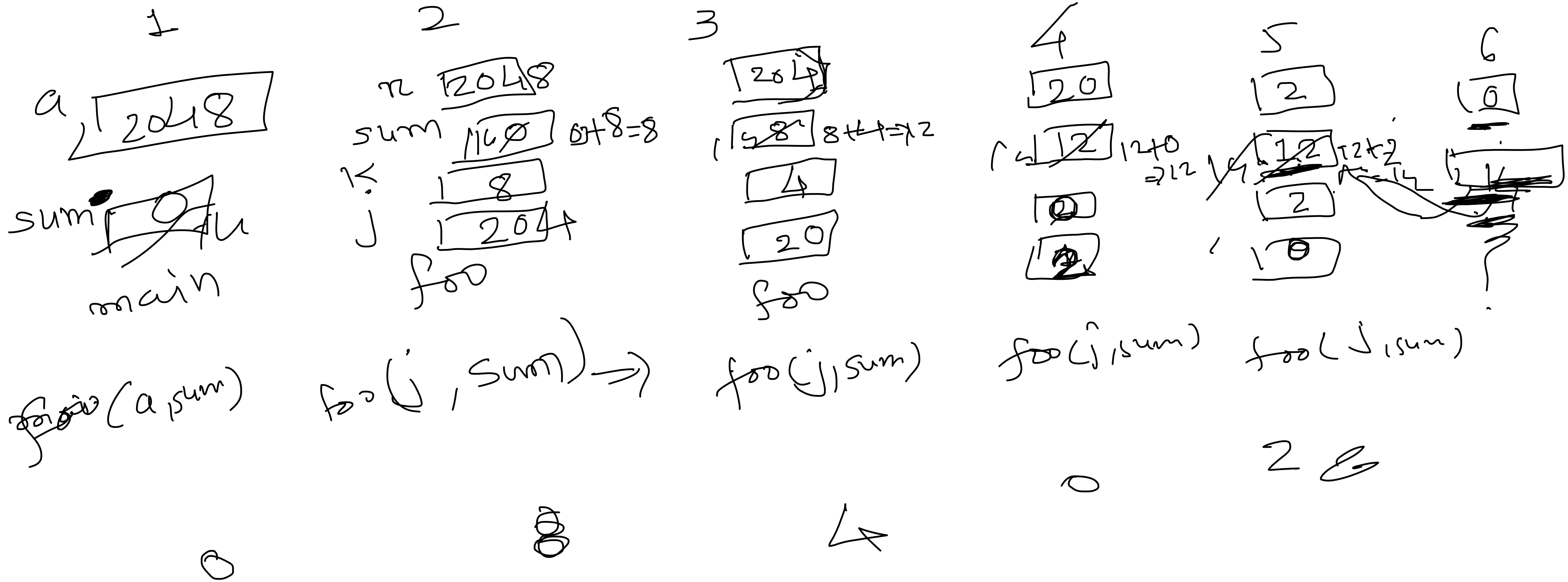
Let’s solve it

```
void foo (int n, int sum)
{
    int k = 0, j = 0;
    if (n == 0) return;
    k = n % 10;
    j = n / 10;
    sum = sum + k;
    foo (j, sum);
    printf ("%d", k);
    return (sum);
}
```

```
1  int main()
2  {
3      int a = 2048;
4      int sum = 0;
5      sum = foo (a, sum);
6      printf ("%d", sum);
7      return 0;
8  }
```


1
2
3
4
5
6
a
2048
sum
main
foo(a, sum)
sum
j
foo
foo(j, sum)
foo(j, sum)
foo(j, sum)
foo(j, sum)
0+8=8
20
12
2

20480

main 5th/6t

sum = foo (a, sum) ;

output will be

2 0 4 8 1 4 8

int foo ( )

{

if ( ) return (sum) ;

;

sum = foo ( )

return(sum);

int a=10 b=20;

printf("a is %d", b);

printf("b is %d", a);

---

smart swapping

1st std -

char *pa, float *pa, dou *pa;

```
1 void swap (int *pa, int *pb)
2 {
3
4     int temp;
5     temp = *pa;
6     *pa = *pb;
7     *pb = temp;
  }
```

declaring a pointer

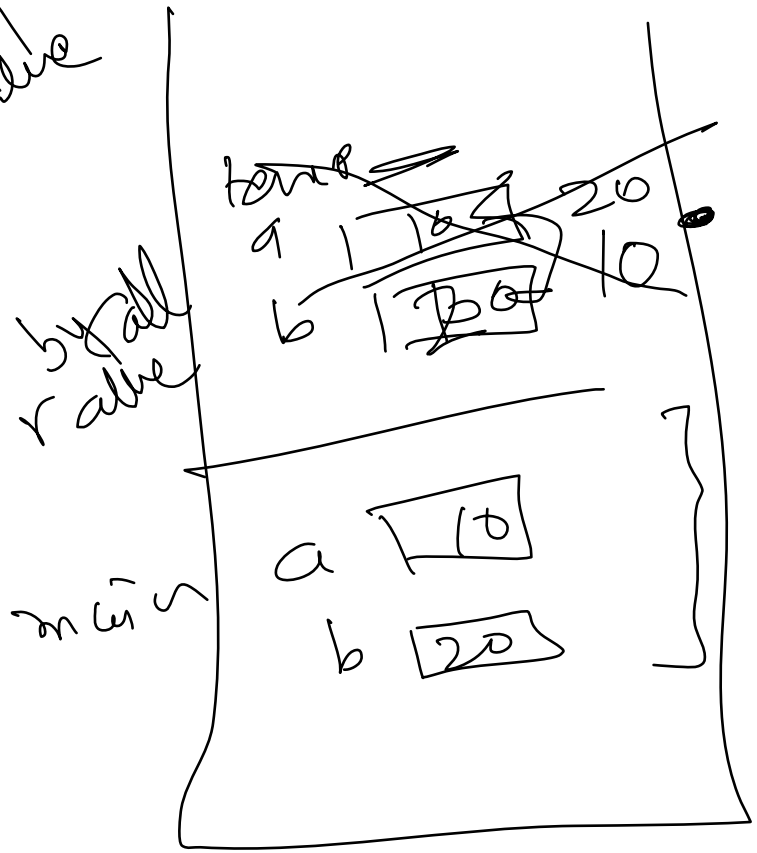
reading data from location pointed by pa

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

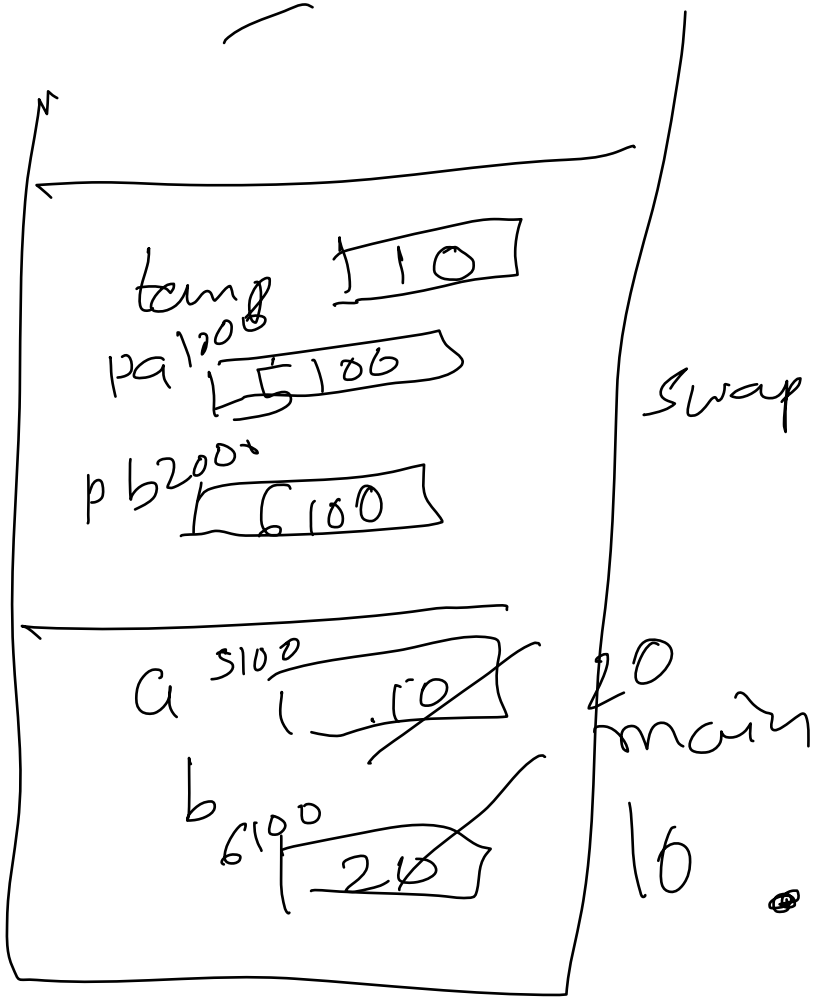
without pointer

Copy

pass by value



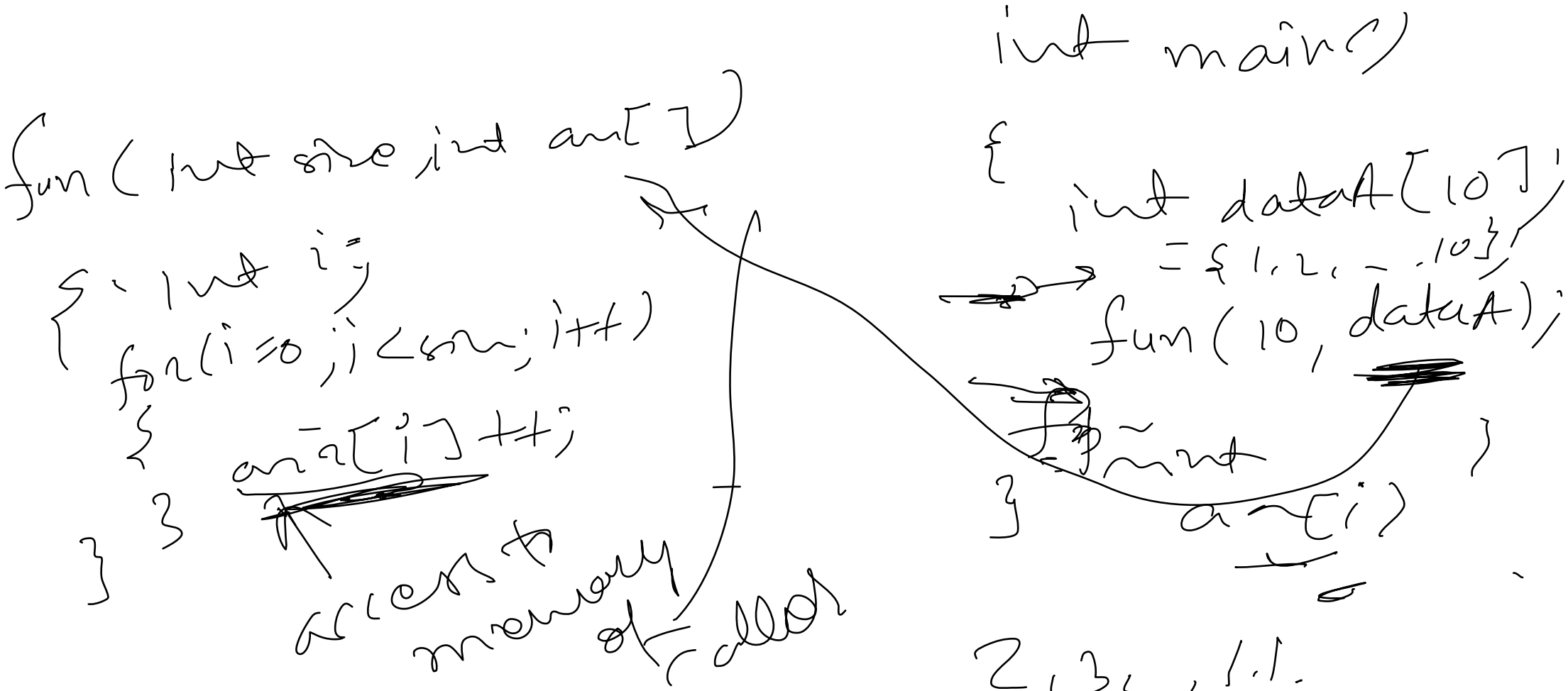
with pointer



Passing array as an argument

With the use of pointers, called function can access memory of caller function variables.

Array name is constant pointer, when you pass array as argument, you are passing only address.


fun(int size, int arr[])
{ int i;
for(i=0;i<size;i++)
{
arr[i]++;
}
}
arr is to memory of caller
int main()
{
int data[10]
={1,2,...10};
fun(10, data);
}
a[i]
2, 3, , 1.1.