Pointers → ①

Date 4/oct/2023

Pointers → Is a type of variable → which stores the address of other variable

'&' → "address of" operator to return the address of a variable

```
int a = 10;
cout << a → ?  → 10
cout << &a → ? → 104
```

(a: 104 | 10)

(104 | 5 → value)

- ptr = value stored in ptr
- &ptr = address of ptr itself
- *ptr = (I) add pe jaao → value present at add stored in ptr
  (II) value print kro
- &a → add. of a
- a → value of a

→ Creation of pointer :→

int * | ptr = ------ (address)

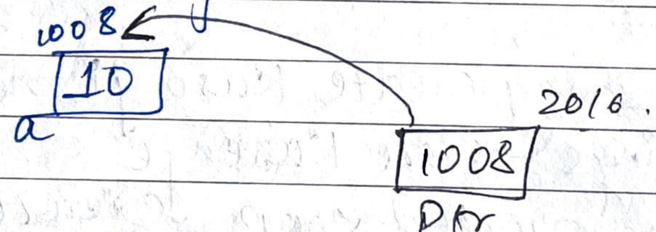
- int * → Pointer to integer data
- ptr → variable/pointer name

{ char * ...
long * → Pointer to long data
⋮ }

Ptr is a pointer to integer

```
int a = 10;
int * ptr = &a
```

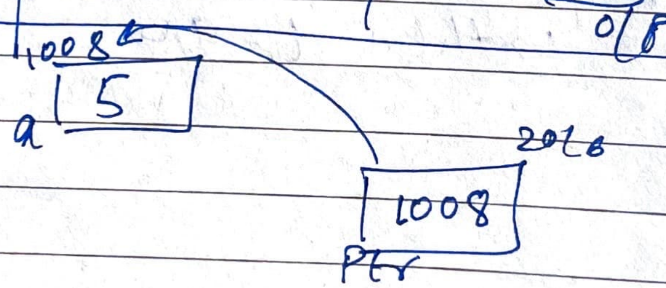
creates a pointer and point it to address of a



⇒ Access :→ value stored at address stored by ptr.

↳ dereference opr :→ "*"   Ex :→ cout *ptr = 10

```
int a = 5
int *ptr = &a
⇒ ptr = 1008
&ptr = 2016
*ptr = 5
```



In pointer there will always be same type of data i.e, Address.

Date

⇒ Declare the pointer :-

```
int *ptr;
cout << *ptr.
```

→ Bad practice for declaring the pointer

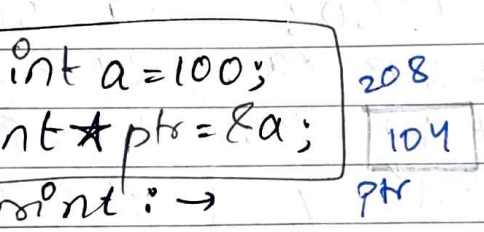
because here you are trying to access a random memory which your program may not be allowed to access (Gives Runtime error)

✓ Good practice

```
int *ptr = 0;
```

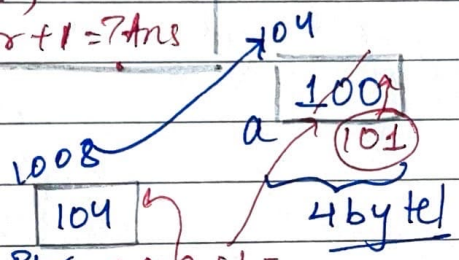
Because here atleast we know that our pointer is pointing to a null variable. (which will make easier to debugg)

Q
```
int a = 100;
int *ptr = &a;
```



Print :→
a = 100
&a = 104
*a = error
ptr = 104
*ptr = 100
&ptr = 208
(*ptr)++ = ~~100~~ 100+1 = 101
++(*ptr) = 101+1 = 102
*ptr = *ptr / 2 = 102/2 = ~~51~~ 51
*ptr = *ptr - 2 = ~~51~~ - 2 = 49

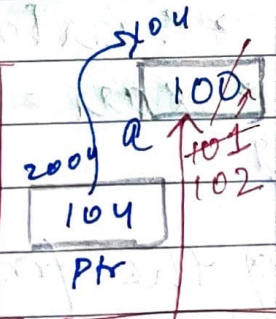
Q
```
int a = 100;
int *ptr = &a;
a = a + 1;  = ? ans
ptr = ptr + 1 = ? ans
```



a = a + 1 =
ptr = ptr + 1
104 + 1 = 108

Cox integer is of 4 byte, it will point to next int box

Q
```
int a = 100.
int *ptr = &a
a = a + 1
*ptr = *ptr + 1
```



value present at address stored in ptr.

101 + 1 = 102

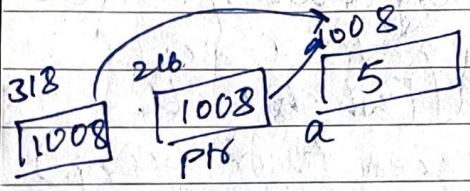
Can be said *ptr also

Pointer copy

Date

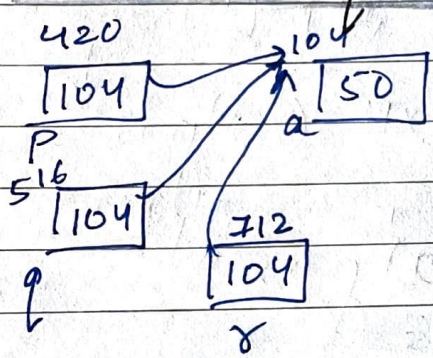
Q
```
int a = 5;
int *p = &a
int *q = p;
```

a = 5 | *p = 5
&a = 1008 | q = 1008
*a = error | &q = 318
p = 1008 | *q = 5
&p = 216



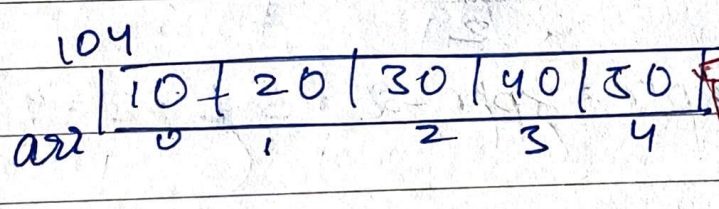
Q
```
int a = 50;
int *p = &a;
int *q = p;
int *r = q
```

a = 50 | q = 104
&a = 104 | &q = 516
*a = Error | *q = 50
p = 104 | r = 104
&p = 420 | &r = 712
*p = 50 | *r = 50



⇒ Pointer with array :→

```
int arr[5] = {10, 20, 30, 40, 50}
```



arr = Base addd (104)
arr[0] = 104
&arr[0] = 104
&arr = 104

arr, arr[0], &arr[0], &arr

They all are the name of the first indeg array they all return the Base address of first block of the array.

{arr[i] in CPU breakdown → *(arr+i) ⇒ *(BOA+i)} { i[arr] is valid }

Date

① int arr[5] = {10, 20, 30, 40, 50}

↳ Print → arr = 104.

→ &arr = 104

arr[0] = 10

&arr[0] = 104

*arr = 10

*arr+1 = 11

*(arr)+1 = 11 → (104+1) = 108 [1 index]

*(arr+1) = 20

*(arr+2) = 30 → (104+2) = 112.

*(arr+3) = 40. → (104+3) = 116.

values

104	108	112	116	120
10	20	30	40	50
0	1	2	3	4

arr

Note:→ int a = 5;

int *p = &a

p = p+1 → is possible or not?

yes it is possible since it moves from 104 to 108 [but throws garbage value]

214	104
[104]	[5]
p	a

↳ In case of static arr

arr[5] = {1, 2, 3, 4, 5}

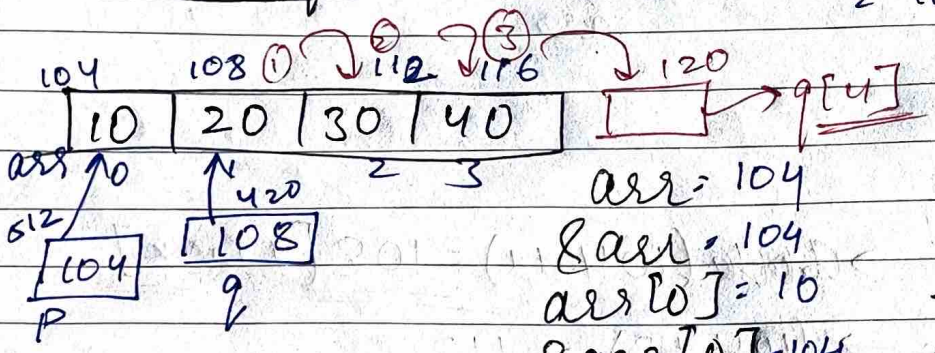
↳ arr = arr+1 ? → ye hum kar sakte hai kya??

↳ It gives error not possible.

Spiral

Date _____

① `int arr[4] = {10,20,30,40}`
`int *p = arr ;` → `arr[0]`
`int *q = arr+1;` → `arr[1]` → 104 + 1
= 104 + 1×4 = 108

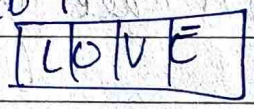


arr = 104
&arr = 104
arr[0] = 10
&arr[0] = 104
p = 104
&p = 512
*p = 10

q = 108
&q = 420
*q = 20
*p+1 = 10+1 = 11
*(p)+2 = 10+2 = 12
*(q)+2 = 20+2 = 22
*(q+4) = ~~20~~ → ~~108~~ + q[4]
~~104~~ 108 + 112 + 116
120 → q[4] access out of bound

`int arr[4] = {10,20,30,40}`
sizeof(arr) = 4×4 = 16
`int *ptr = arr`
sizeof(P) = 8

② `Char Ch[50] = "Love"`
`Char * cptr = ch`
`cout << cptr;` → Love



*cptr ⇒ *(cptr) ⇒ *(cptr+0) ⇒ *cptr[0]
⇒ 'L'

Dono ek hi baat hai

Date

Q
```
char ch[50] = "Love";
char *cptr = ch / &ch
```

ch = Love   &cptr = 208
&ch = 104   *cptr = *(cptr+0) → cptr[0] → l
ch[0] = l   cptr → Love

208 [104] cptr   [L|o|v|E] 104

Q
```
char ch[30] = "Statement"
char *cptr = &ch[0]
```

216 [104] cptr → 104 [S|t|a|t|e|m|e|n|t] (index 0 1 2 3 4 5 6 7 8)

ch = Statement   cptr = Statement   cptr + 2 = atement
&ch = 104   &cptr = 216   *cptr = cptr[0] = S
*(ch+3) = ch[3] = t   *(cptr + 3) = t (cptr[3])   cptr + 8 = t

Q
```
char ch = 'a';
char* cptr = &ch
cout << cptr
```

Spiral