Discussion of MS2 - Part II

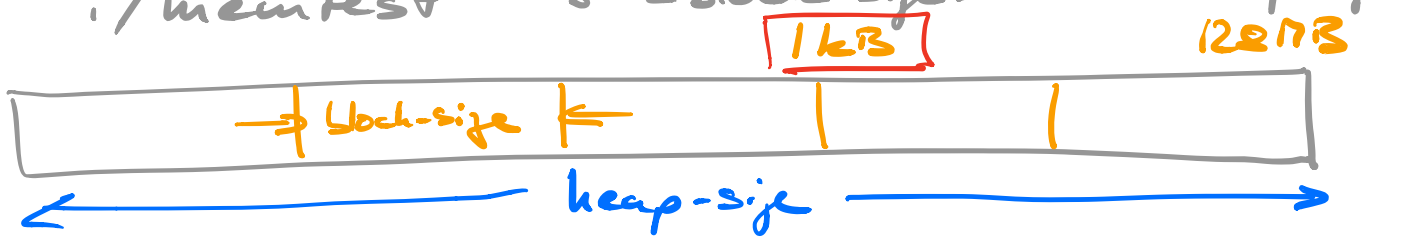
1- block sizes

2- segment headers: use + complications

3- walk through the implementation of MS2

1.) Block Sizes

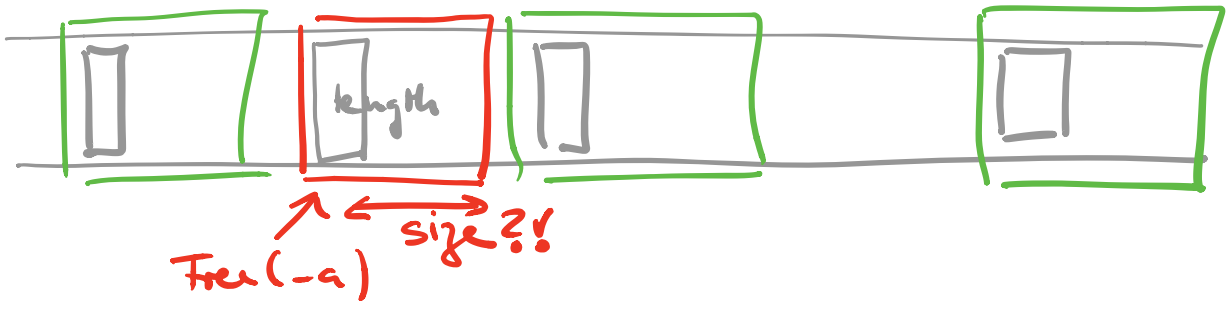
./memtest -b <block-size> -s <heap-size>



- memory is allocated in multiples of block-size

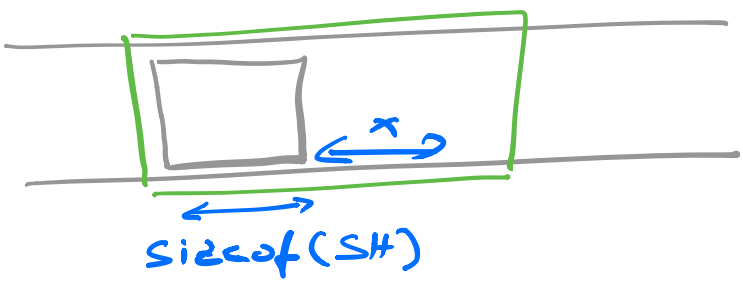
[Malloc (3.5 kB) → 4kB]

- **Segment Headers**



Further 2 small complications

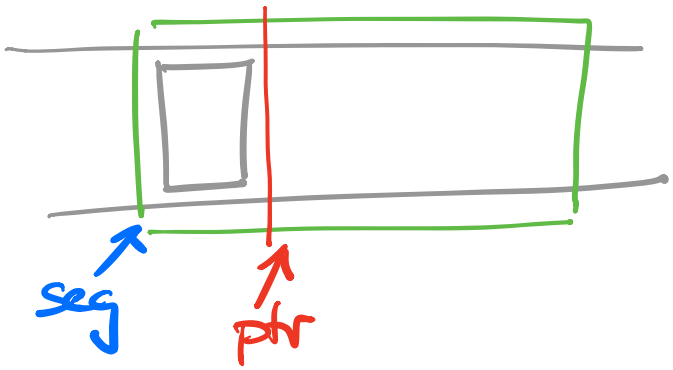
1. **Allocate:**



sizeof(SH) + x bytes + round up!

2. Allocate / Free

ptr ← Malloc(_length)

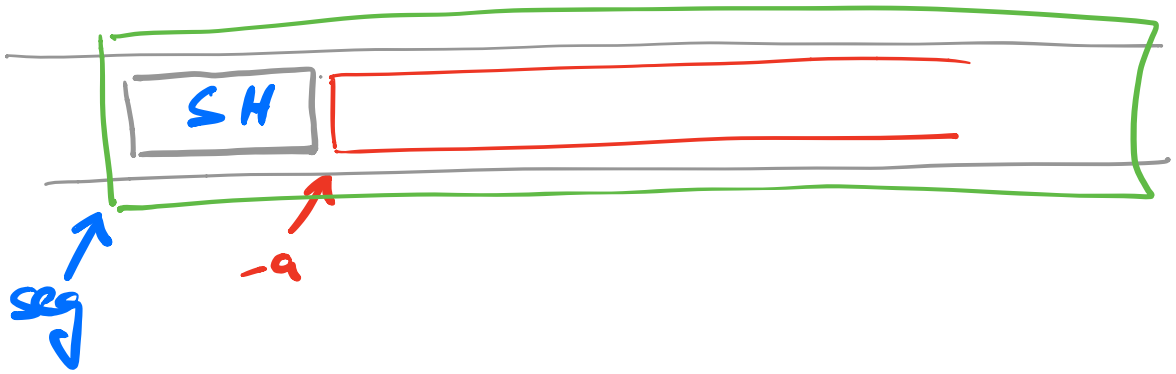


SH * seg;

How do I get ptr ?!

void * ptr = (void*) ((char *) seg + sizeof(SH));

Free(void * _a)



SH * seg = (SH*) ((char *) _a - sizeof(SH));

Add seg to the freelist.

Walk through the implementation of MS2

Prereqs: You have a working FreeList class!

Recall: FL is just a doubly-linked list of SH's.

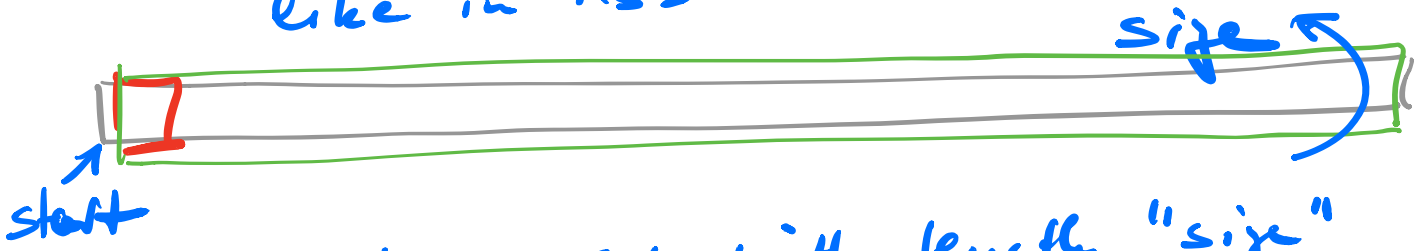
MAKE SURE THAT YOUR FL WORKS!

FList:

Constr.
Add
Remove

My Allocator: constr.

1. start = std::malloc(----)
   like in MS1



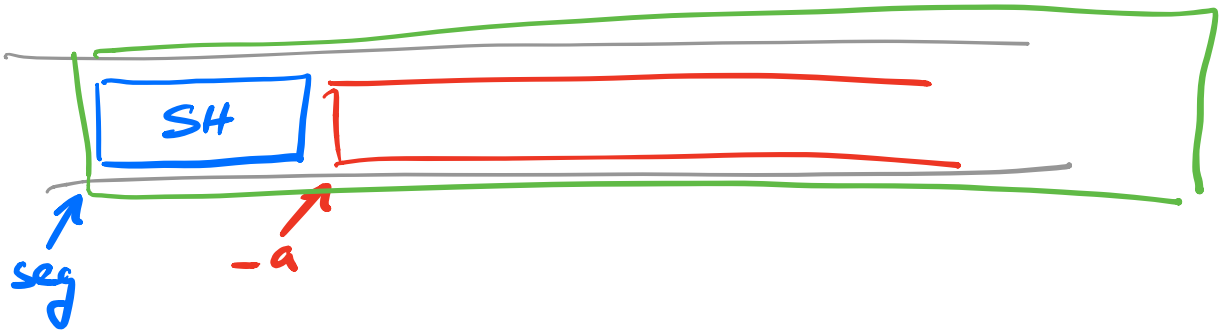
2. create a SH with length "size" using "placement new":

```
SH * seg1 = new (start) SH(size);
```

3. make sure that we have free_list constructed.
4. free_list -> Add(seg1)

~MyAllocator() same as for ms1

Free(void *_a):




```
SH * seg = (SH *)((char *)-a - sizeof(SH));
free_list -> Add(seg);
```

Malloc(size_t _length):

1.) round up to next multiple of block size.

size_t len = ⌈ _length + sizeof(SH) ⌉ round up to next multiple of block size.

3kB — 2.5kB — 1kB

2.) find a long-enough free segment.

```
SH* seg = free_list->head();
while( seg != null && seg->length < len)
    seg = seg->next;
```

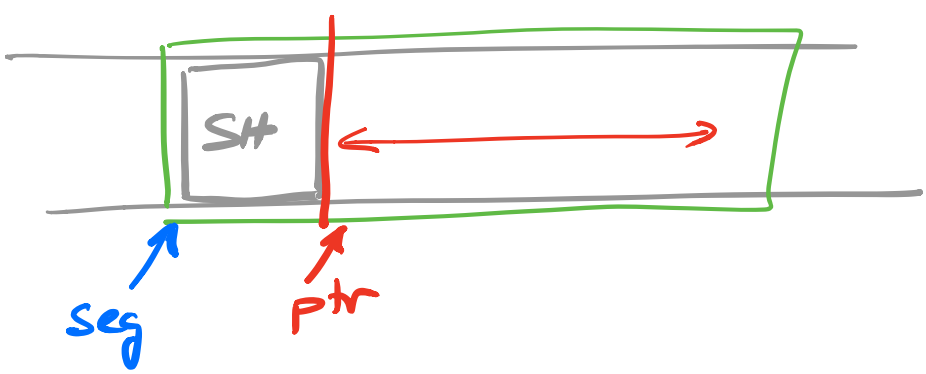
3.) if (seg == null) → return null;

4.) remove seg from free_list

5.) if (seg->length > len) {

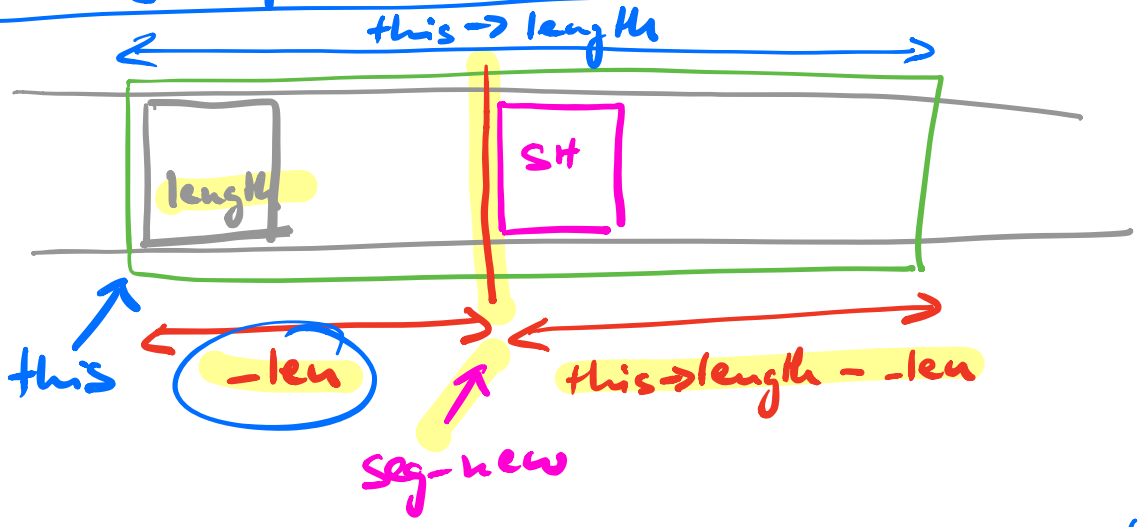
```
    SH* seg2 = seg->Split(len);
    free_list->Add(seg2);
}
```

6.)




```
void* ptr = (void*)((char*)seg + sizeof(SH));
return ptr;
```

SH :: split (size_t _len) → SH*




```
SH * seg_new = new ((void*)((char*)this+_len)) SH(this->length - _len);
this->length = _len;
return seg_new;
```

Diagnostics

- lot of pointer arithmetic
- casting