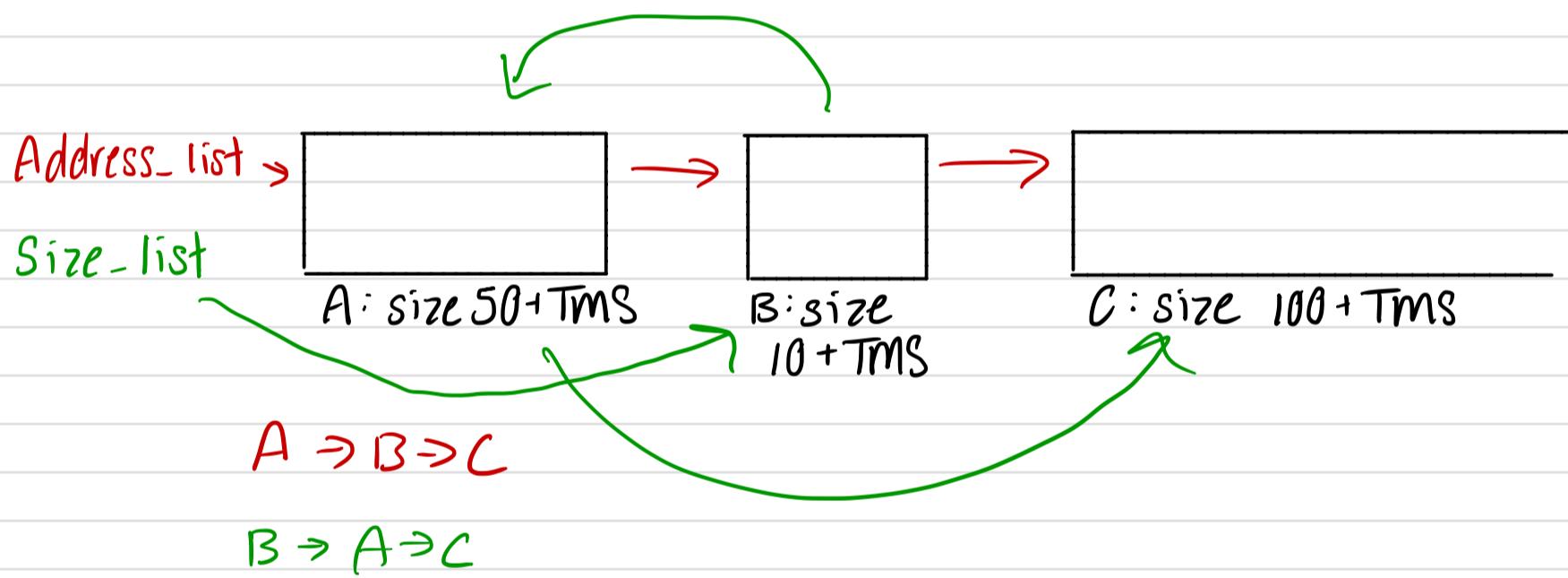
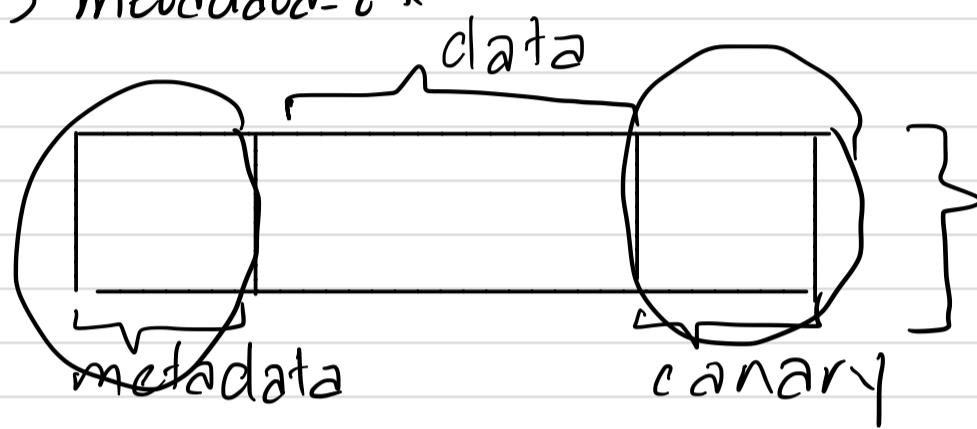


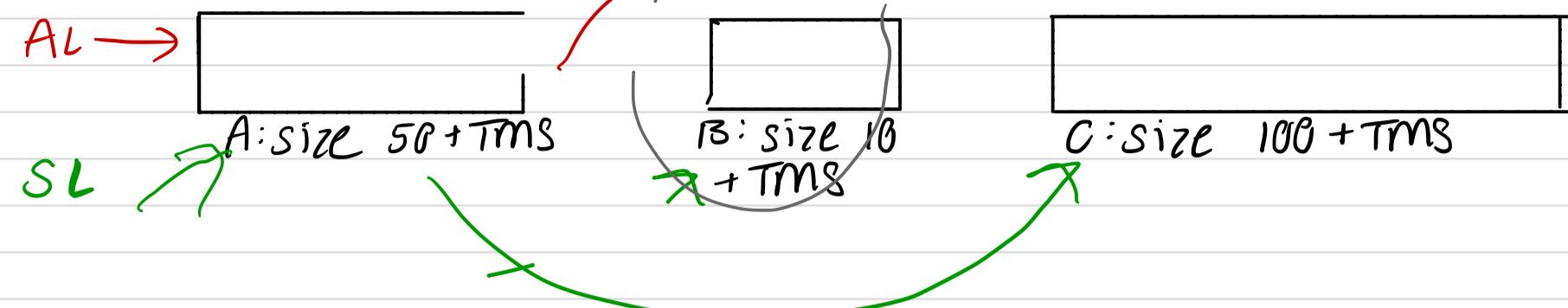
Malloc Precitation

Part A: The Block

```
typedef struct metadata {  
    unsigned long canary,  
    unsigned long size,  
    struct metadata * next-addr,  
    struct metadata * next-size,  
    struct metadata * prev-addr,  
    struct metadata * prev-size,  
} metadata_t*
```



Part B: Malloc



Case 1: my-malloc(10)

- ① Add TMS to user requested to get size of block actually want
 $10 + \text{TMS}$
- ② Iterate through size list to find block of best size
- ③ Once we find our perf block remove from address + size lists

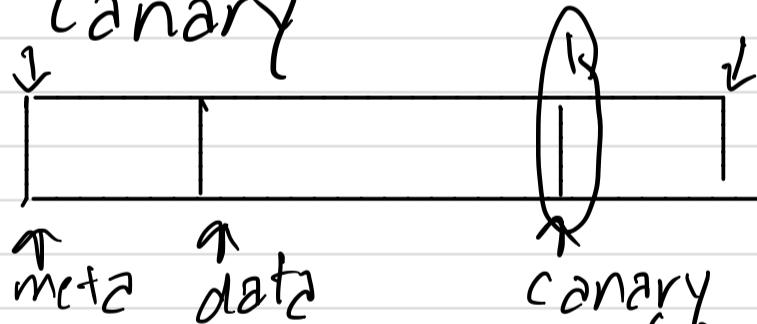
$$B \rightarrow p_2 \rightarrow n_2 = B \rightarrow n_2$$

$$B \rightarrow n_2 \rightarrow p_2 = B \rightarrow p_2$$

$$B \rightarrow p_s \rightarrow n_s = B \rightarrow n_s$$

$$B \rightarrow n_s \rightarrow p_s = B \rightarrow p_s$$

- ④ Set Canary

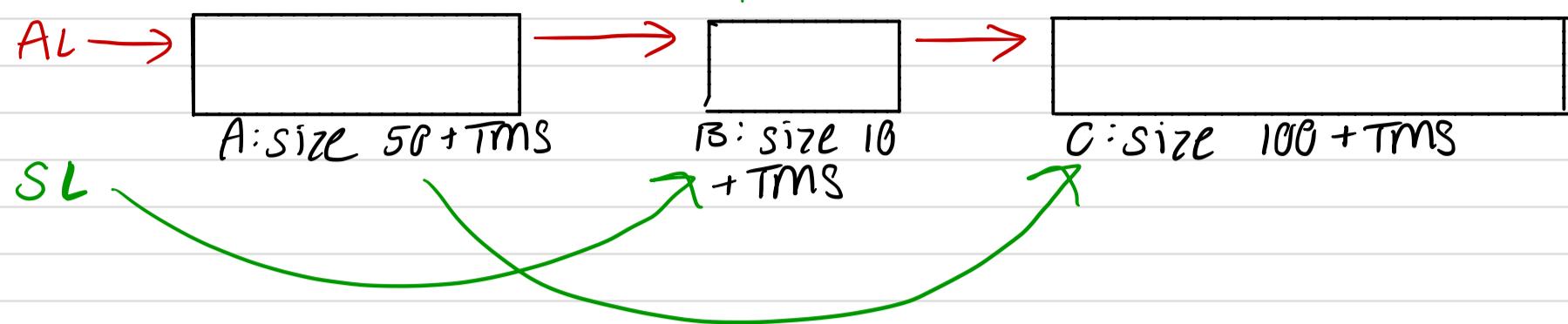


$B \rightarrow \text{canary} = \text{lovely function}$

$(\text{uint8_t} * B + B \rightarrow \text{size} - \text{sizeof}(\text{unsigned long}))$

- ⑤ Return pointer to user data
 $\text{return } B + 1$

Part B: Malloc



Case 2: my-malloc(40)

Note: consider min-block-size = 5

① $40 + TMS$

split block(!)

$40 + TMS + \text{MIN-BLOCK-SIZE}$

($1 + TMS$)

②



A left
in the
block

pointer return

$40 + TMS$

→

③a) set canaries

③b) proper spot in
address list!

③c) return a pointer
to the user data

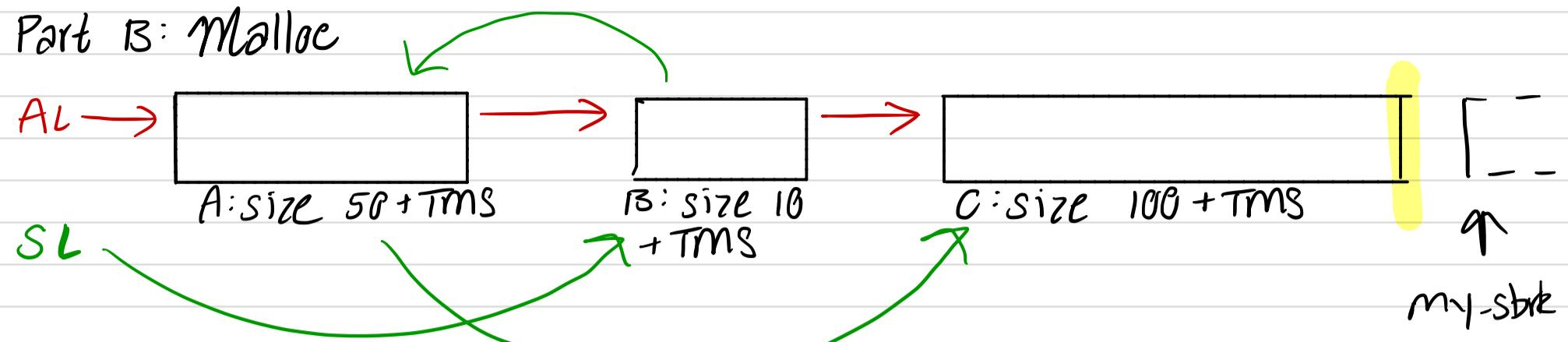
④b) remove size list

⑤a) null out all pointers

④c) $A \Rightarrow \text{size} = A \Rightarrow \text{size} - 40 + TMS$

④d) Reenter into size list

Part B: Malloc



Case 3: `my-malloc(97)`

Note: consider min-block-size = 5

① $97 + \text{TMS}$

$$97 + \text{TMS} + \text{MBS}(5) > C \rightarrow \text{size}$$

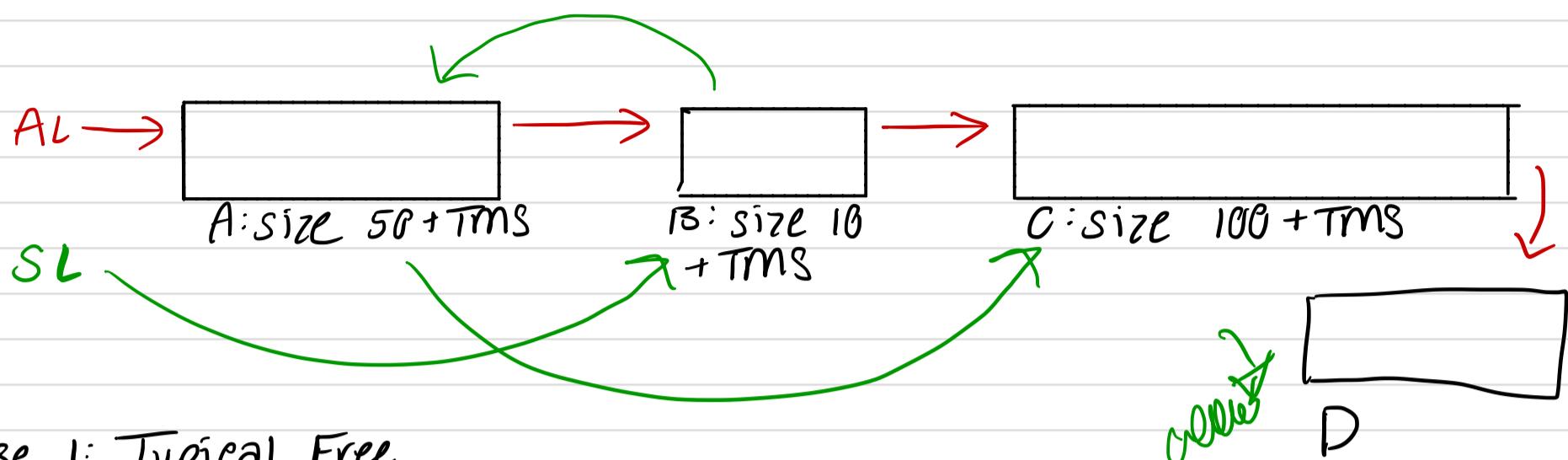
ONLY WAY TO SPLIT

$$C \rightarrow \text{size} > \text{User-req} + \text{TMS} + \text{MBS}$$

② Call `my-sbrk!`

③ Re-go through the entire size-list
to find block we want to return
to user

Part C: Free

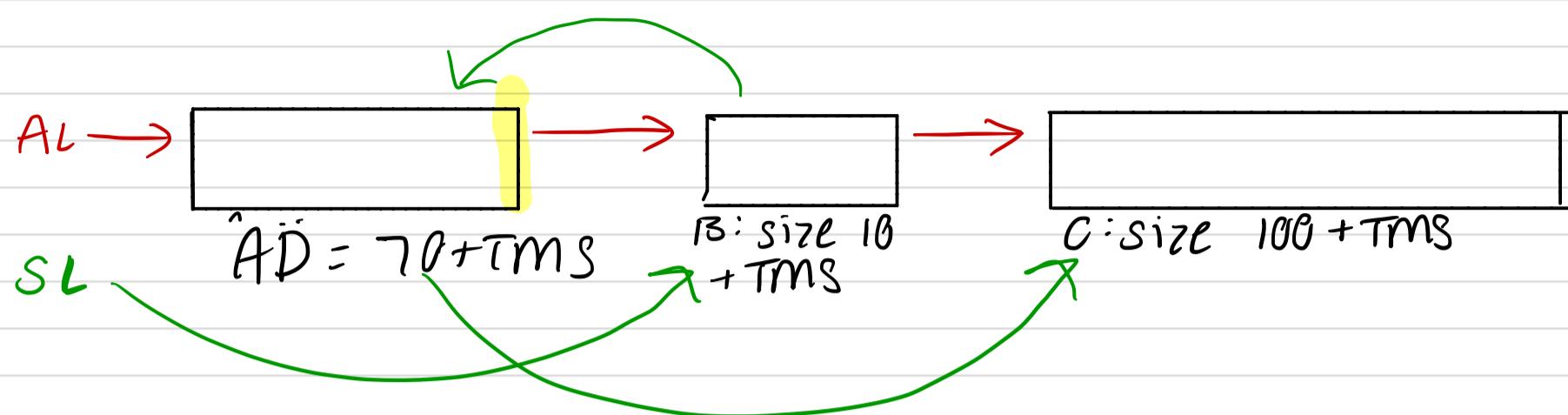


Case 1: Typical Free

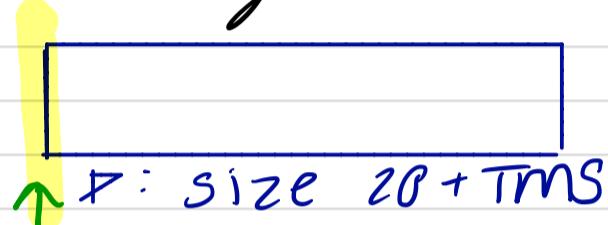


- ① get ptr to metadata of block
 $(\text{metadata_t}^*)\text{ptr} - 1$
- ② check canaries
↳ if bad set CANARY-CORRUPTED & return
- ③ iterate through addr list to find proper place of our block
 (vintptr_t^*)
^ allows us to compare pointers!
- ④ Add block to size list if no merging

Part C: Free



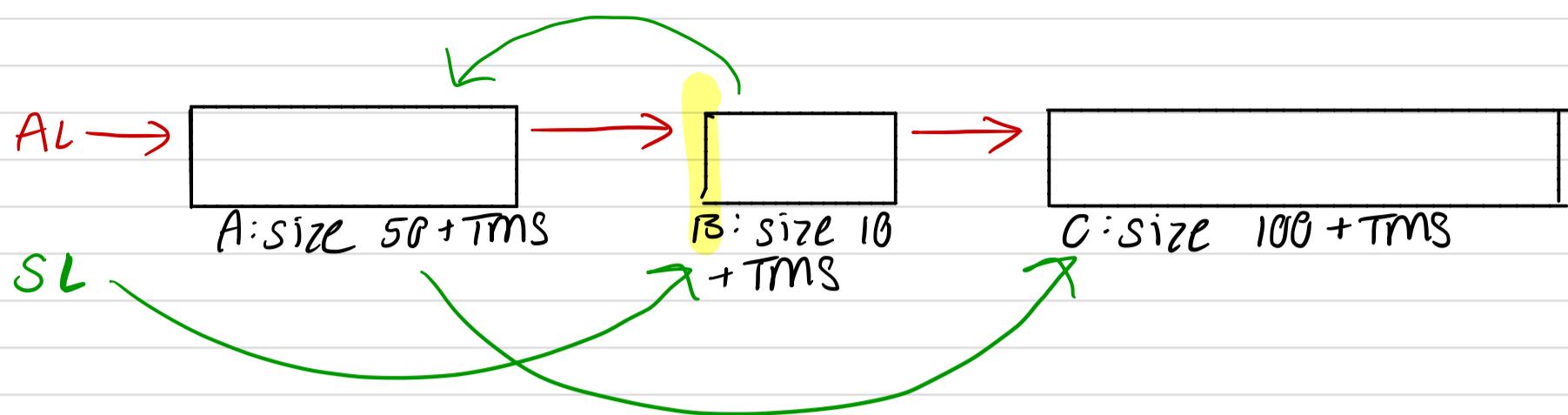
Case 2: Left Merge



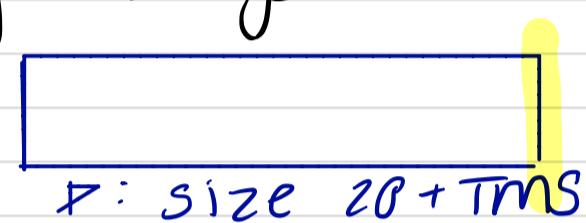
$$A + A \Rightarrow \text{size} = D$$

- ① Keep A in the address list where it is
- ② $A \Rightarrow \text{size} = A \Rightarrow \text{size} + D \Rightarrow \text{size}$
- ③ Remove A from size list & reenter A in proper location

Part C: Free



Case 3: Right Merge



$$P + D \Rightarrow \text{size} = B$$

$$\textcircled{1} \quad D \Rightarrow pa = B \Rightarrow pa$$

$$D \Rightarrow na = B \Rightarrow na$$

$$B \Rightarrow pa \Rightarrow na = D$$

$$B \Rightarrow na \Rightarrow pa = D$$

$$D \Rightarrow \text{size} = B \Rightarrow \text{size} + D \Rightarrow \text{size}$$

Remove B from size list

Add DB into size list

Part D: Realloc
my-realloc(ptr, size)

Case 1: Pointer null

ret my-malloc(size)

Case 2: Pointer non-null, canaries corrupted

errno = CANARY CORRUPTED
return null

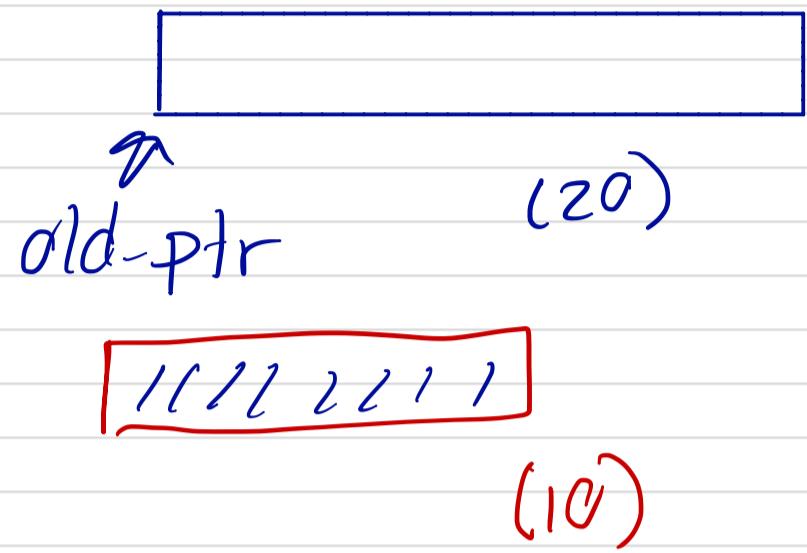
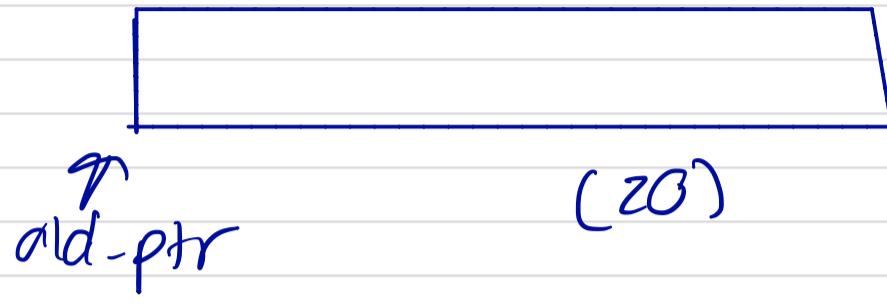
Case 3: Size == 0, Pointer non-null

free(ptr)
return null

Part D: Realloc

Case 4: Pointer non-null & size != 0

new-ptr = my-malloc(size)

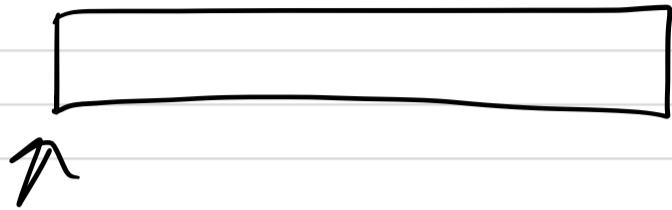


Highly suggest checking out memcpy

my-free (old-ptr)
return new-ptr

Part E: Calloc

my-malloc



Set everything in the user portion
of block to 0

highly suggest checking out memset