CS 326 Generalized Block List

```
struct block_hdr *bp;
```

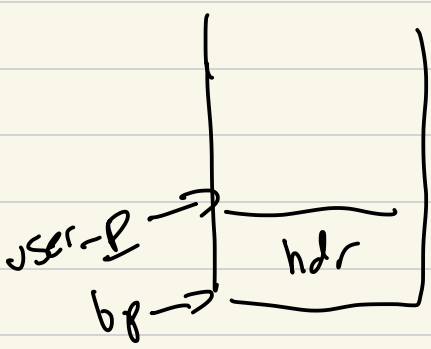
block pointer

```
void *p;

p = sbrk(4096)

bp = (struct block_hdr *) p;

bp -> size = nbytes;
```

```
int *ip;
ip = (int *) malloc(4);
```

sizeof(int)

```
void *ptr

ptr = malloc(32);
```




```
ip = ip + 1;
```

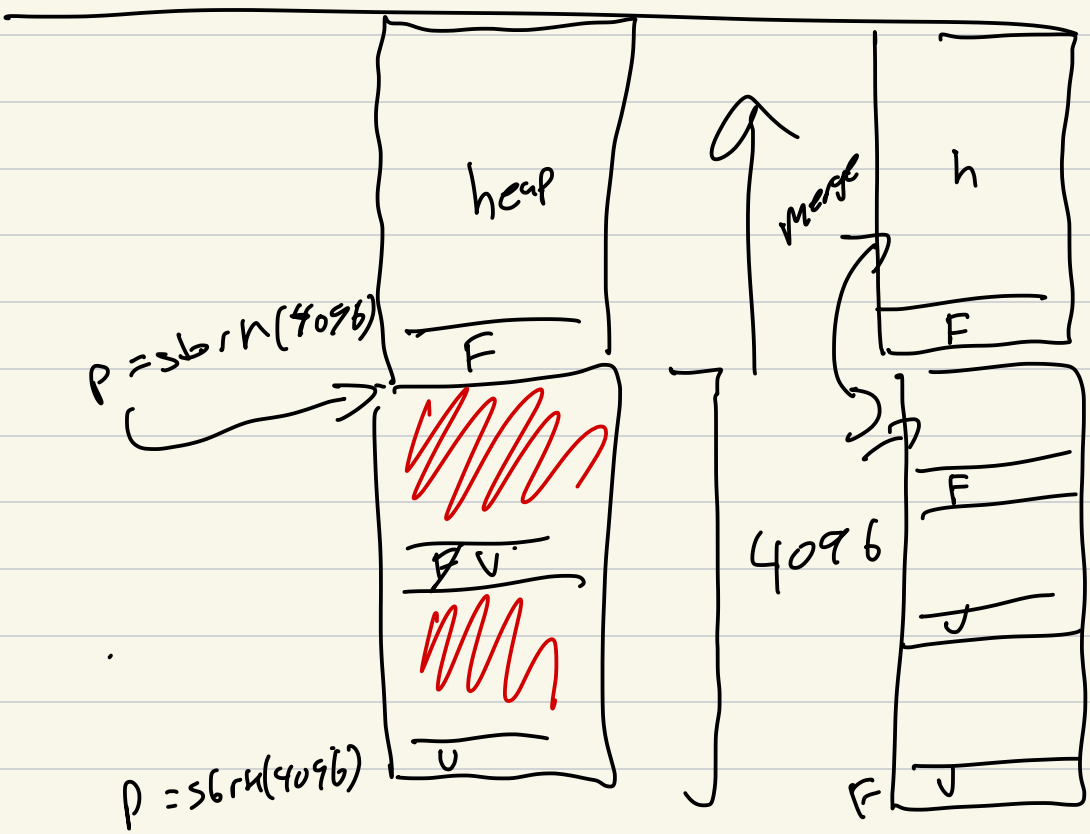
void *user_p;

user_p = ((void*) bp) + sizeof(struct block_hdr);

32

= bp + sizeof(struct block_hdr);

32 x 32



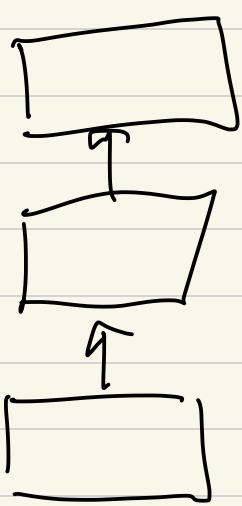
Generalized Block List Management

malloc(3)

malloc_name()

initialization

allocate



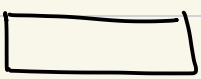
allocate

→ look for free block

iterate over block_list

either

1) find a free block



2) do not find a free block

→ bp → free block

→ NULL (0)

if (bp == 0) {

// allocate heap space

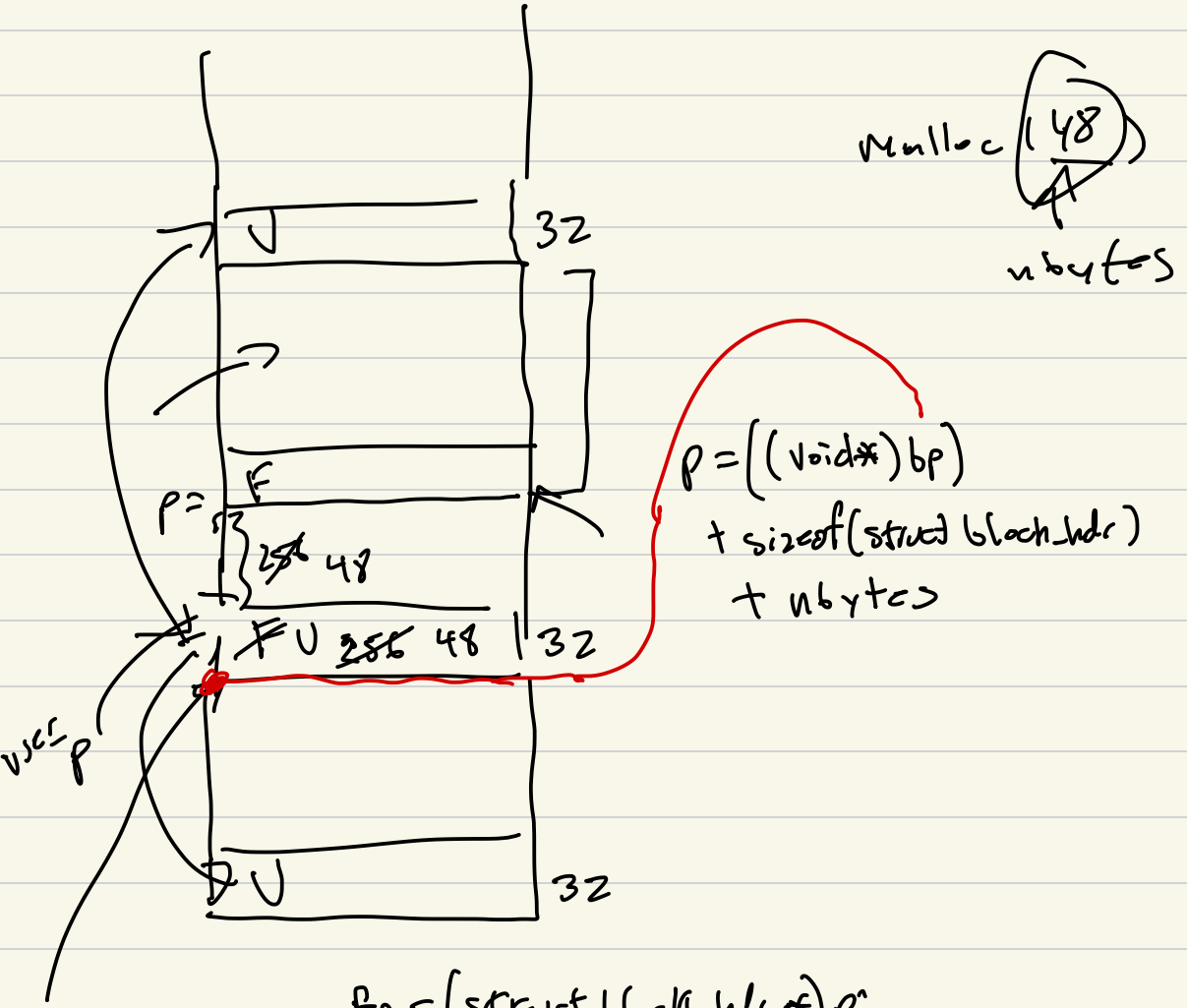
fp = | 32 | 4064 |

bp = ____

}

0

bp → block

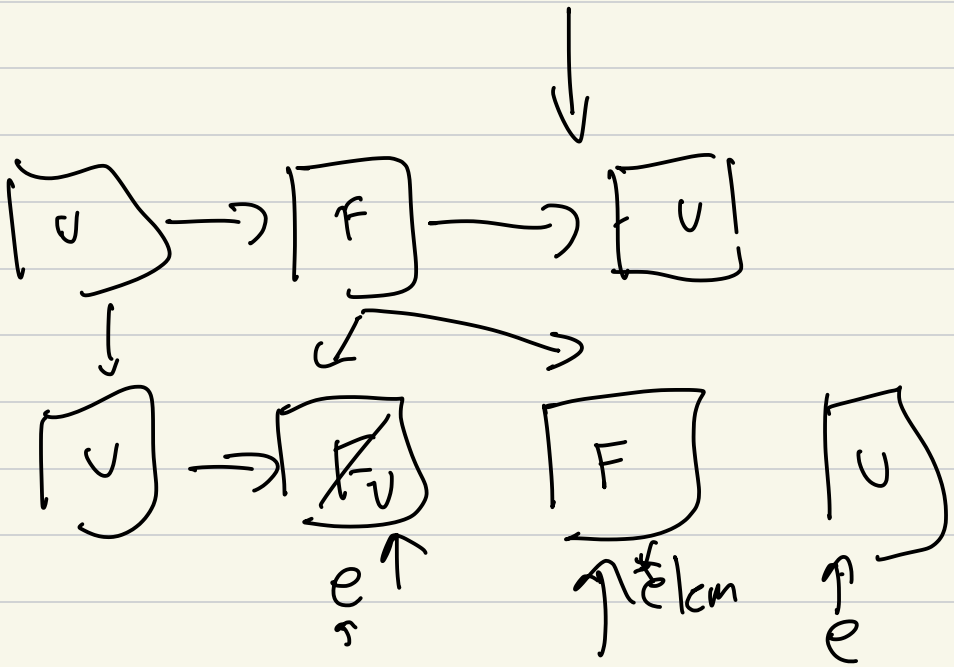


fp = (struct block_hdr *) p

fp -> used = 0

fp -> size = org_size
- (nbytes + sizeof(struct block_hdr));

fp -> name[0] = '\0';



e = list_next(e)

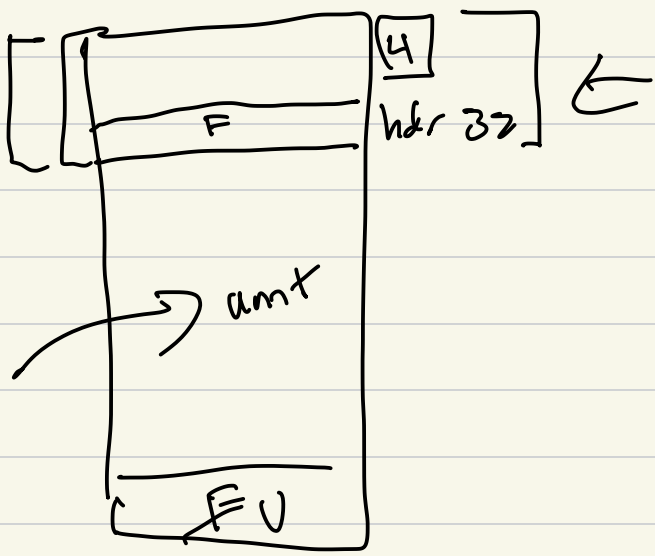
list_insert(e, *elem)

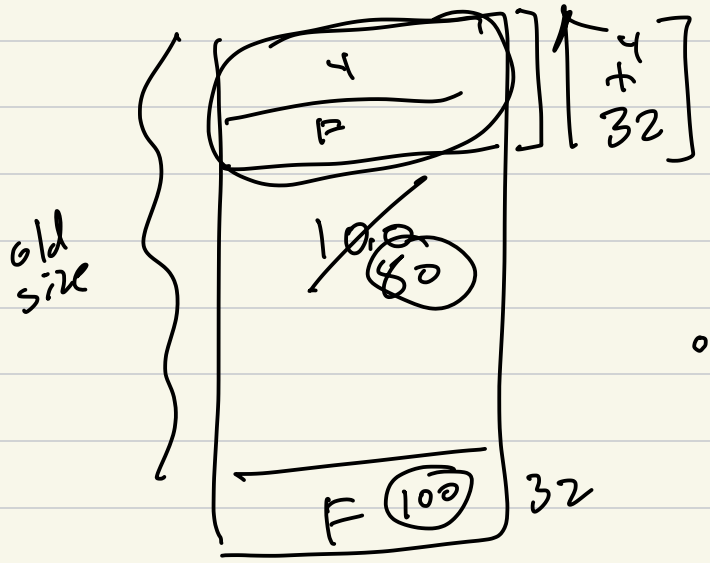
Splitting Free blocks

1) amount requested equals size of free block

2) need to split free block into used block and free block

3) 32 + 4 = 36
hdr + min





malloc(80)

old_size − (80 + 32 + 4)

100 − (116)