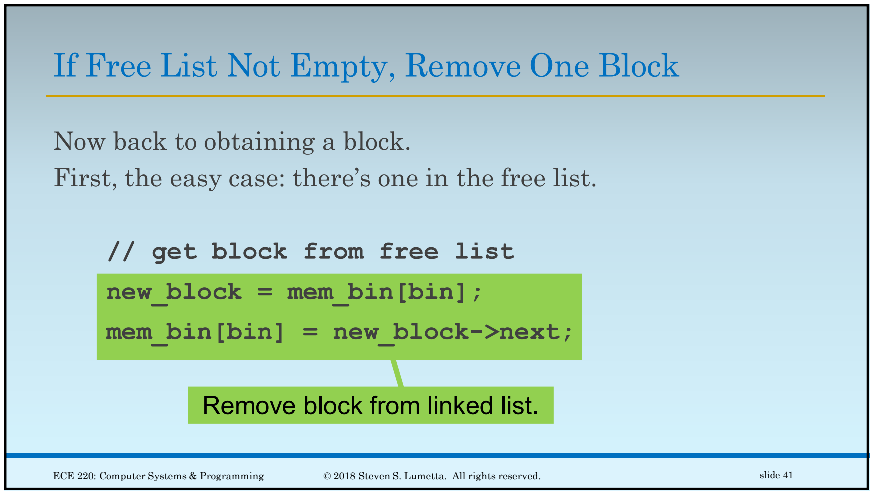


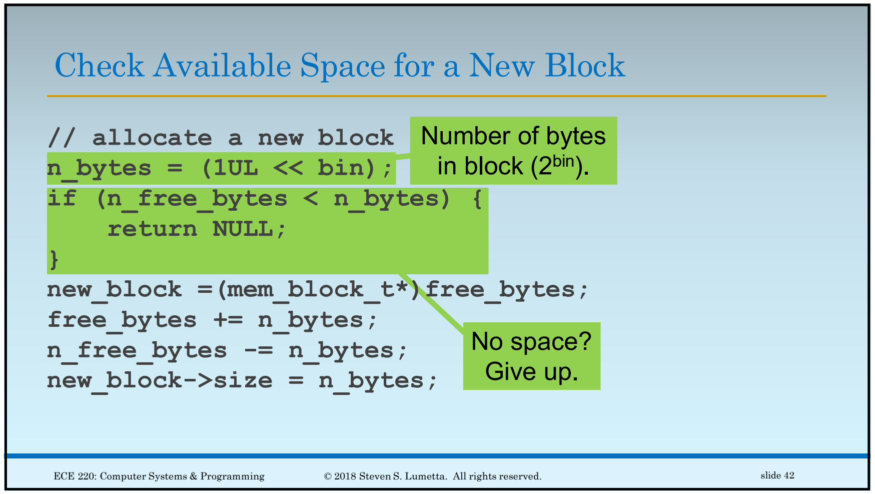
If Free List Not Empty, Remove One Block

Now back to obtaining a block.

First, the easy case: there's one in the free list.

```
// get block from free list
new_block = mem_bin[bin];
mem_bin[bin] = new_block->next;
```

Remove block from linked list.

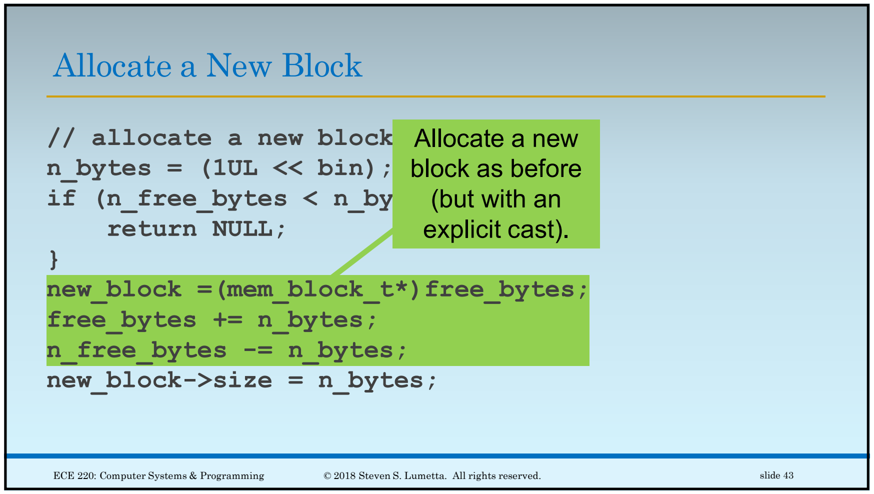


Check Available Space for a New Block

```
// allocate a new block
n_bytes = (1UL << bin);
if (n_free_bytes < n_bytes) {
    return NULL;
}
new_block =(mem_block_t*)free_bytes;
free_bytes += n_bytes;
n_free_bytes -= n_bytes;
new_block->size = n_bytes;
```

Number of bytes in block (2^{bin}).

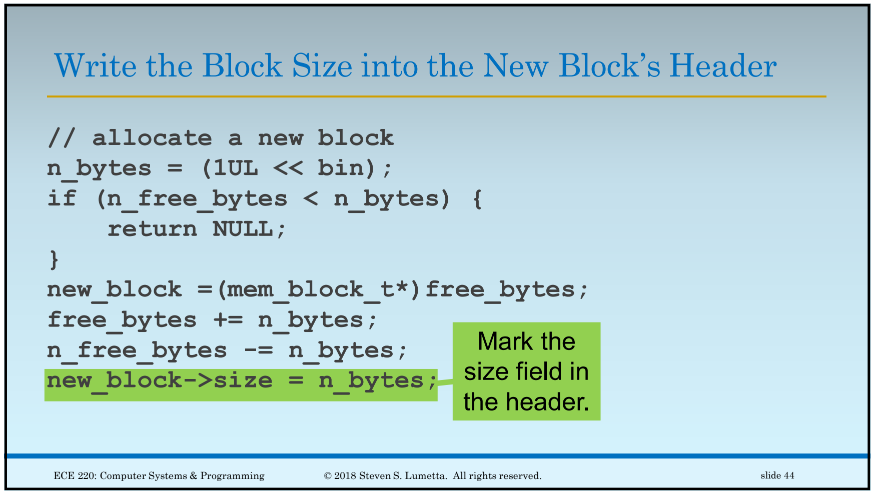
No space? Give up.



Allocate a New Block

```
// allocate a new block
n_bytes = (1UL << bin);
if (n_free_bytes < n_by
    return NULL;
}
new_block =(mem_block_t*)free_bytes;
free_bytes += n_bytes;
n_free_bytes -= n_bytes;
new_block->size = n_bytes;
```

Allocate a new block as before (but with an explicit cast).



Write the Block Size into the New Block's Header

```
// allocate a new block
n_bytes = (1UL << bin);
if (n_free_bytes < n_bytes) {
    return NULL;
}
new_block =(mem_block_t*)free_bytes;
free_bytes += n_bytes;
n_free_bytes -= n_bytes;
new_block->size = n_bytes;
```

Mark the size field in the header.