

$a = 8$

✓ Sum $a+b$

✓ mult $a \times b$

Factorial

✓ 11

✓ 11

✓ 11

$b = 9$

Prime nu. check

✓ 11

✓ 11

✓ 11

a

b

$a+b$

$a-b$

a

b

$a+b$

$a-b$

a, b

a

b

$a+b$

$a-b$

✓ Odd b Even

✓ 11

✓ 11

✓ 11

a

b

$a+b$

$a-b$

```
int main () {
```

```
    int a, b
```

```
    cin >> a >> b
```

{

- ~~[fact of a]~~
- ~~[fact of b]~~
- ~~[" " a+b]~~
- ~~[" " a-b]~~

{

- ~~[a is odd or even]~~
- ~~[b " " " "]~~
- ~~[a+b " " " "]~~

~~[a-b is odd or even]~~ ↗

~~[Prime a]~~

~~[" b]~~

~~[" a+b]~~

~~[" a-b]~~

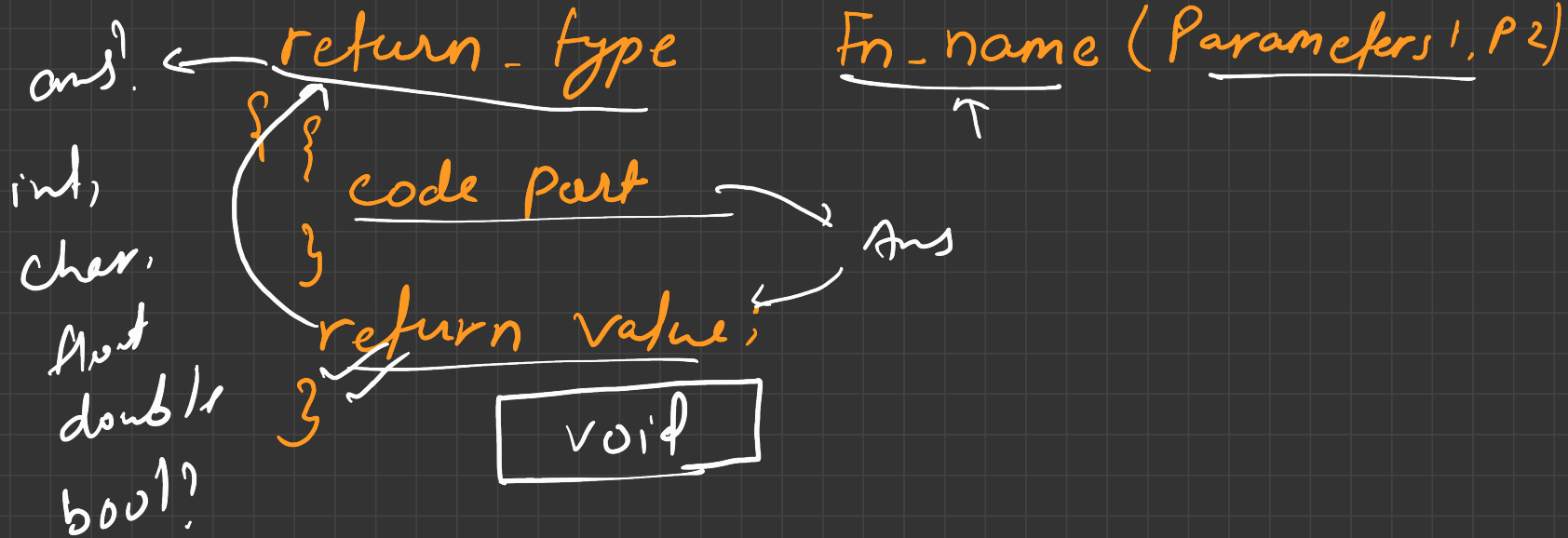
}

for (i=2; i < a; i++)
for (i=2; i < b; i++)

Reusability ↑

Readability ↑

How to make function (Fn)?



Factorial (int n) → declare of fn

int
{

int ans = 1;

for (i = 1; i <= n; i++)

{
 ans = ans + i;
}

return ans;
}

define of
function

bool Prime (int n) → declare

{ if ($n < 2$)
return 0;

for (i = 2, $i < n$; $i++$) {

if ($n \% i == 0$) {

return 0;

}

return 1;

}

define

```
int main () {
```

```
    int a, b
```

```
    cin >> a >> b ;
```

a int n

n = a * a

fn call → Prime (a) → Argument

```
    cout << Prime (a)
```

```
    cout << Factorial (a) << endl;
```

```
    cout << factorial (a + b)
```

a, b

```
int sum (int n , int m) {  
    int ans = a + b  
    return ans ;  
}
```

```
int main () {  
    cout << sum (a, b)  
}
```

① Pass by value

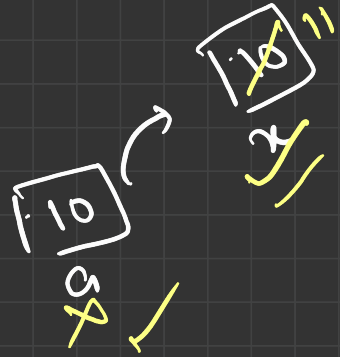
② Pass by reference?

① Pass by value

void

inc (int x)
x + 1;

int main() {
int a = 10;
inc(a);
cout << a;
}

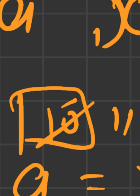
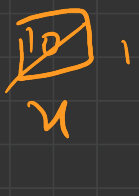


10

Pass by Reference

```
void Inc (int &x)
{
    x++;
}
```

```
int main()
{
    int a = 10;
    inc(a);
    cout << a;
```

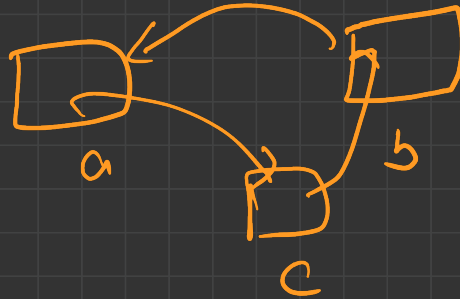
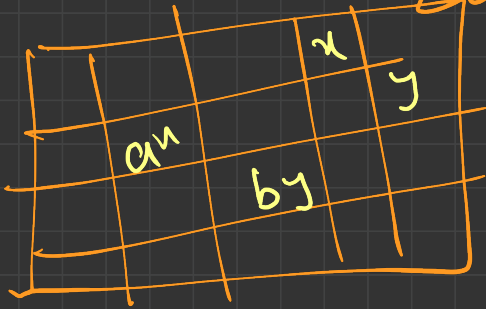
a, x  

Swap

```
int main() {
```

```
    swap(a, b)
```

```
    cout << a << b
```



```
void swap(int a, int b)  
{  
    int c;
```

```
    c = a
```

```
    a = b
```

```
    b = c
```

```
}
```

In overloading

```
void swap (float &c, float &d) {  
    float r = c;  
    c = d;  
    d = r;  
}
```