

---

---

---

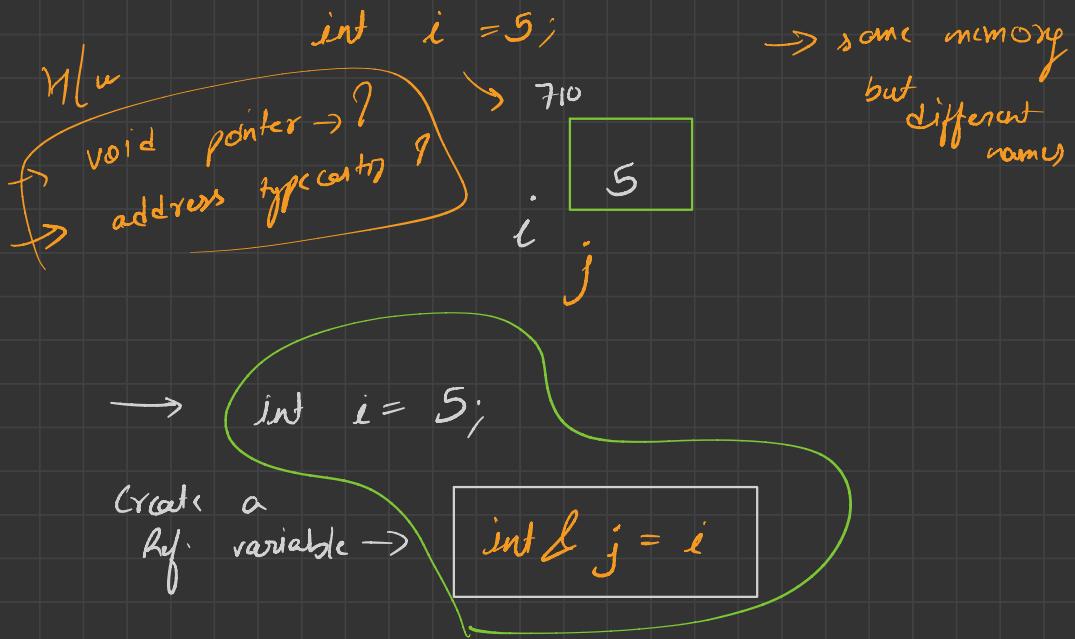
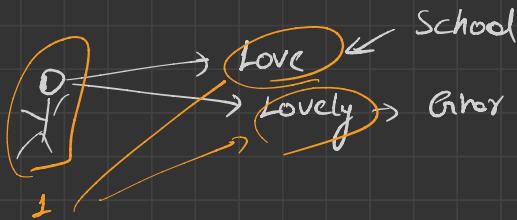
---

---

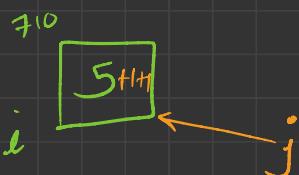


# Lecture - 28

## ① Reference variable



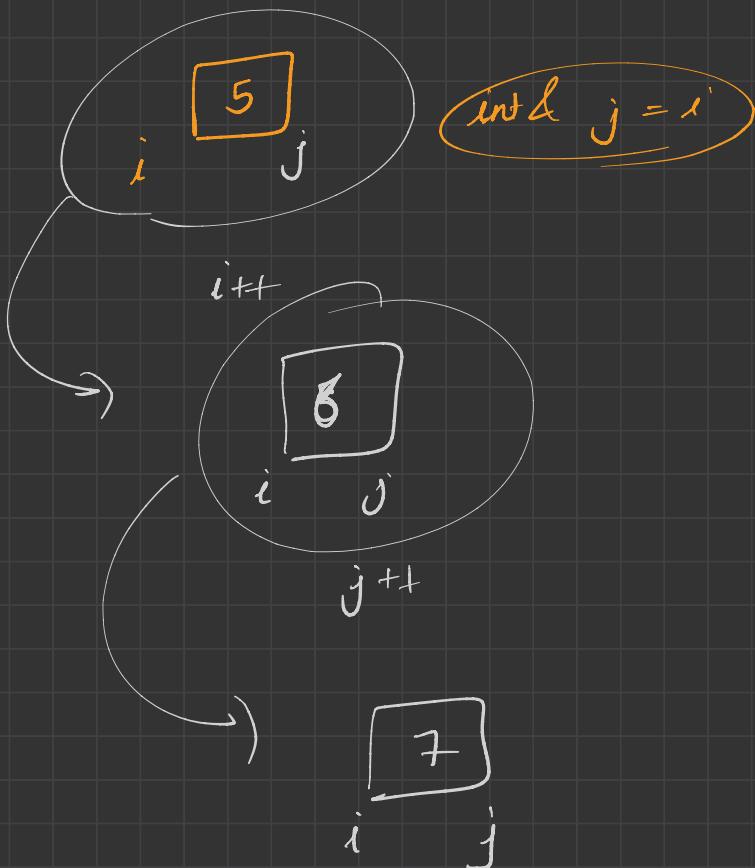
①



②

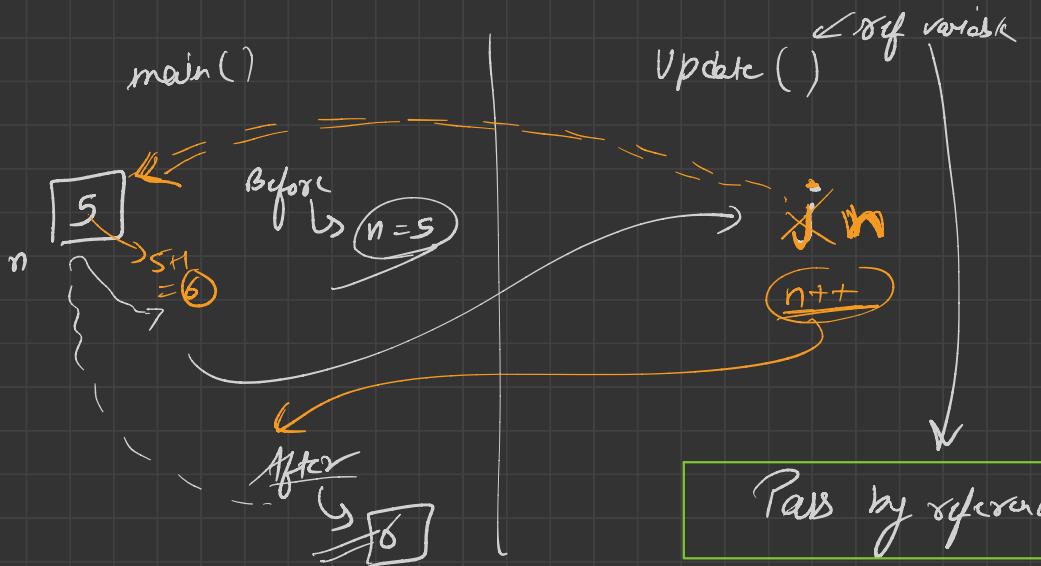
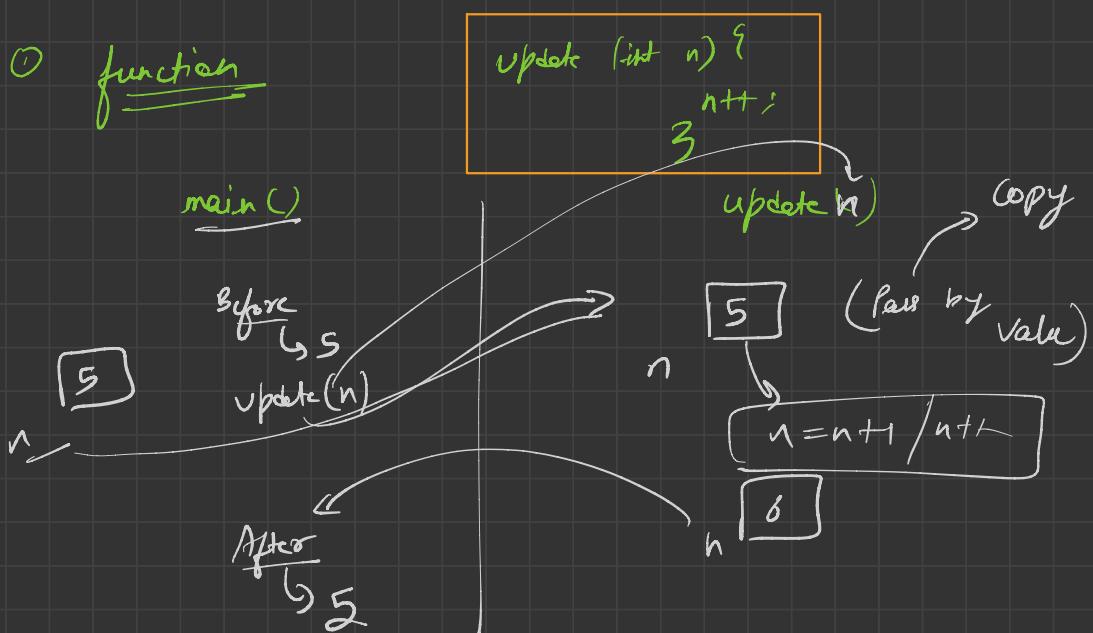
$i++$

$j++$

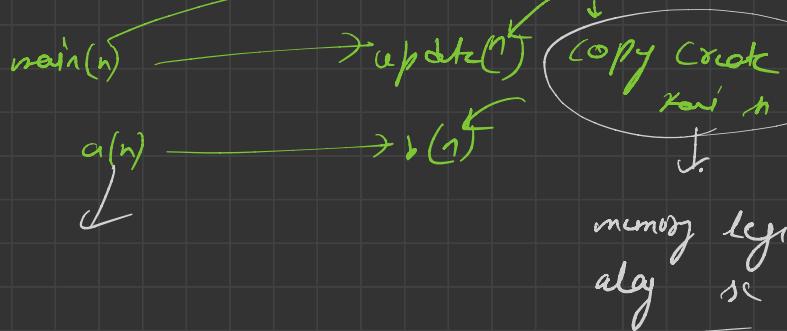


why - ?

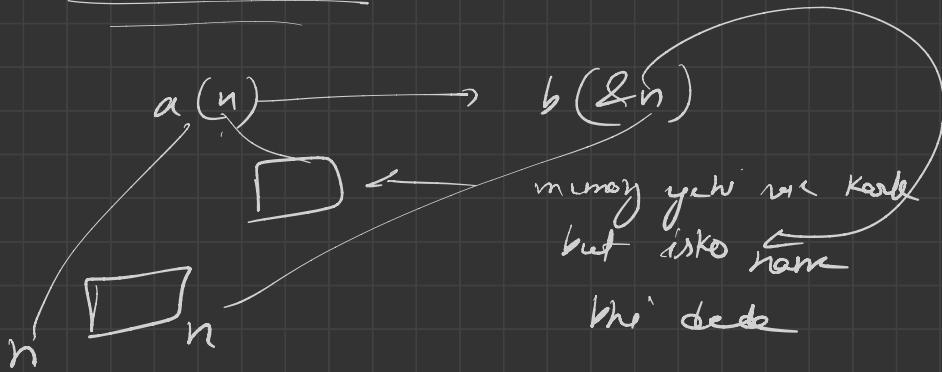
① function



Pars by value



Pars by reference



→ same memory

memory different how

void update (int & n)

{

}

int &

memory  
addr. ni krpao

pseudo krpao or ne  
class krpao

update (int n)

BAD

practice

int a = 10;

int l ans = a;

return ans;

3

Ref van - ?

↳ Pass by value ↳ Diff  
↳ Ref ↳ Diff

↳ func → Pass by ref

→ return by ref

Doubtback

→ Array

`int n;`

`cin >> n;`

`int arr[n];` → BAD practice

runtime → No

compile time -

→ why?

`int arr[10000];`

10000 → 32k

Program start

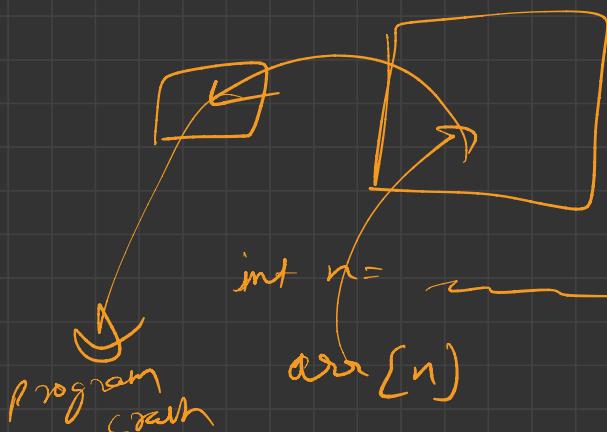
↓

memory

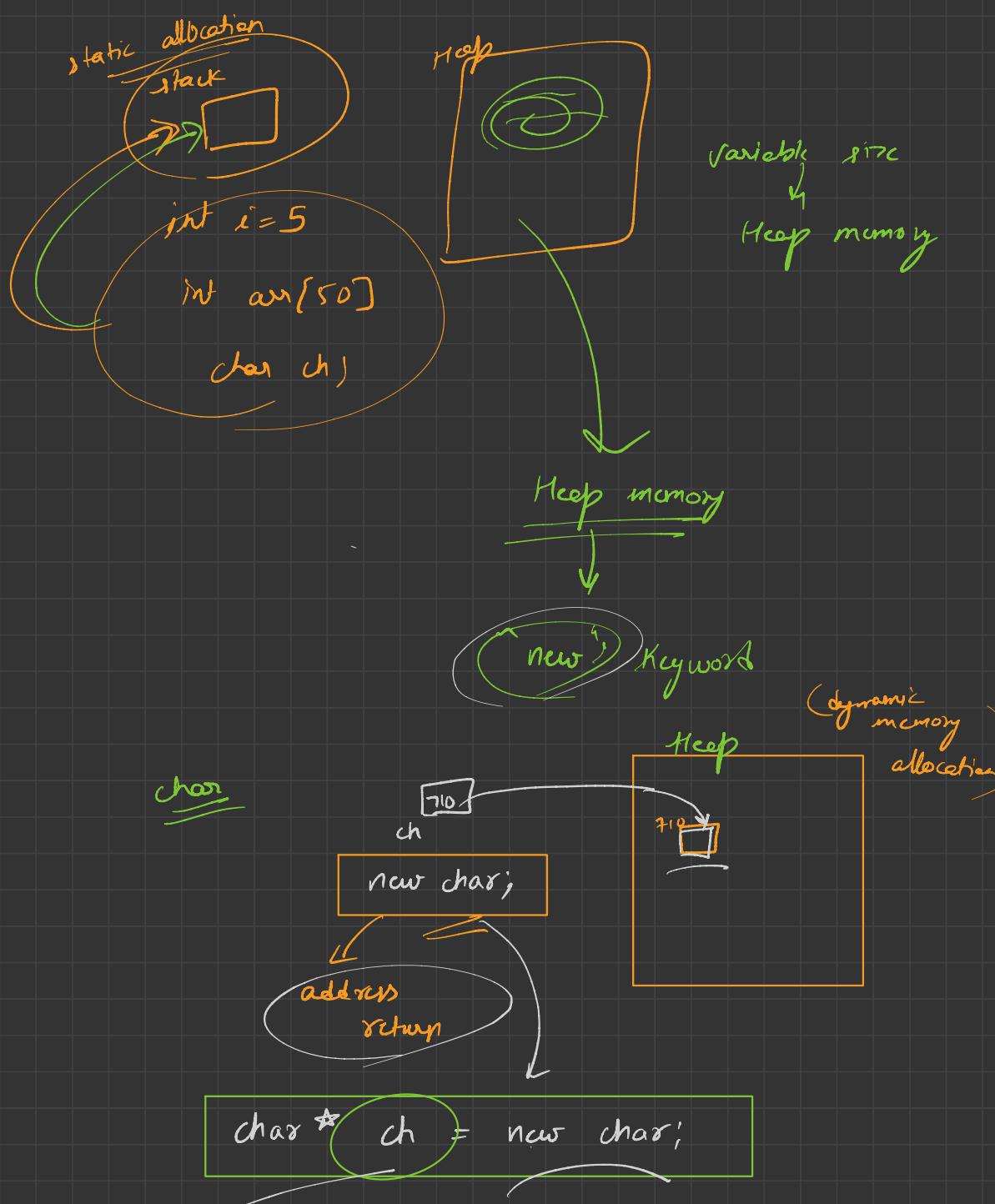
stack

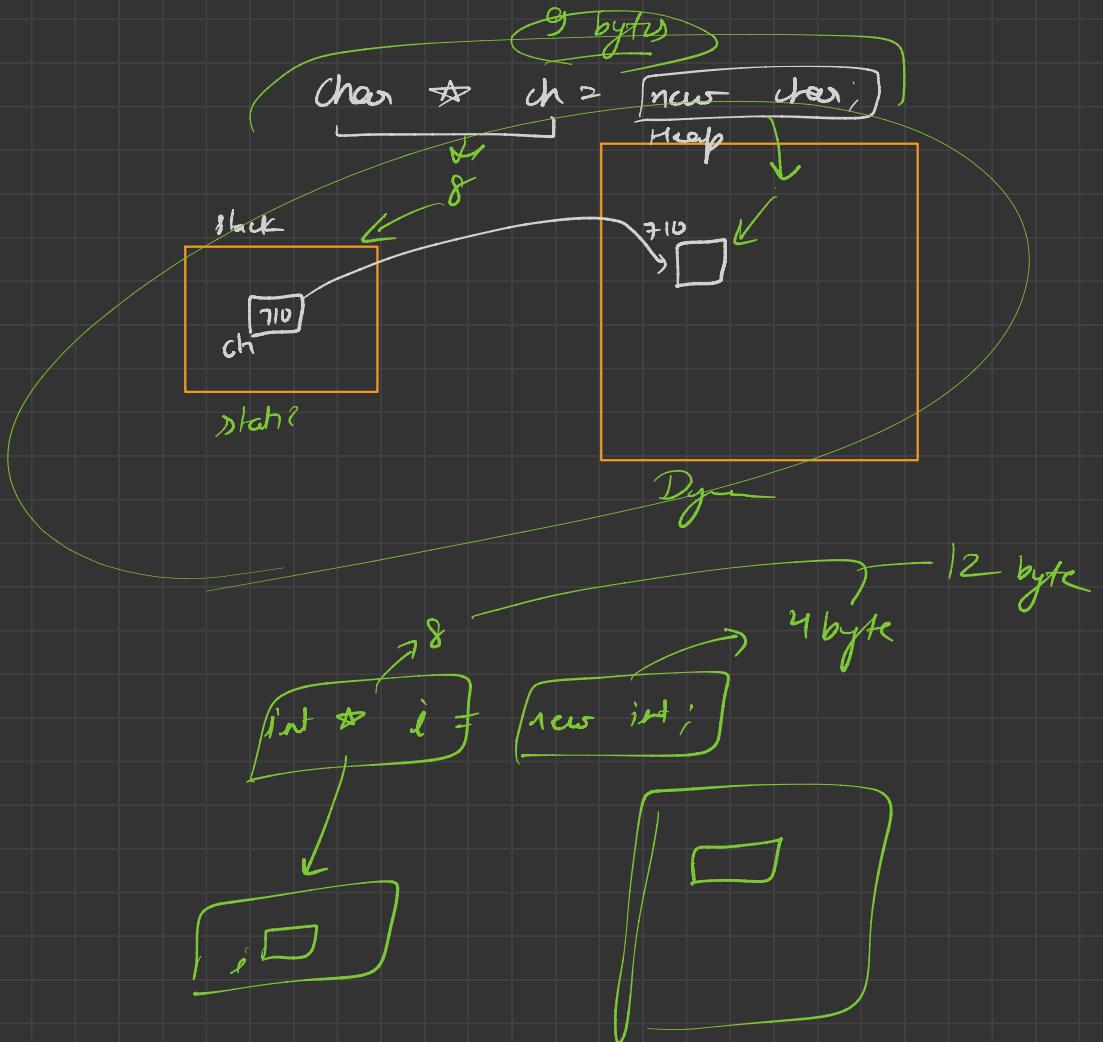
Heap

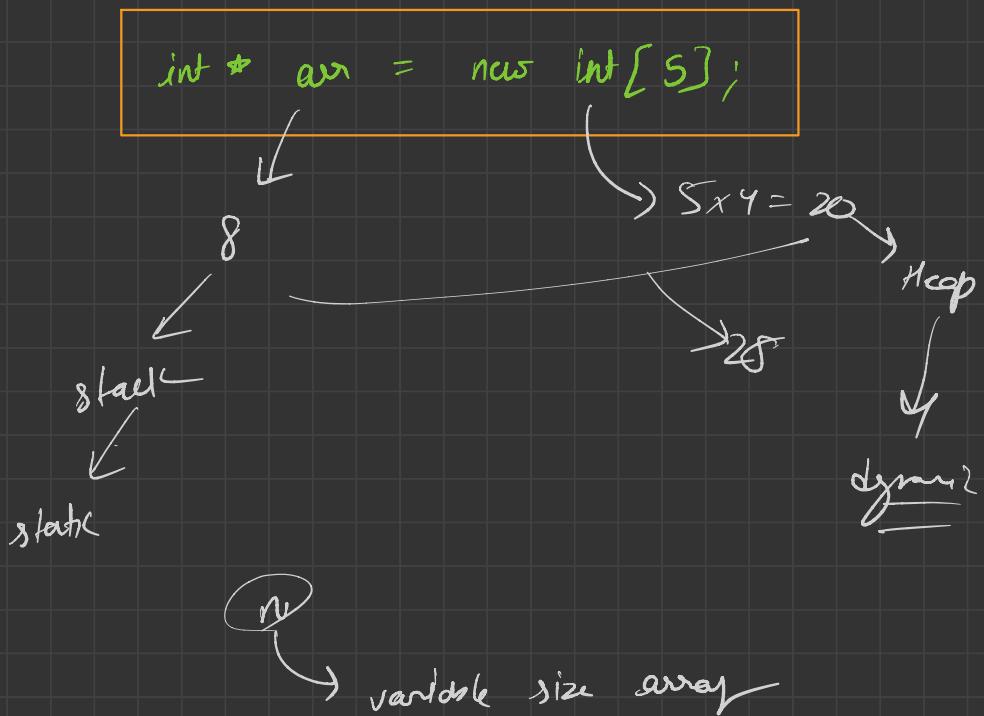
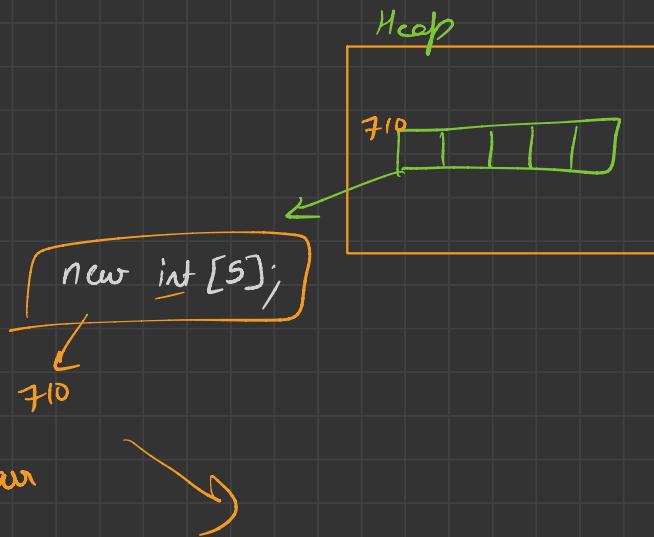
compile time

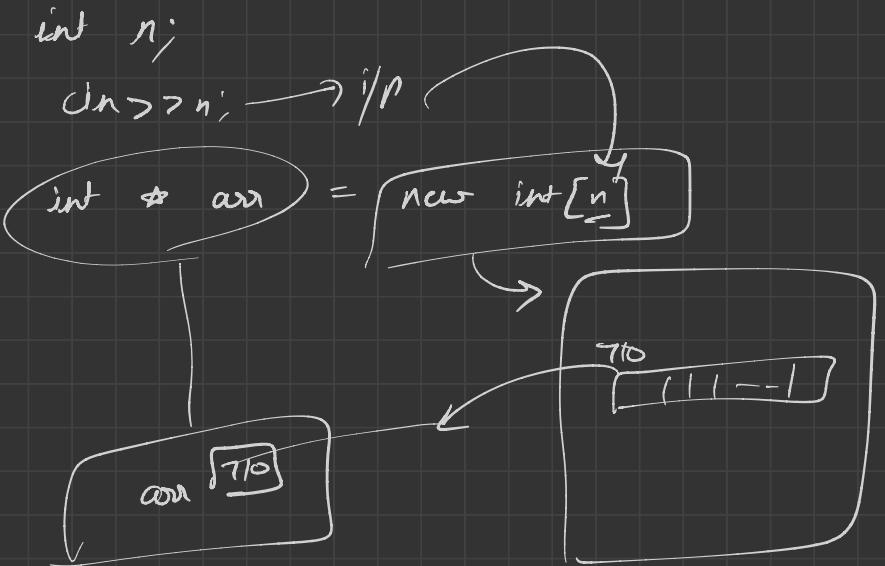


BAD practice









~~Ans~~

$$\text{arr}[i] = *(\text{arr} + i)$$

$\downarrow$        $\downarrow$   
 add      2



Diff b/w :-

Static

vs

Dynamically

①

int arr[50]



$$50 \times 4 = 200 \text{ bytes}$$

↓  
Stack

int arr = new int[50]



8 bytes

↓  
Stack

$$50 \times 4 \rightarrow 200$$

208

C/C -> L

while ( true )

{

int i = 5;

}

1 int memory

4 bytes

Call - 2

while ( true )

{

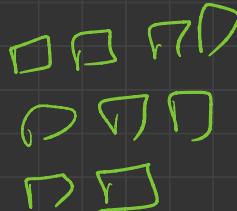
in \* p = new int;

3

8

4 4 4 4

loop



Program crash

static allocation

memory automatically released

Dynamic allocation

manually release

" delete " keyword

int

$\text{int} \Rightarrow i = \text{new int}$

for style  
element  
deletion

$\downarrow$

$\boxed{\text{delete } i}$

int arr[n]

arr

$\text{int} \star \text{arr} = \text{new int}[50];$

for array  
deletion

$\downarrow$

$\boxed{\text{delete } [\underline{\underline{\text{arr}}}] ;}$