CS & IT ENGINEERING

Programming in C

Arrays and Pointers

Lec- 07

By- Pankaj Sharma sir

TOPICS TO BE COVERED

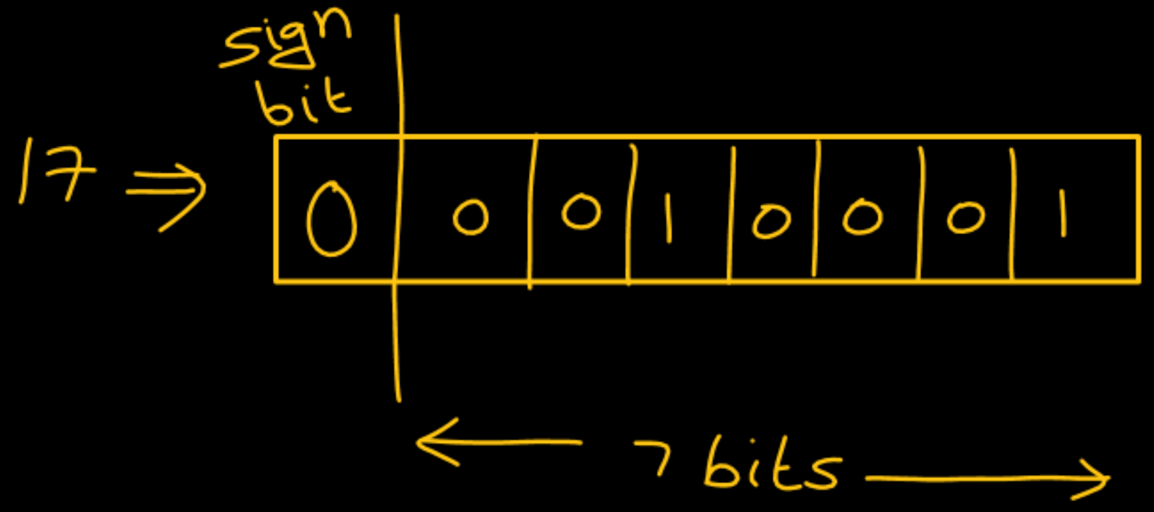
- Arrays and Pointers

Signed Magnitude

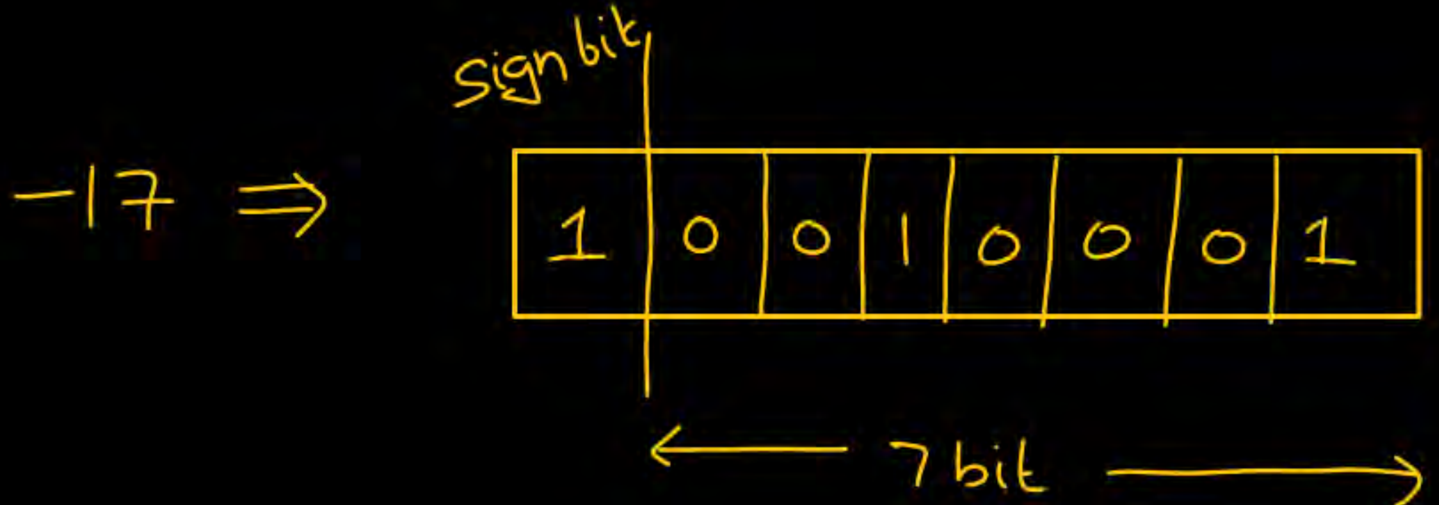
Sign bit | Magnitude

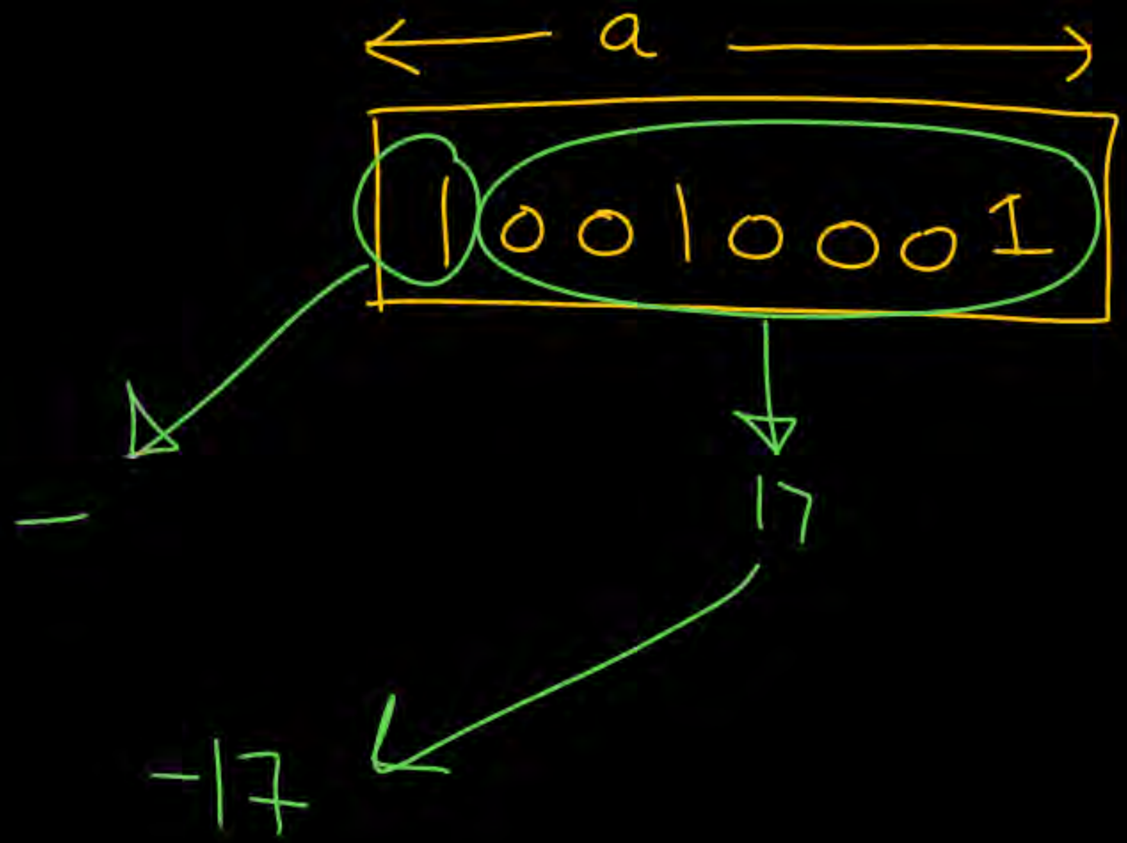
$+17 \Rightarrow 0$

$-17 \Rightarrow 1$



2	17	Rem
2	8	1
2	4	0
2	2	0
2	1	0
	0	1


-17 ⇒
sign bit
1 0 0 1 0 0 0 1
7 bit


a
1 0 0 1 0 0 0 1
-
17
-17

$$\textcircled{1}\underline{0000100}$$

$$\Rightarrow -4$$

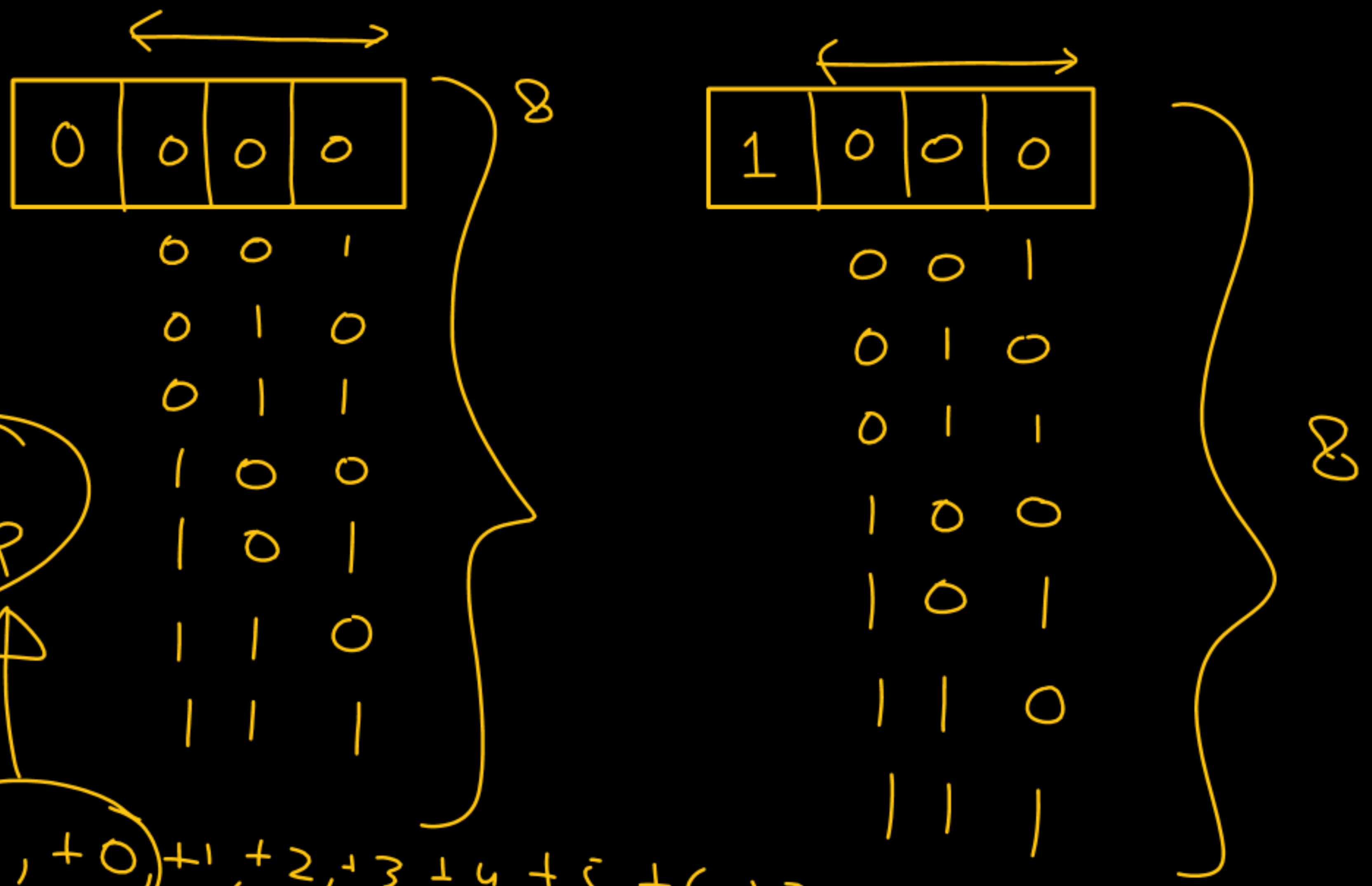
$$\left.\begin{array}{l} 00000000 \Rightarrow +0 \\ 10000000 \Rightarrow -0 \end{array}\right] \times$$

2's complementation

1) +ve no $\implies$ As it is.

2) -ve no $\implies$ In 2's complementation form

$\boxed{\text{4 bit}}$ $2^4 \Rightarrow 16$



2rep

$-7, -6, -5, -4, -3, -2, -1, -0, +0, +1, +2, +3, +4, +5, +6, +7$

x : 00101010

1's comp : 11010101

x : 00101010

$$\begin{array}{rl} & \overset{1}{} \\ \text{1's comp:} & 11010101 \\ & 00000001 \\ \hline \text{2's comp} \Rightarrow & 11010110 \\ \hline \end{array}$$

1's comp + $\overset{\text{Add}}{1}$

$0 + 0 = 0$

$0 + 1 = 1$

$1 + 0 = 1$

$1 + 1 = \cancel{\boxed{2}}$

$\downarrow$

10

Carry $\nearrow$ $\nwarrow$ Sum

2's comp direct

x : 1 0 1 1 0 1 | 1 0 $\longleftarrow$

0 1 0 0 1 0 | 1 0

8 bit

(i) +11 ⇒ 00001011

(ii) −11 ⇒ a) +11 ⇒ 00001011

b) 2's comp ⇒ $\boxed{11110101}$

a

$\boxed{\textcircled{1}1110101}$

⇓

2's complementation

−

$\textcircled{-11}$

2's comp

$\underbrace{00001011}$

⇓

11

(2's comp)

$$\begin{tikzcd}
 & 1\,1\,1\,1\,0\,1\,0\,1 \arrow[dl] \arrow[dr] & \\
\text{Value} \arrow[d] & & {} \\
\text{(i) ~~Ft.~~ Method} & & \text{(ii) Method}
\end{tikzcd}$$

$$\begin{array}{cccc} \boxed{1} & \overset{\times}{0} & 1 & \overset{\times}{0} \\ 2^3 & 2^2 & 2^1 & 2^0 \end{array}$$

$$1 \times 2^3 + 1 \times 2^1$$

$$= \boxed{10}$$

$$\begin{array}{cccccccc} 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 \\ 2^7 & 2^6 & 2^5 & 2^4 & 2^3 & 2^2 & 2^1 & 2^0 \end{array}$$

$$-2^3 - 2^1 - 1$$

$$= -8 - 2 - 1 = \boxed{-11}$$

$$\begin{array}{cccccccc} 1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 2^7 & 2^6 & 2^5 & 2^4 & 2^3 & 2^2 & 2^1 & 2^0 \end{array}$$

$$\Rightarrow$$

$$-2^6 - 2^3 - 2^1 - 2^0 - 1$$

$$= -64 - 8 - 2 - 1 - 1$$

$$= \boxed{-76}$$

0000	$\Rightarrow$	0
0001		1
0010		2
0011		3
0100		4
0101		5
0110		6
0111		7

$2^3\ 2^2\ 2^1\ 2^0$				
1 0 0 0	$\Rightarrow$	$-2^2-2^1-2^0-1$	$=$	-8
1 0 0 1	$\Rightarrow$	-2^2-2^1-1	$=$	-7
1 0 1 0	$\Rightarrow$	-2^2-2^0-1	$=$	-6
1 0 1 1	$\Rightarrow$	-2^2-1	$=$	-5
1 1 0 0	$\Rightarrow$	-2^1-2^0-1	$=$	-4
1 1 0 1	$\Rightarrow$	-2^1-1	$=$	-3
1 1 1 0	$\Rightarrow$	-2^0-1	$=$	-2
1 1 1 1	$=$	-1	$=$	-1

$$\boxed{-8, \ldots, -3, -2, -1, 0, 1, 2, \ldots 7}$$

$\longleftarrow$ 31 bit $\longrightarrow$

$\boxed{0 \mid 0\,0\,0\,0\,0\,0 - - - \cdots\cdots 0} \rightarrow 0$

$111 \Rightarrow 2^3 - 1$

$1111 \Rightarrow 2^4 - 1$

$11111 \Rightarrow 2^5 - 1$

$\downarrow$

$0\ 1\,1\,1\,1 - - - \cdots\cdots 1 \Rightarrow 2^{31} - 1$

$\boxed{1 \mid 0\,0\,0\,0 \cdots\cdots\cdots 0} \Rightarrow -2^{30} \cdot 2^{29} \cdot 2^{28} \cdots 2^{0} - 1 \Rightarrow -2^{31}$

$2^{30} - - - - - 2^1\, 2^0$

$-(2^{31} - 1) - 1$

$= -2^{31} + \cancel{1} - \cancel{1} = -2^{31}$

$\boxed{1 \mid 1\,1\,1 \cdots - \cdots\cdots - 1}$

-1

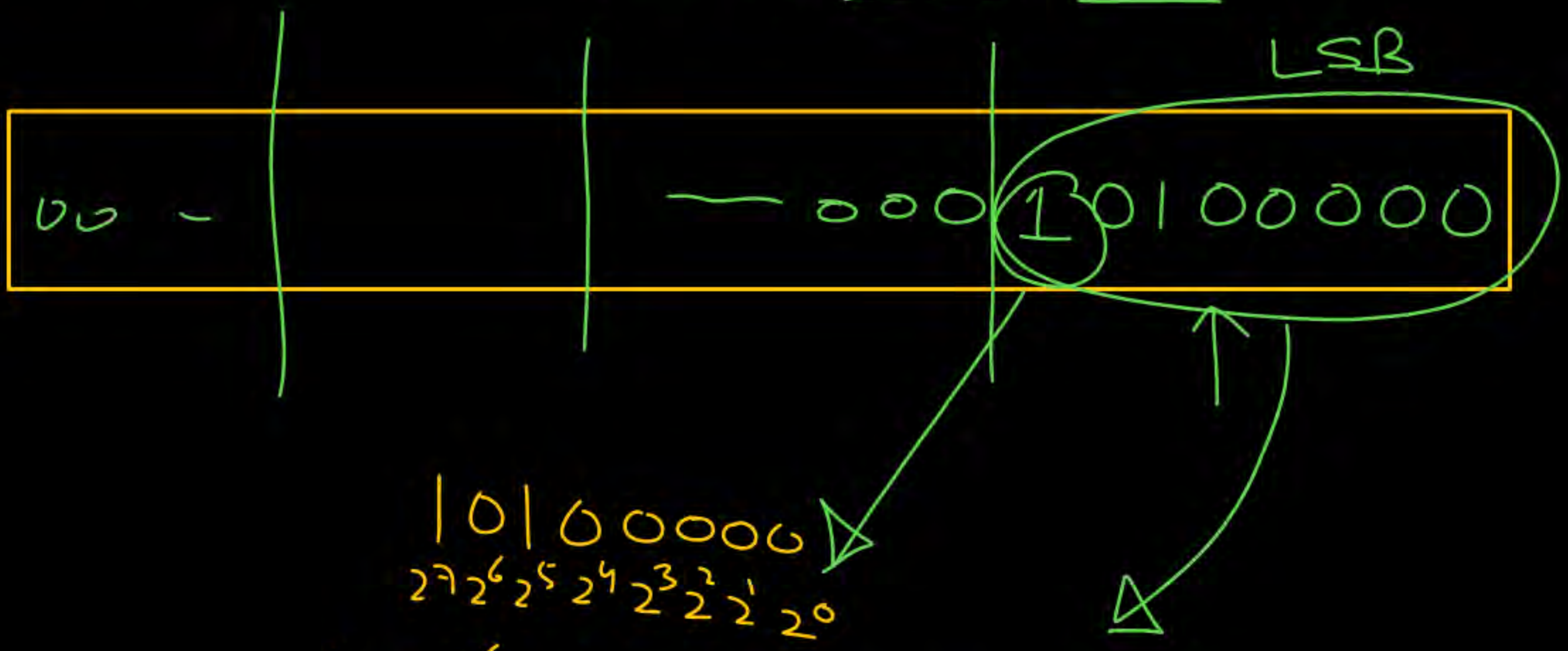
```
int a = 256;

char *p;

p = (char*)&a ;

printf("%d", *p);
```

compiler को

धरना

Little Endian

Big Endian



$160 \Rightarrow \underline{128} + \underline{32}$

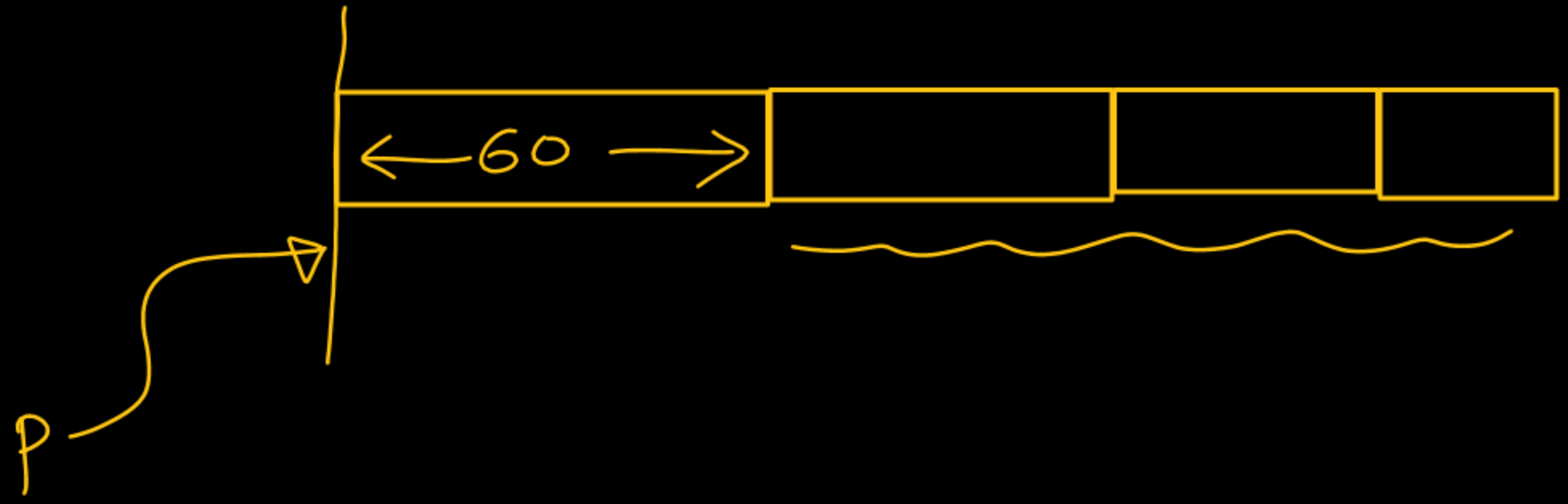
10100000

$2^7 2^6 2^5 2^4 2^3 2^2 2^1 2^0$

$= -2^6 - 2^4 - 2^3 - 2^2 - 2^1 - 2^0 - 1$

$= \underbrace{-64-16-8-4-2-1-1}_{-96}$

```
char a = 60;
int *p = (int*) &a;
printf("%d", *p);
```



→ garbage

int size ⇒ 4 byte

```
int *p;                |   char *p;
                       |
  ⋮                    |     ⋮
                       |
p = p+2                |   p = p+2
```

```
void *p;
```

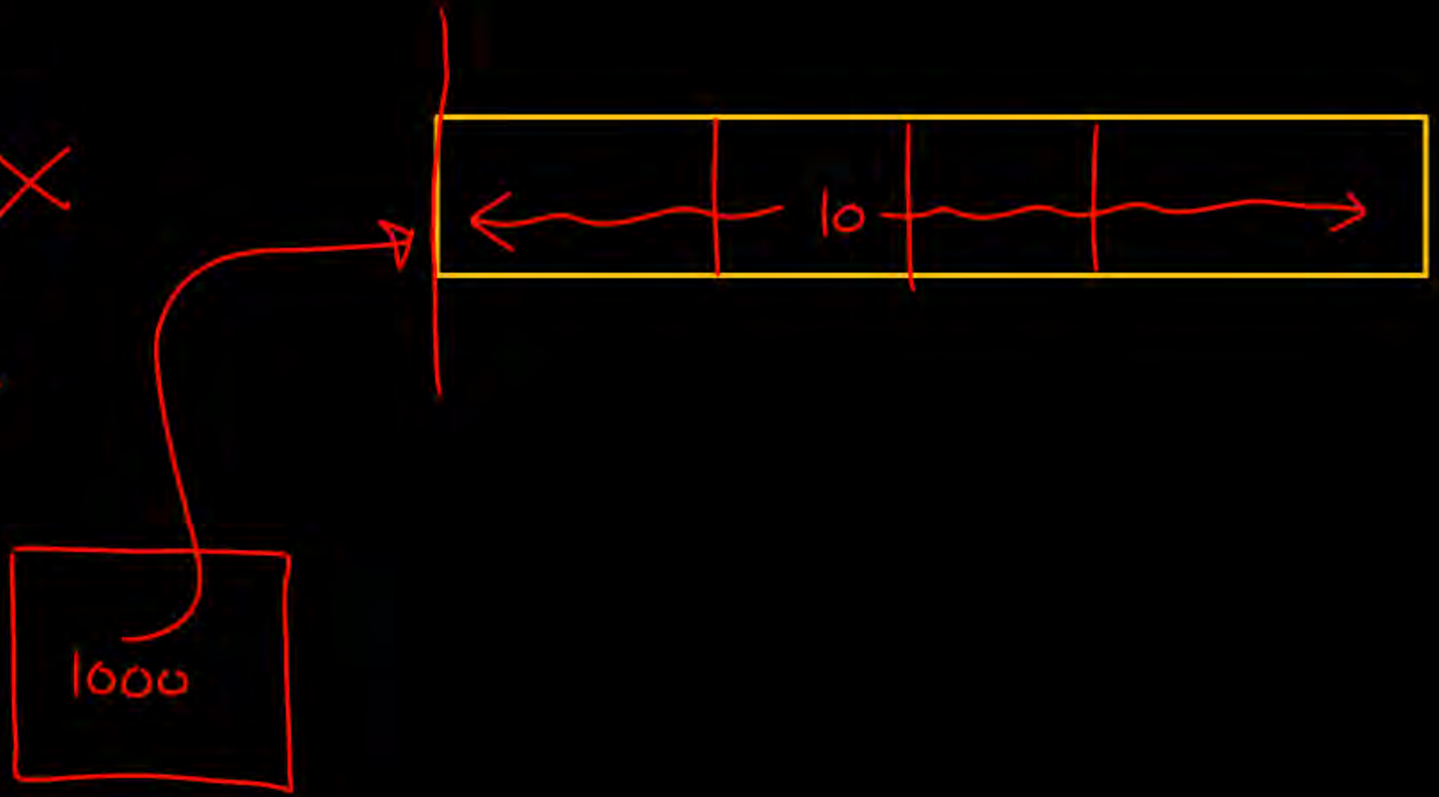
p is a pointer

```
int a=10;
float b=17.38;
p = &a
```

Compiler will shout ✗

or not ✓

```
printf("%d",*p);
```



void Pointer

1. Can not derefrence directly.
2. first typecast then ________

```
void *p;

int a = 60;

p = &a;

printf("%d", *(int *)p);
```

$*p \Rightarrow *\underbrace{(int *)}p$

$\boxed{\text{typecasting}}$

$\boxed{\text{int *p}}$;

=

$$\underbrace{p = p + 2}_{2 \times \text{sizeof(int)}}$$

char *p;

=

$$p = p + \underbrace{\textcircled{2}}_{2 \times \text{sizeof(char)}}$$

```
void *p;

  =

$\boxed{p = p+2;}$
```

Do not perform any arithmatic on void pointer

$\rightarrow$ Error (logically)

Wild Pointer

आवारा

Uninitialized Pointer

Garbage

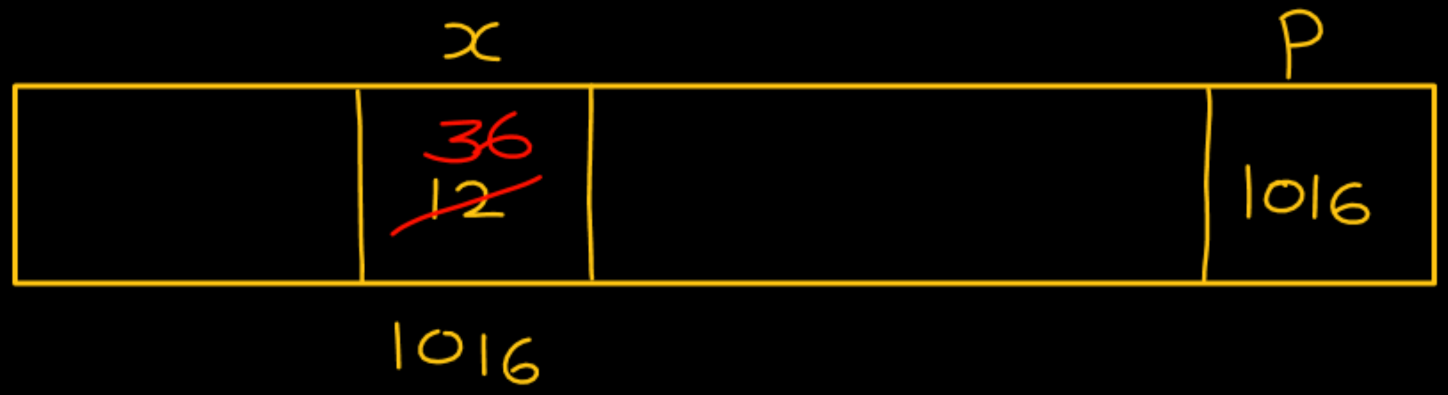
```
void main()
  {
  int a ;
  printf("%d",a);
  }
```

```
void main(){

   int *p;
   ≡

   }
```

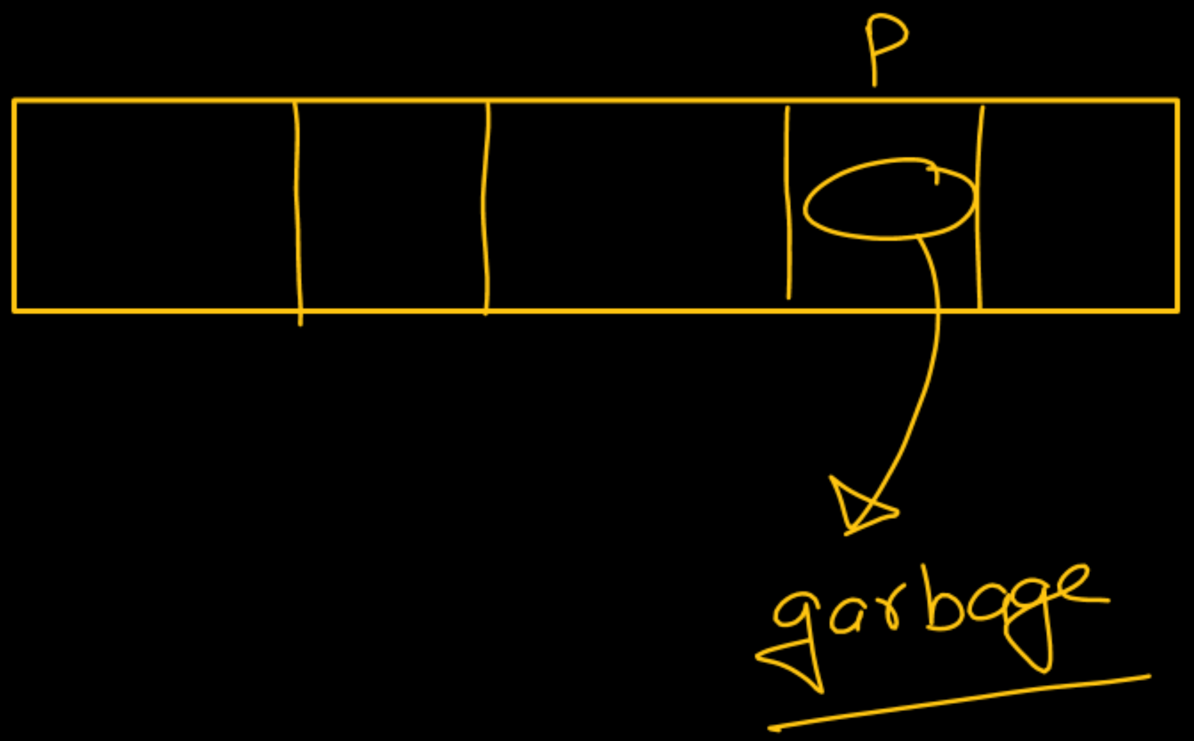
negligible

```
void main(){
      int *p;
      int x = 12;
      *p = 36;
```

danger



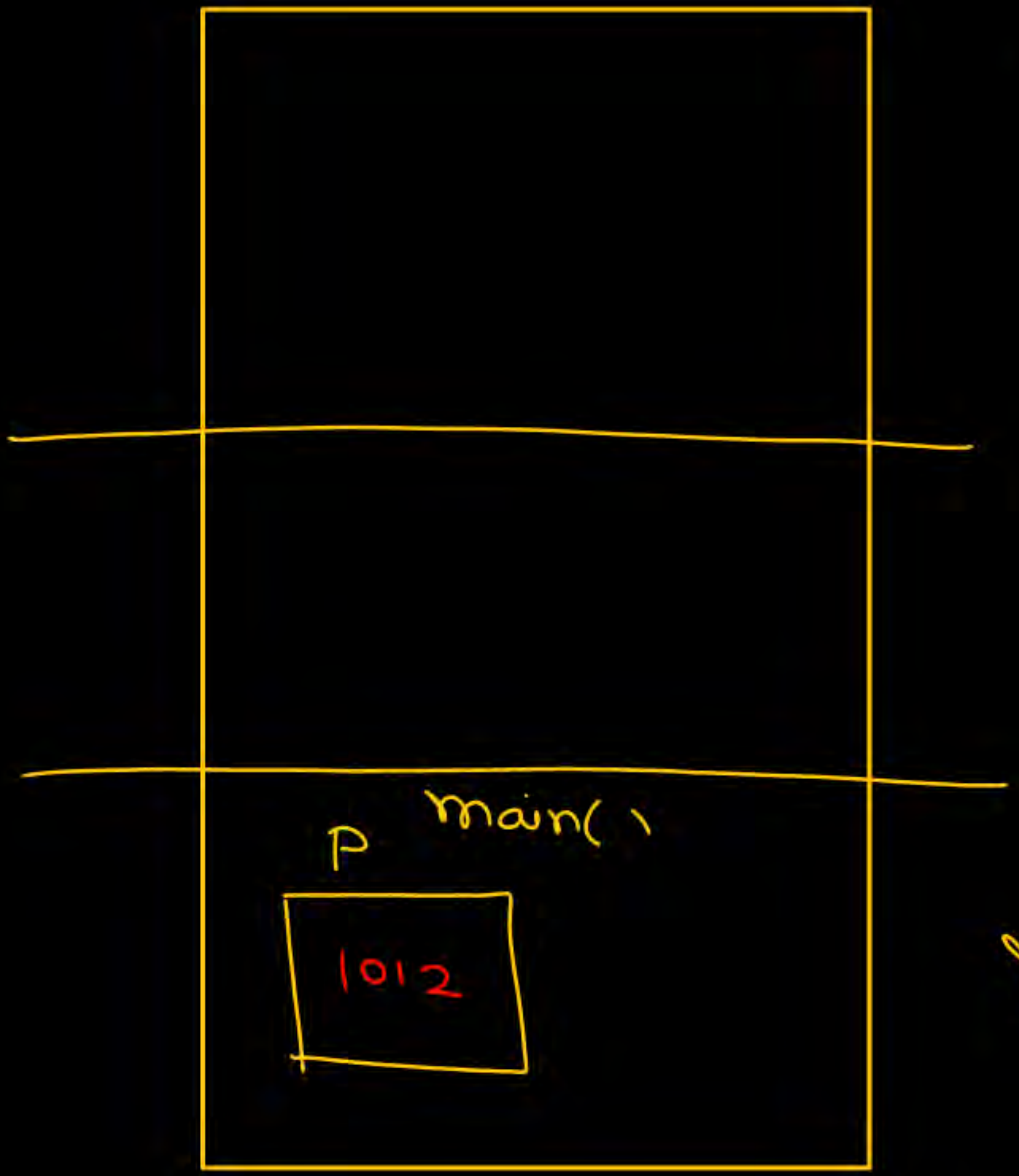
```
void fun( ){
    int *p;

    int a ;

    *p
}
```



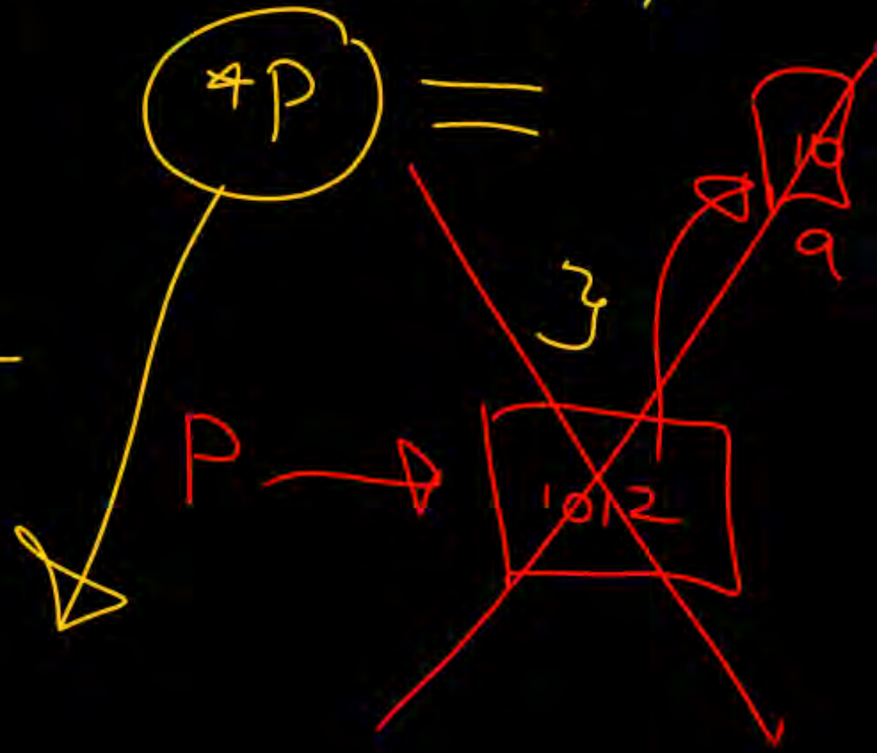
Dangling Pointer

```
int* fun(  )
  {
      int a = 10;
      int *q = &a;
      return q; ,
  }
```

$p \rightarrow$



```
void main(){
    int *p;
    p = fun();
    *p ==
}
```



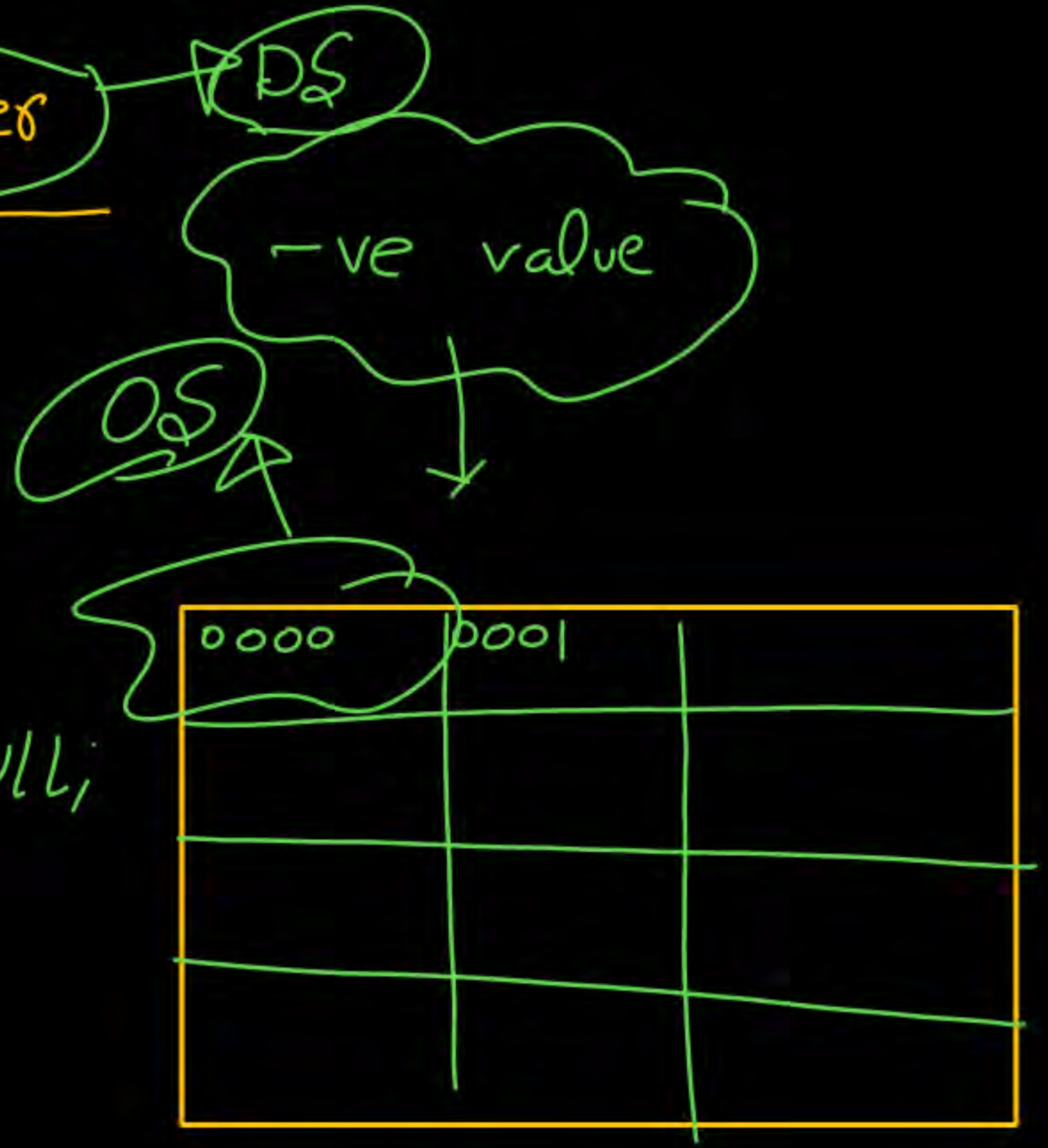
return type

```
int * f(   )
{
  static int a = 10;
  return &a;
}
```

lifetime → through-out the program

```
void main(){
    int *p;
    P = f();
    printf("%d", *p);
}
```

NULL Pointer



if(0)

ptr == NULL;

if(ptr)

int *P = (int*) 0 ;] P ⇒ NULL

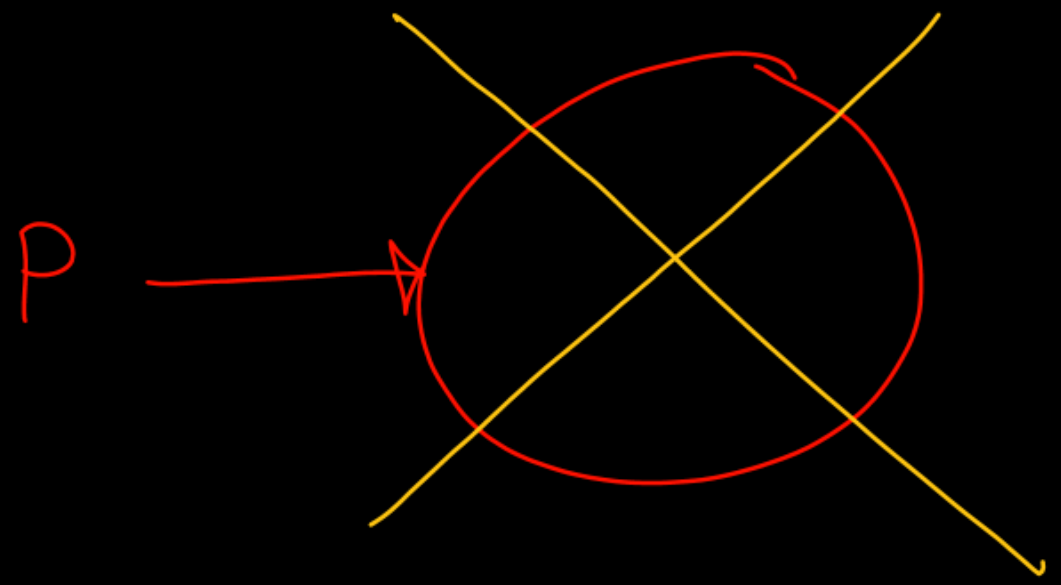
DS

Dynamic Memory Allocation

void * ✓

Dang ✓

NULL ✓



$P \longrightarrow ?$

THANK YOU GW SOLDIERS !