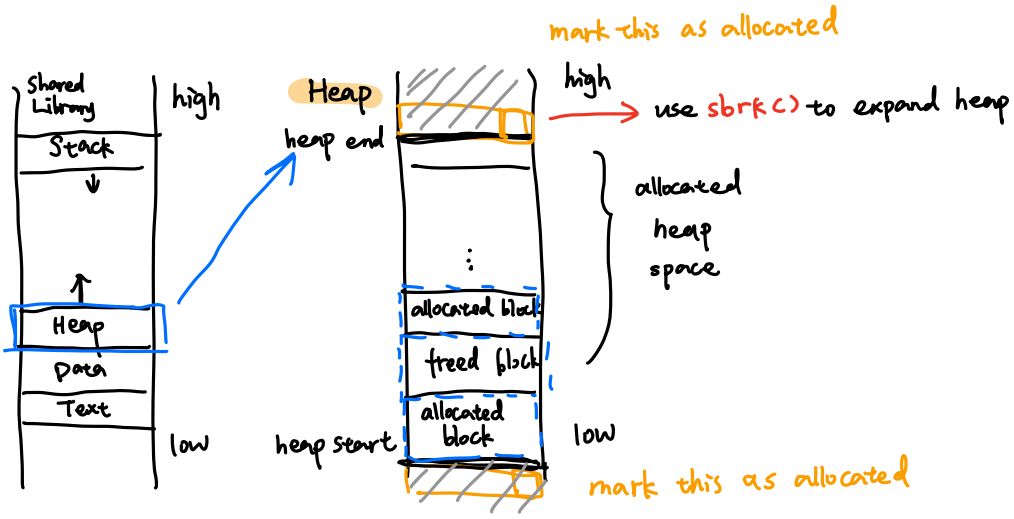
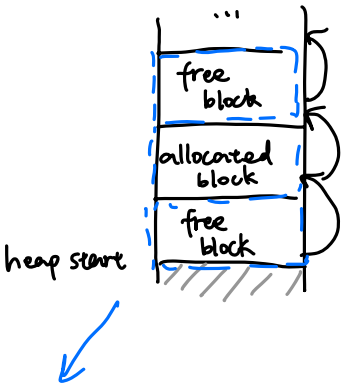
Malloc LAB

Memory Structure

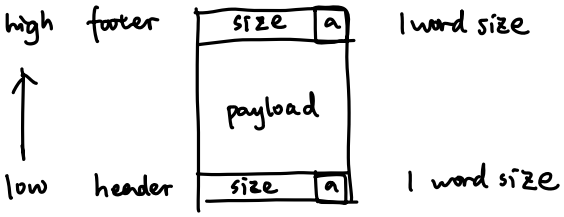


Implicit List:

all block are linked



Block Design



1 word size

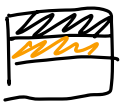
a for current block allocated status
0: free , 1: allocated

1 word size

Coalescing

① 

change current block header & footer status

② 

calculate new block size
change current block header size & status

③ 

calculate new block size
change previous block header size & status

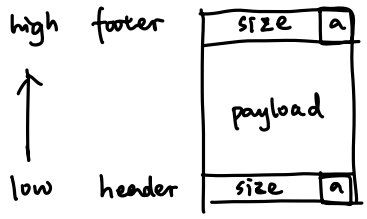
④ 

calculate new block size
change previous block header size & status

Explicit List

introduce a free_list to store the free blocks

allocated block (same as previous)

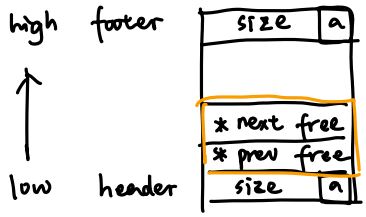


1 word size

1 word size

a for current block allocated status
0: free , 1: allocated

* free block design

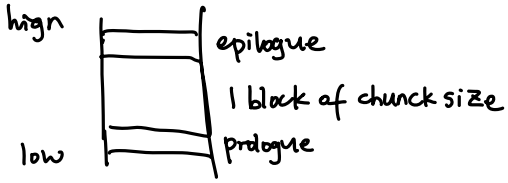


1 word size

1 word size

a for current block allocated status
0: free , 1: allocated

initializing



* prev free = NULL
* next free = NULL

(free_head)
init

(NULL) → [ ] → NULL

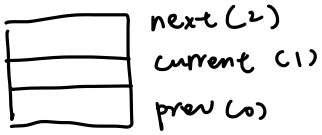
malloc

splited.prev = ori.prev
splited.next = ori.next

if (first_block)

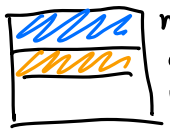
free optimized

init



new_block = block 1
new_size = get_size (block 1)

case 1

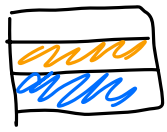


next (2)
current (1)
prev (0)

when block 0 is free.

new_block = block 0
new_size += get_size (block 0)

case 2



next (2)
current (1)
prev (0)

when block 2 is free

new_size += get_size (block 2)

last

write_block (new_block, new_size, free)

Eliminating footer

allocated block

ensure that when this block is freed, it still has space to store the 2 free pointers, avoiding overwrite the next block

high footer

payload

at lease 3 word size

low header

size | b | a

1 word size

a for current block allocated status

b for previous block allocated status

0: free , 1: allocated

free block

high footer

size | b | a

1 word size

a for current block allocated status

b for previous block allocated status

0: free , 1: allocated

* next free
* prev free

low header

size | b | a

1 word size

why?

when coalescing, we need to search previous block's alloc status.

considering we do not have footer, and we are currently at the heap start, which means that the previous' block is prologue, where we could not access it's header for info, so we need a bit to record prev. block's status.