Parameter Passing Methods

Pass by value :

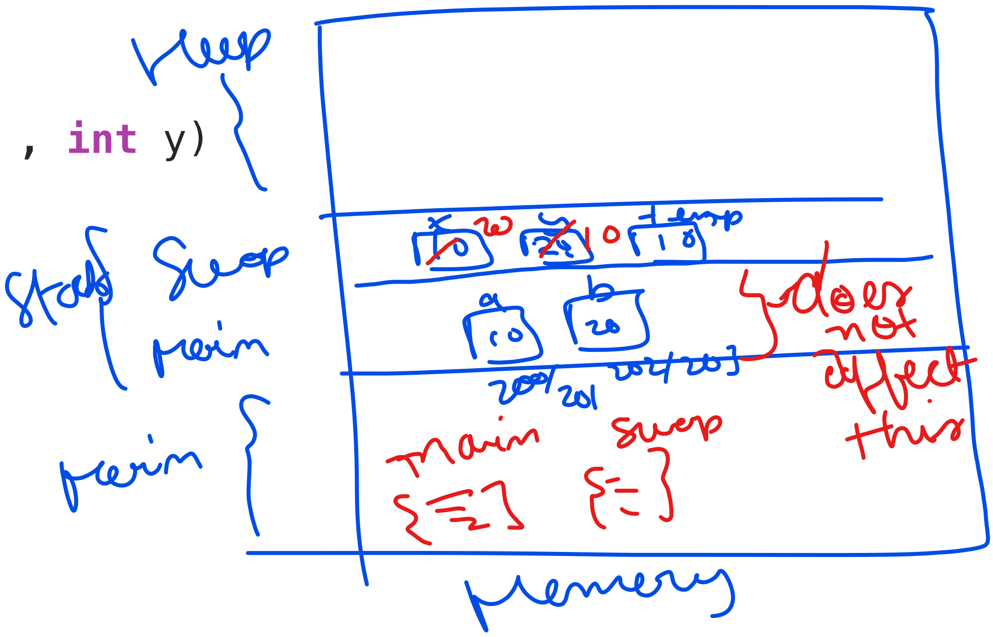
↳ Return stufs

- In pass by value actual parameters will not be modified if any changes are done to the formal parameters

Example :

```
void swap(int x , int y)
{
    int temp;
    temp = x;
    x = y;
    y = temp;
}

int main()
{
    int a ,b;
    a =10;
    b=20;
    swap(a,b);
    printf("%d %d", a ,b);
}
```



Frequent

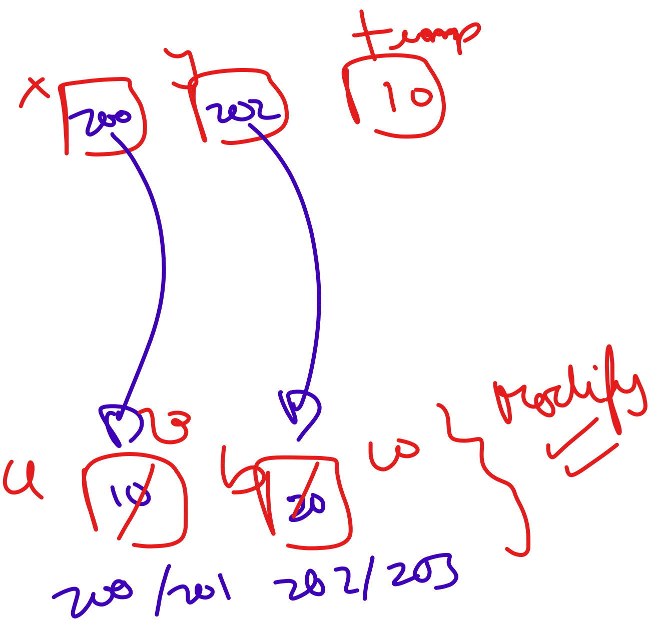
Call by address :

- Here the address of actual parameters are passed to formal parameter and formal parameters must be pointers

- Any changes done inside function will modify the actual parameters

```
void swap(int *x , int *y)
{
    int temp;
    temp = *x;
    *x = *y;
    *y = temp;
}

int main()
{
    int a,b;
    a =10;
    b=20;
    swap(&a,&b);
    printf("%d %d", a,b);
}
```



- One function cannot access value of another function directly but it can access it indirectly through pointers

- Thus call by address is a suitable mechanism for modifying actual parameters

Call by reference :

A) not advirable :(

B) lun memory

func code is copied to the main function during function call.
dont use it at heavy functions

here source code is procedural,
but machine code is monolithic.

- References are part of c++ programming, its one of the useful and powerful mechanism of this language

- To make a function as call by reference we just need to add & in the parameters, these are the references

```
void swap(int & x , & int y)
{
    int temp;
    temp = x;
    x = y;
    y = temp;
}

int main()
{
    int a ,b;
    a =10;
    b=20;
    swap(a ,b);
    printf("%d %d", a ,b);
}
```

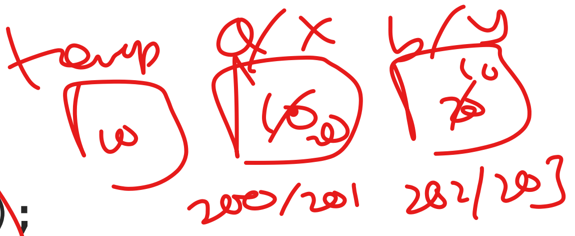
Just here only :)

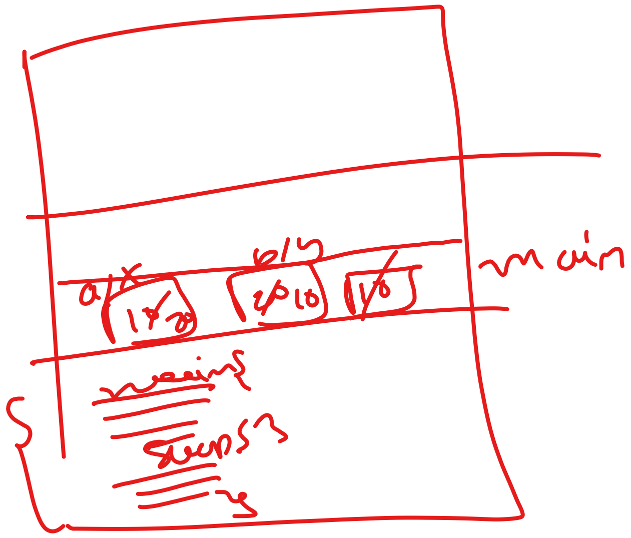
here ,swap is became part of the ,main function,thats why it can access its variables directly...

swap is not seperate fuction

A

B no change





swap is inside main