1/27

Show images example

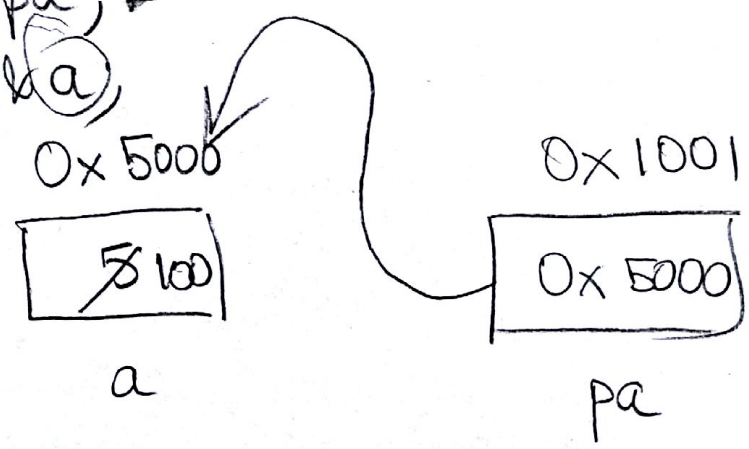
Definition

A pointer is a variable, whose value is the address of another variable.

i.e it stores a memory location.

```
type *identifier;
int *ip;
```

```
int a = 5;
int *pa;
pa = &a;
```



printf ("%d", a) → 5

("%p", ap) → 0x5000

("%p", &a) → 0x5000

// ("%p", &pa) → 0x1001

Dereference operator * → The same as, just using 'a' here

("%d", *pa) → 5.

value of the variable that the pointer points to.

*pa = 100;

printf ("%d", a) // => 100

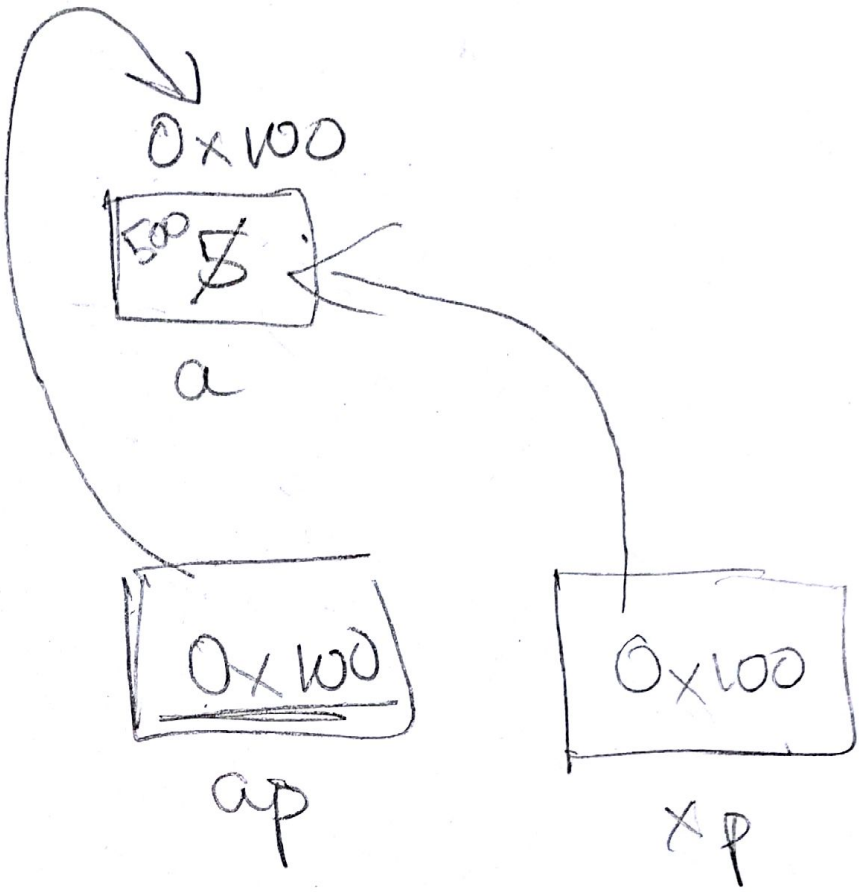
a = 5

int *ap = &a;

&a → 0x100

ap → 0x100

*ap → 5



int * xp;

xp = ap

*xp = 500;

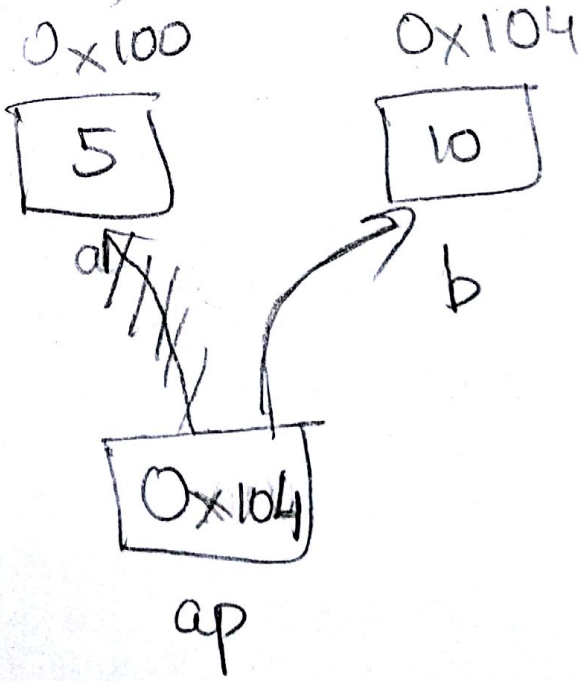
printf ("%d", *ap); → 500

int *ap = &a

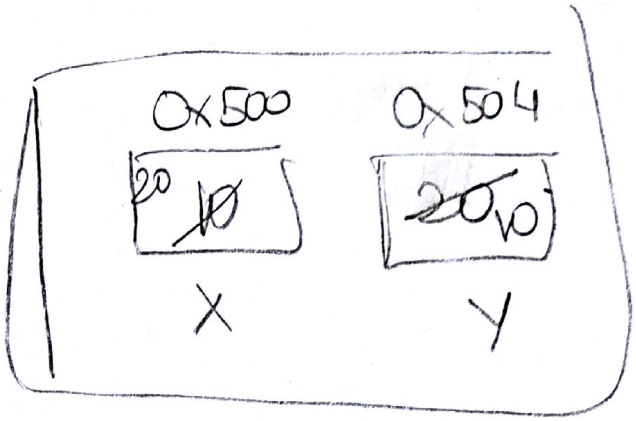
ap = &b;

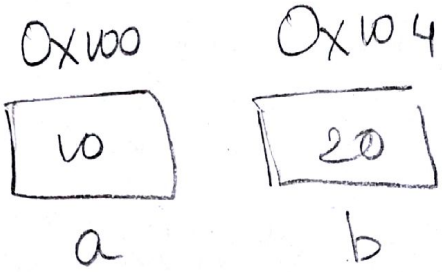
printf ("%d", *ap)

→ 10.



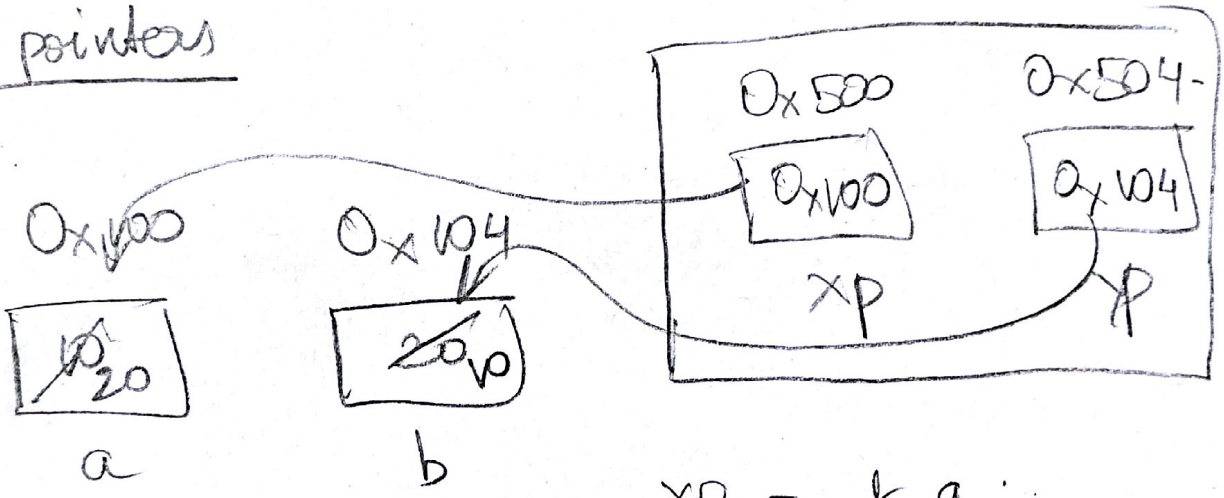
Swap normal






```
x = a;
y = b;
```

Swap pointers




```
int temp = *xp;
*xp = *yp;
*yp = temp;
```

```
xp = &a;
yp = &b;
```

Arrays & Pointers

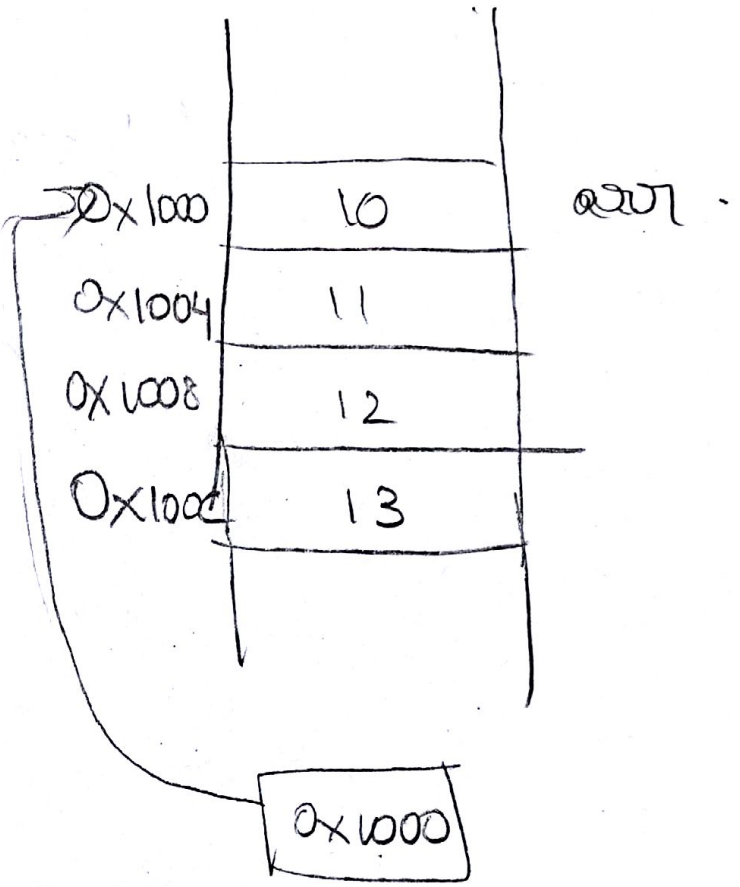
int *ap = &a;

int arr[4] = {10, 11, 12, 13};

printf("%p", arr)

→ 0x1000.

int *ptr = arr;



arr[0] → 10.

("%d", *ptr);

→ 10.

arr[1] → 11.

*(ptr + 1) → 11

0x1000 + 4

~~*~~ (0x1004)

ptr++;

→ 0x1004

ptr++;

→ 0x1008

arr[4] = {10, 11, 12, 13};

arr[10] → C will let you access this even though it is out of bounds
// Undefined behaviour.

Why we use Pointers? What's the point?

→ Not really that useful for basic types.
But powerful for derived types

→ Memory allocation at runtime

→ Function arguments (emulate pass by reference)