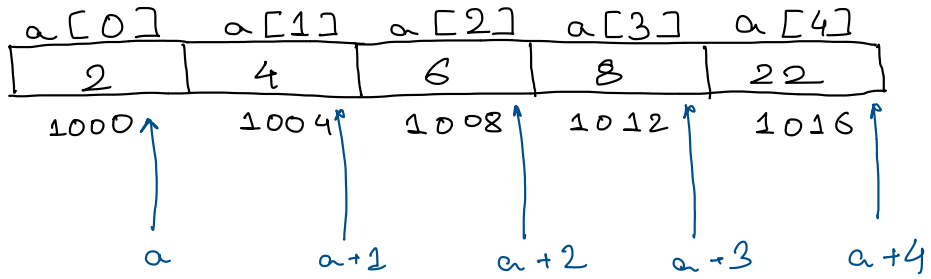
9a - Arrays and Pointers

Thursday, 13 August 2026 04:26

`int a[5];` a is pointer to first element of the array.



So `a = &a[0];`

Also `a[1] = *(a+1)`

De-Referencing

↳ The * is called de-referencing a pointer.
It allows us to retrieve the exact value stored in a pointer's address

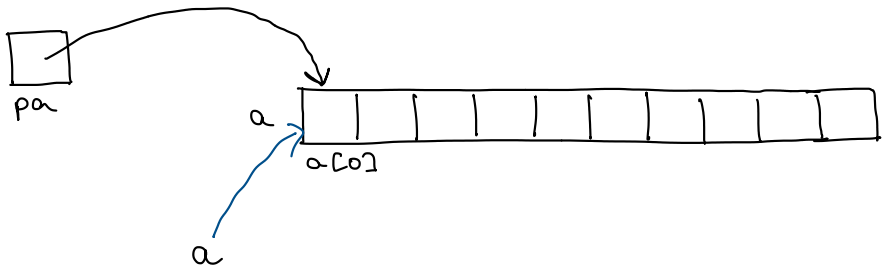
`a[i] = *(a+i)`

∴ p: Designed for specifically address.

Pointer and Arrays

```
int *pa;          // ptr initialised
int a[10];        // arr of size 10 made
pa = &a[0];       // ptr points to index 0 of array
```

∴ pa = a = &a[0];



Difference in pointer pa & a

→ 'a' can't be on LHS

i.e. a = (a+1) is illegal

but we can use pointer

→ pa has it's own memory box

→ sizeof(pa) → size of address (so 4 if int array)

but sizeof(a) → no. of int × size of one int.

'pa' can point to any address

'a' can only point to &a[0];

Q

```
int a[] = {3, 6, 9, 12, 15};
int* addr = &(a[0]);
```

- What is the value of (*addr)+4?

→ *addr is de-referenced ptr so holds value of a[0];

- What is the value of *(addr+4)?

addr+4 points to index 4
*(addr+4) is value at index 4.

Ways to access n^{th} element in array

i) a[n]

ii) n[a]

iii) *(a+n)

iv) pa[n]

v) n[pa]

vi) *(pa+n)