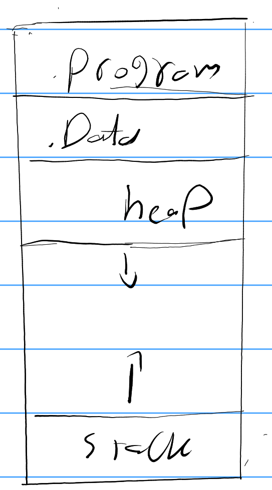
C#2

func() {

Push()

datatype_t var_name = val; → Stack

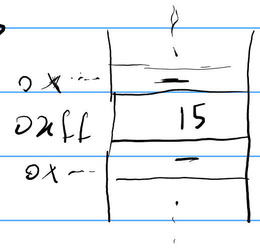
} → pop()



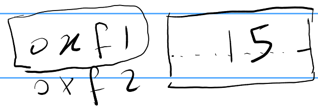
Pointers

short int x = 15

int z = 15

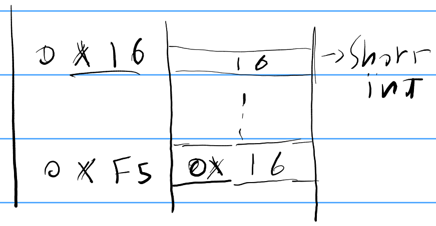


type_t <var_name = val;



short int ab = 16;

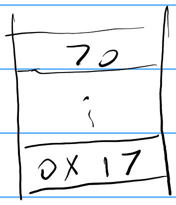
short int *ptr = &ab;



int ac = 70;

int *ptr2 = ∾

ac → 0x17



*ptr2 → ??

*ptr2 → printf("%d\n", *ptr2); → 70 \n

*ptr2 = 50   printf("%d\n", ac); → 50\n

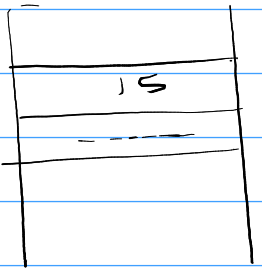
pointers op

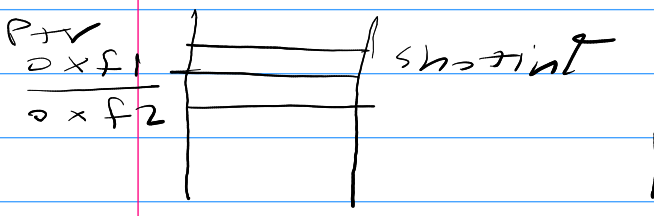
```
short int x = 15
short int *ptr = &x;
ptr += 1;
```

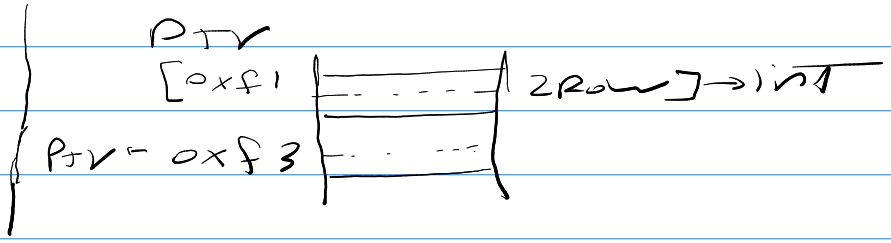
x → 0xf1

0xf2

ptr = 0xf2







int
short

*ptr → 16

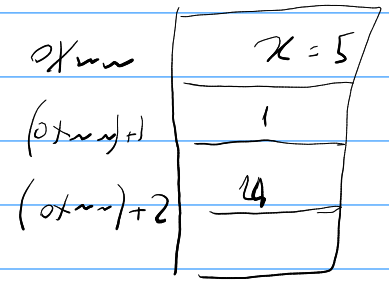
---

int x = 5

int *ptr = &x;

*(ptr+1) = 1;

*(ptr+2) = 4;



int arr[3] = {5, 1, 4}

*(arr+1) ← 1
arr[1] ← 1

*(arr+2) ← 4
arr[2] ← 4

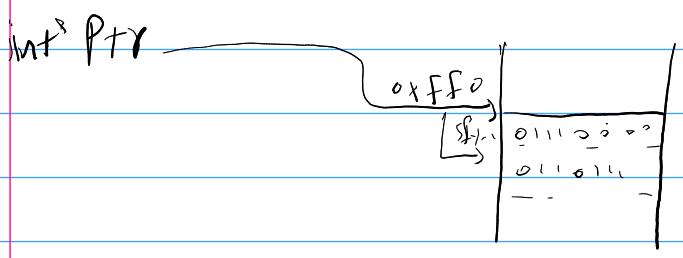
*(arr) ← 5
arr[0] ← 5

ptr[0] ← 5
|||
*(ptr+0) ← 5
*(ptr) ← 5

ptr[1] ← 1
|||
*(ptr+1) ← 1

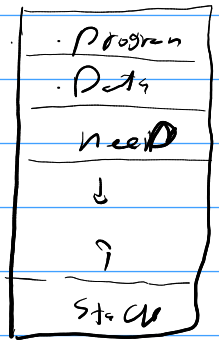
ptr[2] ← 4
|||
*(ptr+2) ← 4

(void *)



malloc
↳ void *

1- runtime memory managment



2- function generating of memory

3- Stack < heap

---

ptr = malloc (size) ⟶ free(ptr)

There is no garbage collector in C]] manually