mm_malloc

var: *heap_start
*heap_end
* heap_current_break
size

size = 0 ?

↓ No

Iterate through the metadata blocks

```
let temp_start = heap_start
    temp_end = heap_current_break
while (temp_start <= temp_end)
    current metadata = temp_start
    size = metadata -> size
    if currentmeta -> status = 'f' and
    size >= target_size
        -> found
    else (not found)
        temp_start += (metadata size
        + size)
-> out of while loop (not found fit size block)
```

↓

Found a size fit block ?

yes

Change the metadata status to 'o'

↓

save a pointer pointing to the address (target) after metadata (9 bytes after)

↓

do split (*old metadata pointer, *target, size tar)

1. check if need to do split
- remain = old meta -> size − sizetar
- if remain <= size_of_metadata (9 bytes)
  -> return
- else (can split)

*new metadata = target + sizetar
- new metadata -> size = remain − 9 (use var)
  -> status = 'f' (use var)

↓

return target.

No

newmeta = mm_sbrk (size + metadata size (9))

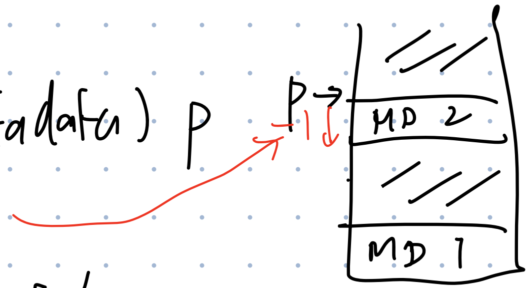
↓

new metadata -> size = size
-> status = 'o'

↓

return newmetadata+1

```
mm_free (void *p)
        current metadata = (*Metadata) p
         currentmetadata -= 1
    current metadata -> status = 'f'
      return
```

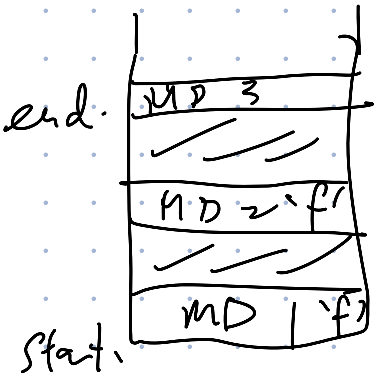


```
mm_combine_nearby_free( )
     2 pointer method

        *start = heap_start
        *end   = heap_start
   while (end and start < heap_current break)
        if start -> status == 'f'
              memorystart = start + 1
              end = start
              memory size = 0
              while (end -> status == 'f')
                  memorysize += end -> size
                  end = go to next metadata

              start -> size = memorysize.

              start = end
              continue
        else start = gotonextmetadata
```



- void * = char* (1 byte for each +1)
- typecasting (void*) name