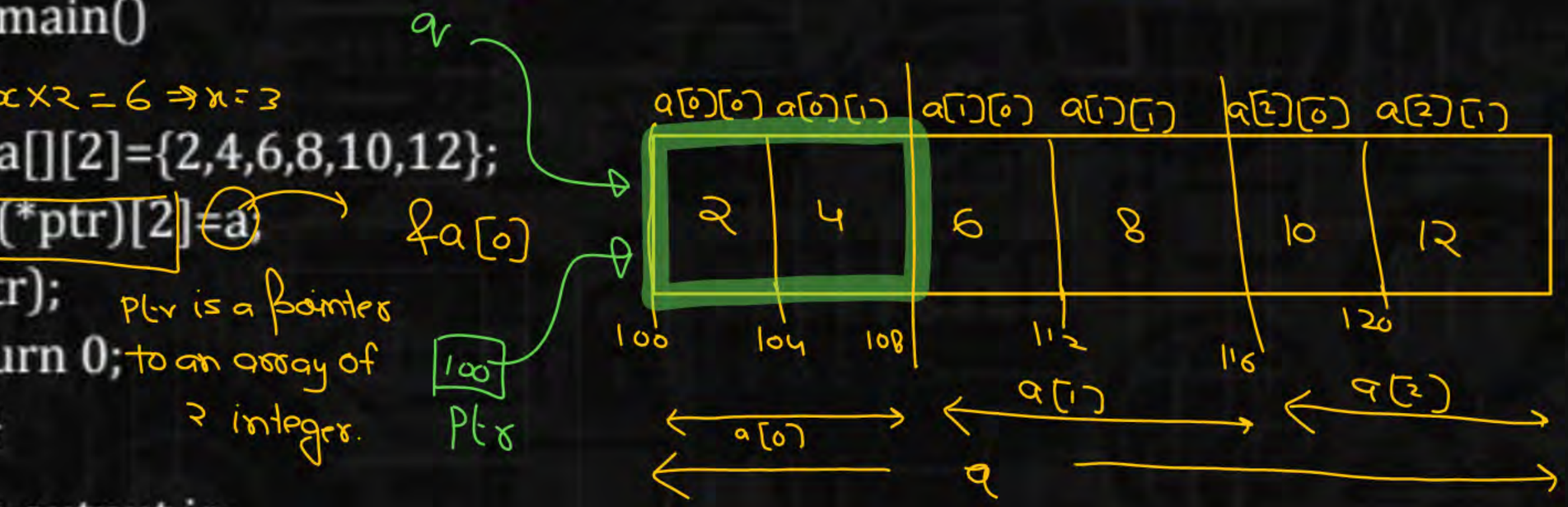
(*q)[1] = *(*q+1) = *(&a[0][0]+1) = *(&a[0][1]) = a[0][1]

A. 4 12

B. 4 8

C. 2 10

D. 2 6



The output is:

Q.6

[MCQ]

Consider the following program:

```
#include<stdio.h>
void f(int (*q)[2]){
printf("%d\t",(*q)[1]);
q+=2;
printf("%d",(*q)[1]);
}
int main()
{
int a[][2]={2,4,6,8,10,12};
int (*ptr)[2]=a;
f(ptr);
return 0;
}
```

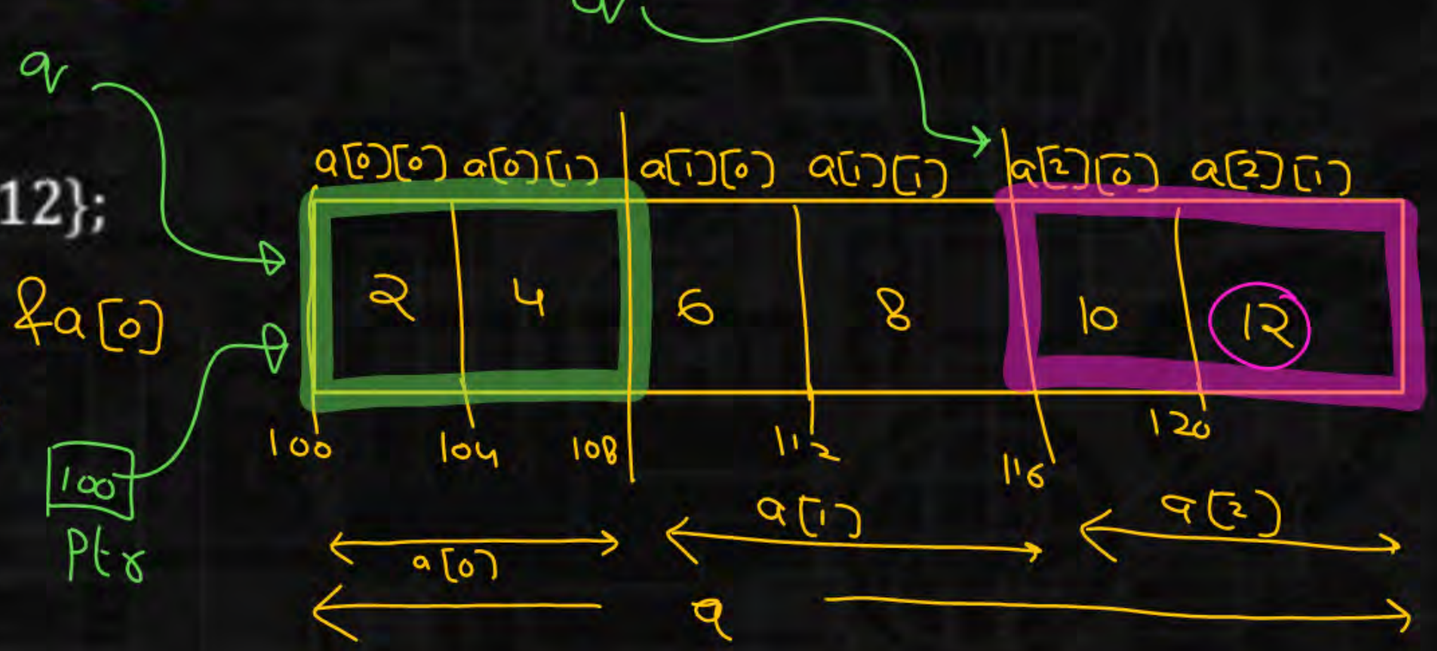
The output is:

A. 4 12

B. 4 8

C. 2 10

D. 2 6

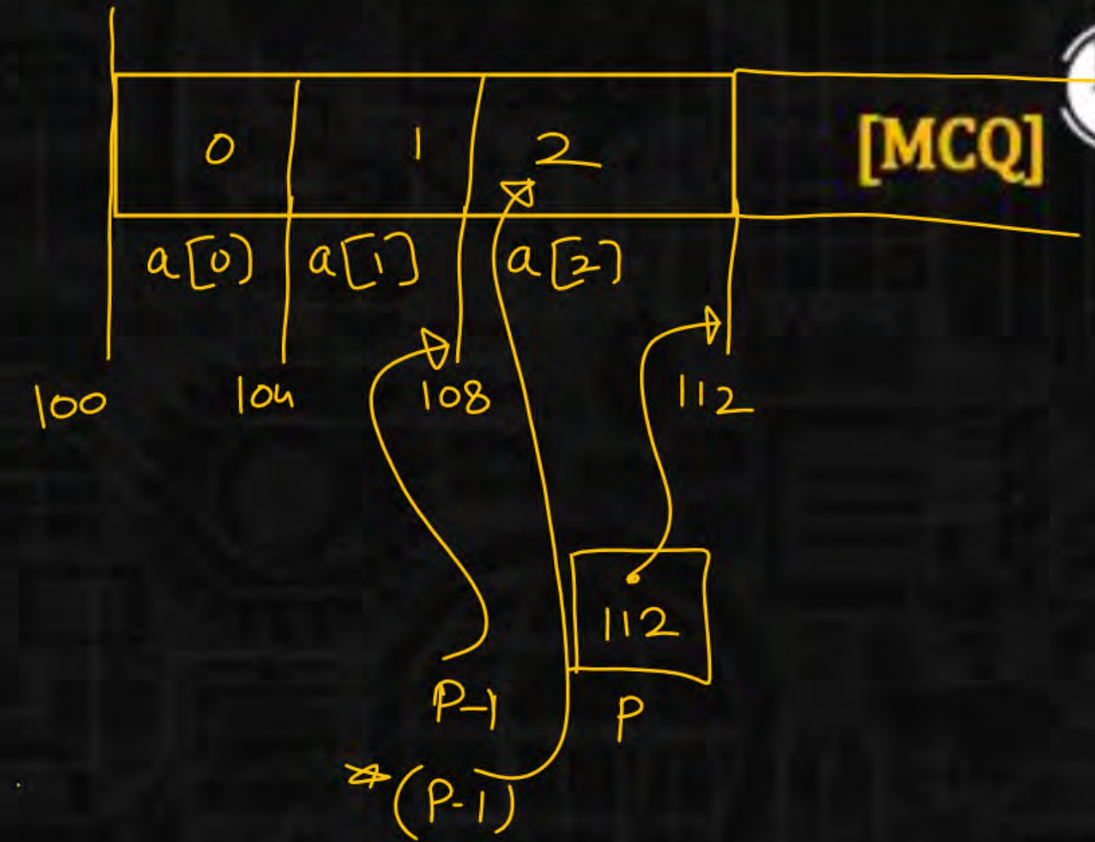


Q.7

[MCQ]

Consider the following program:

```
#include<stdio.h>
int main()
{
 int a[3]={0,1,2};
 int *p=(int *)(&a+1);
 printf("%d\t%d", *(a+1),*(p-1));
 return 0;
}
```



The output is-

A. Garbage value

B. Segmentation fault

C. 1 2

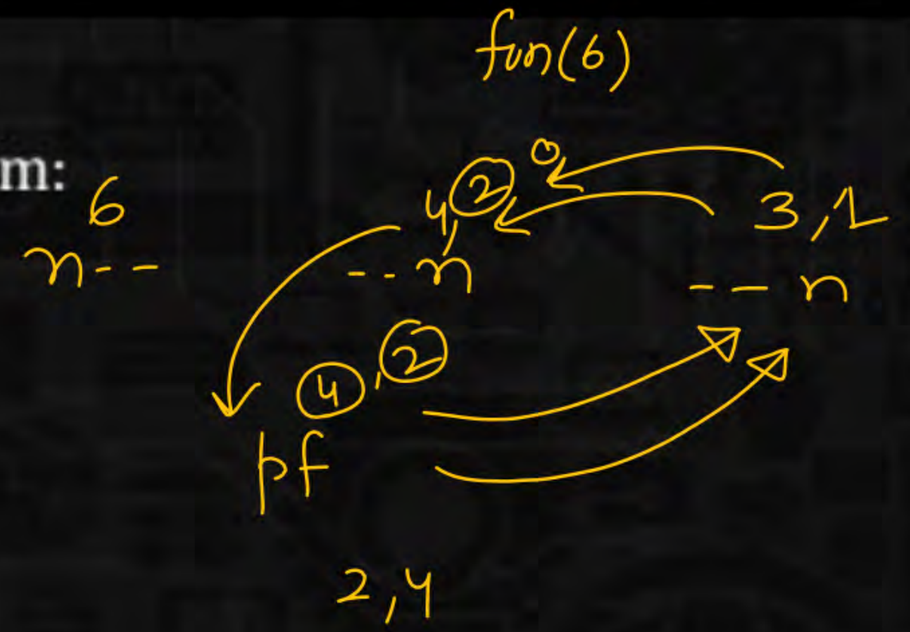
D. Compilation Error

Q.8

[MCQ]

Consider the following program:

```
#include<stdio.h>
void fun(int n){
for(n--;--n;--n)
printf("GATE WALLAH");
}
int main(){
 void (*p)(int)=fun;
 (*p)(6);
return 0;
}
```



The output is-

A. Compilation Error

B. Runtime Error

C. Printf() is executed infinite number of times.

D. Print() is executed two times

THANK YOU GW
SOLDIERS !