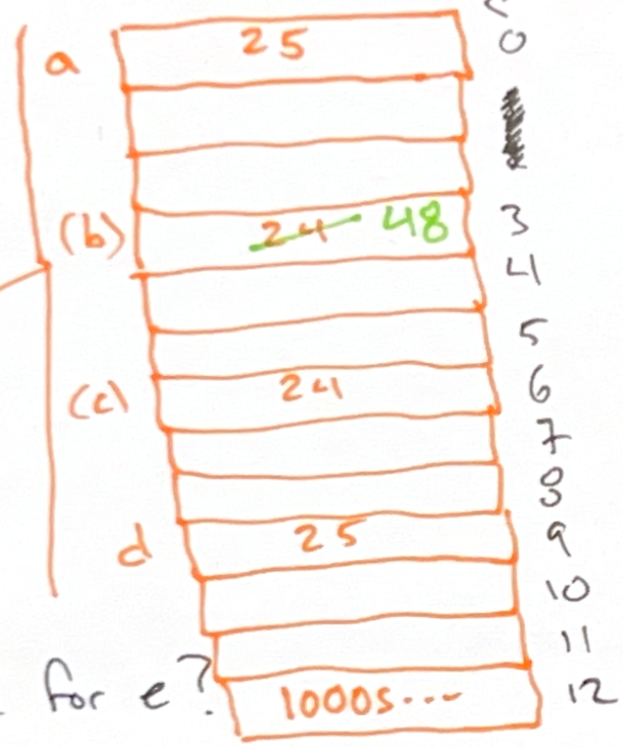
What about adjacent free blocks?

```
a = malloc(16)
b = malloc(16)
c = malloc(16)
d = malloc(16)
free(b)
free(c)
e = malloc(32)
```



start

header	nexti	nexth
25	3	24
24	6	24
24	9	25
25	12	1000...

Can we re-use b/c space for e?

Wednesday malloc would allocate at end of heap!

We need to coalesce (merge, consolidate) adjacent free blocks.

- every time you malloc (in class)

Ideas?

- on free, check next block too PA

```
size_t block_size_of(size_t user_request) {




}

uint64_t* find_block(uint64_t* start, size_t size) {
    int i = 0;
    while(i < HEAP_WORDS) {
        uint64_t header = start[i];
        if(header %2 ==0 && nexth %2 ==0) {
merging ->  start[i] = header + nexth;
            continue;
        } if(header %2 ==0 && header >= size)
            return &start[i];
        i += header/8;
    }

    return NULL;
}

void split_block(uint64_t* block_start, size_t size) {




}
```

insert these

```
(header %2)==0 // is the block free?
header >= size // what is its size?
int nexti = i + header/8 // is the next block free?
uint64_t nexth = start[nexti];
(nexth %2) ==0
```

"Best fit" vs. "First fit" → use the earliest big enough block size

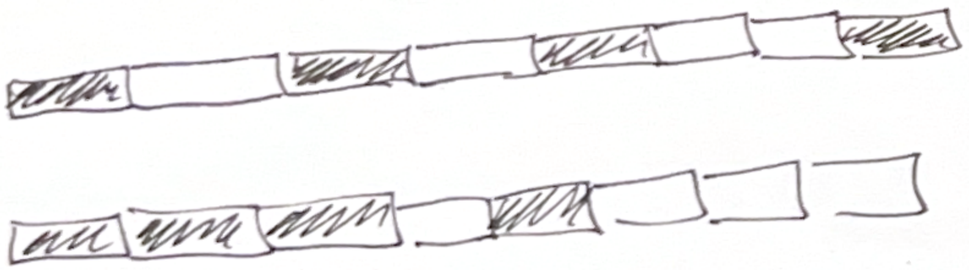
↳ look for closest big enough block size

"smallest big enough block"

Utilization vs. Speed ⟶ how many blocks are inspected per allocation

↳ what % of the heap can be used

or how much extra free space is between start and last allocated block



better utilization