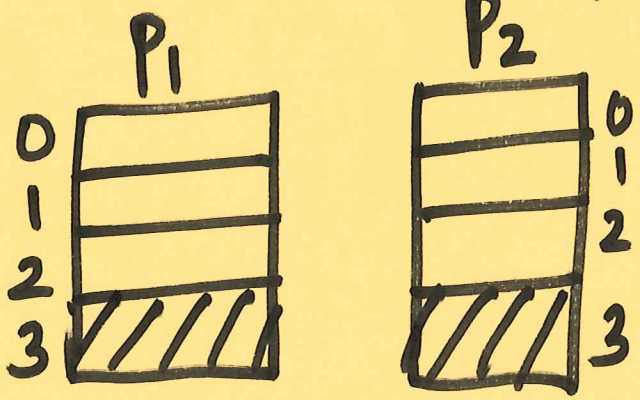
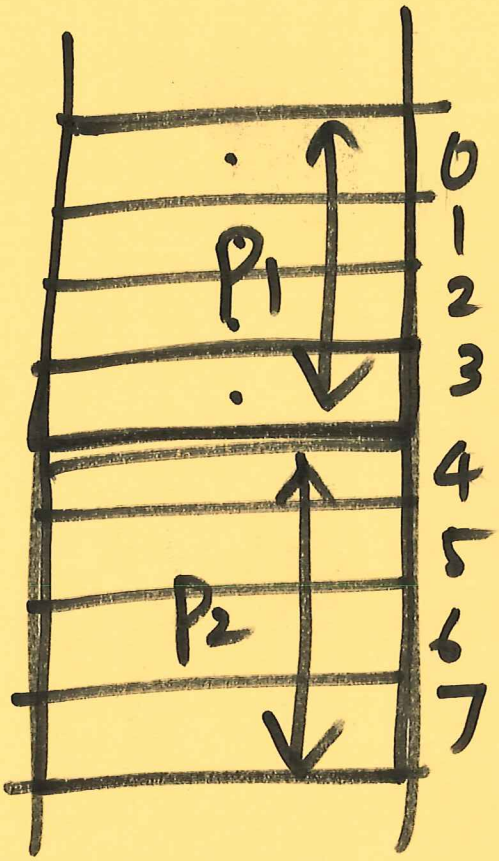
Lecture - 31

Review

Virtual memory





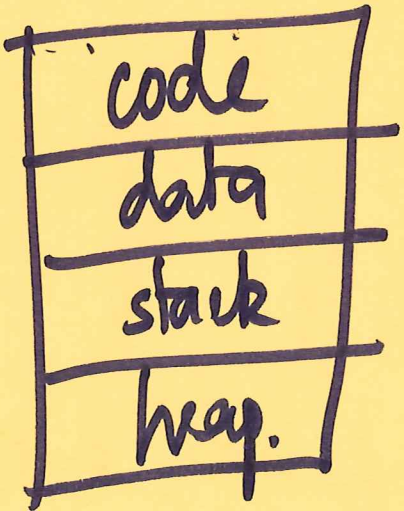
P1

VA	PA
0	0
1	1
2	2
3	3

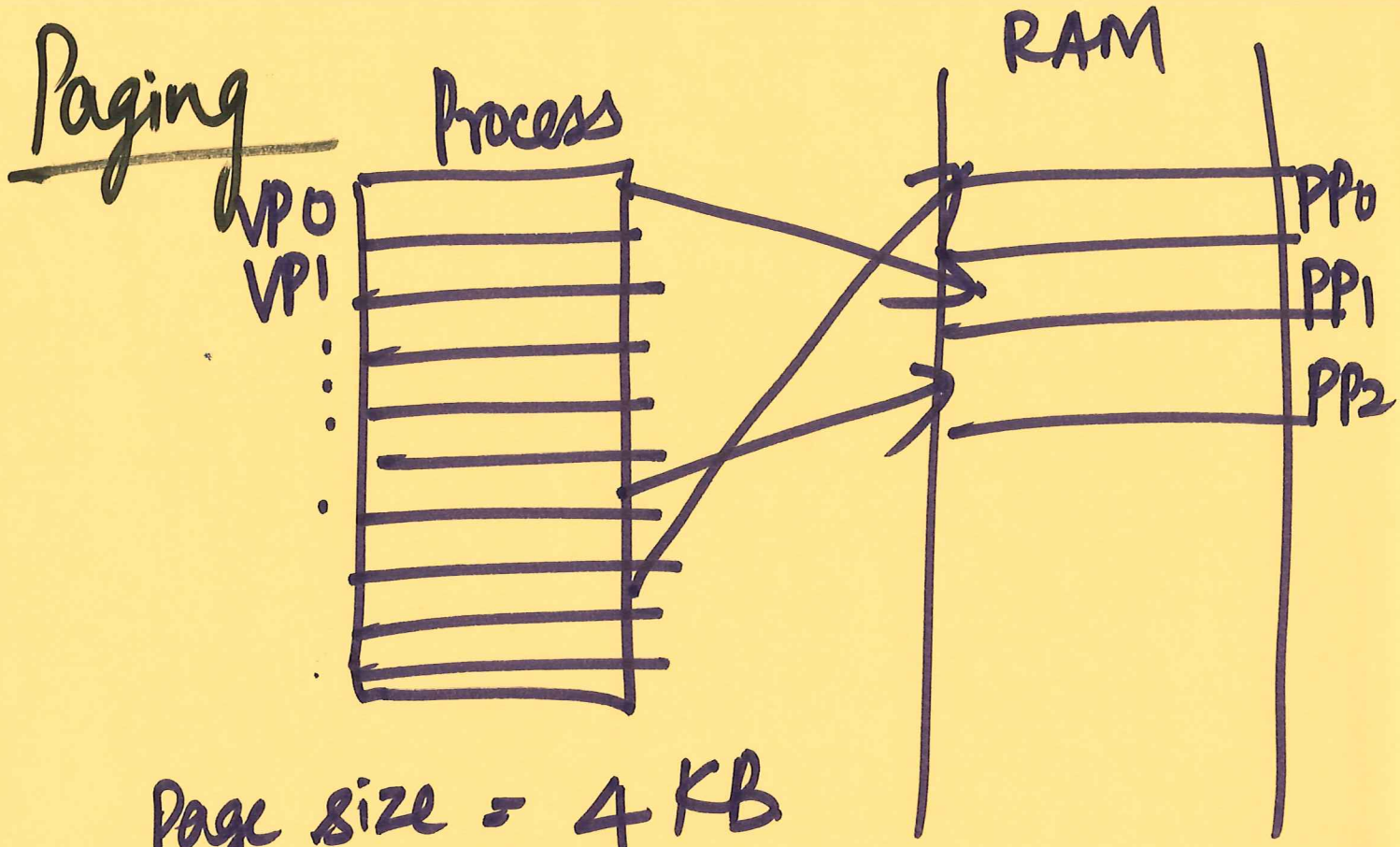
P2

VA	PA
0	4
1	5
2	6
3	7

Segmentation



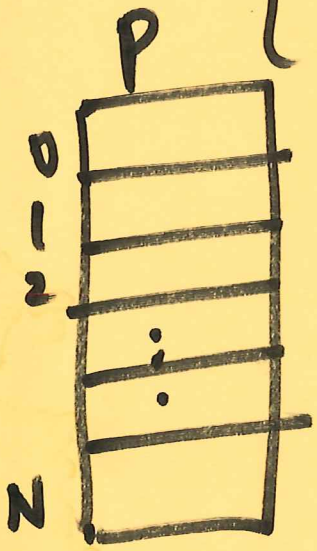
Paging



Page size = 4 KB

Address Space

$\{0, 1, 2, \ldots\ldots N\}$

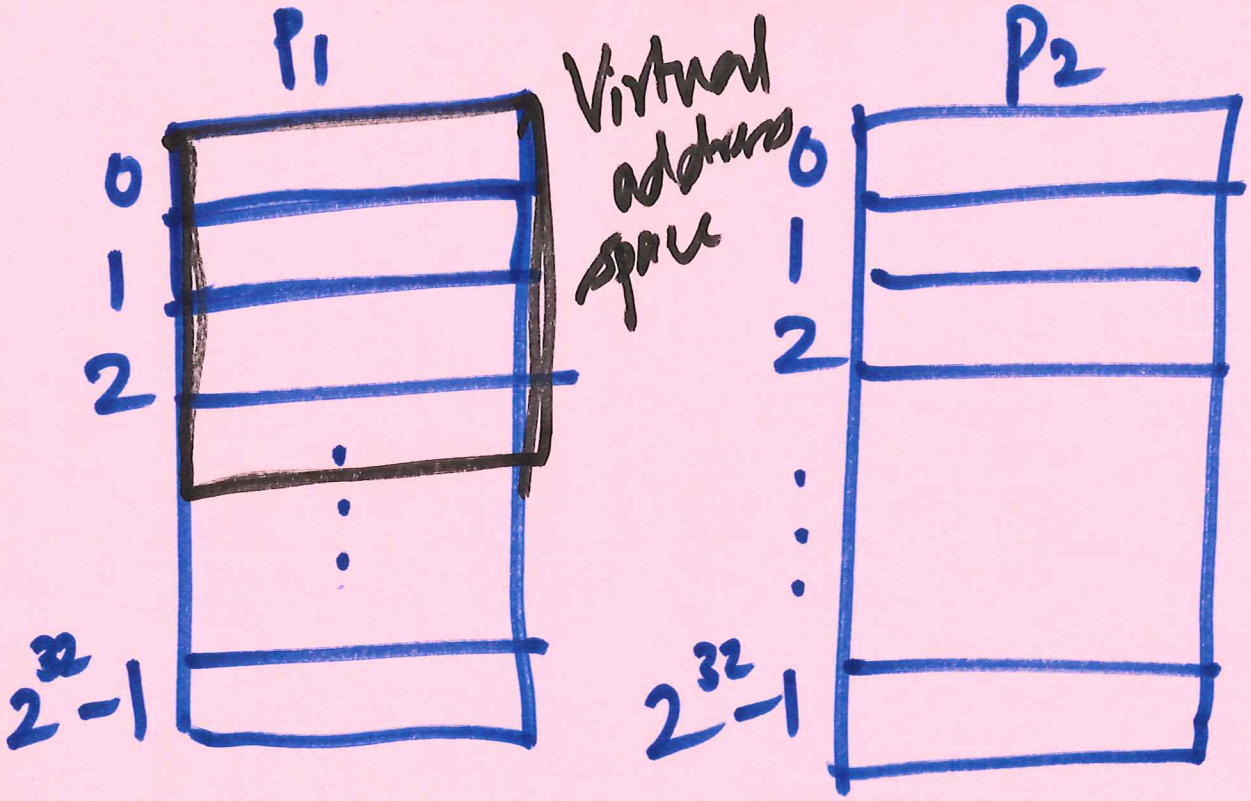


32-bit address space.

each address = 32-bit in size

2^{32} address.

$\{0, 1, 2, \ldots - 2^{32} - 1\}$



$$2^{32} \text{ bytes} = 2 \times 2^{30} \text{ bytes}$$

$$= 4 \text{ GB}$$

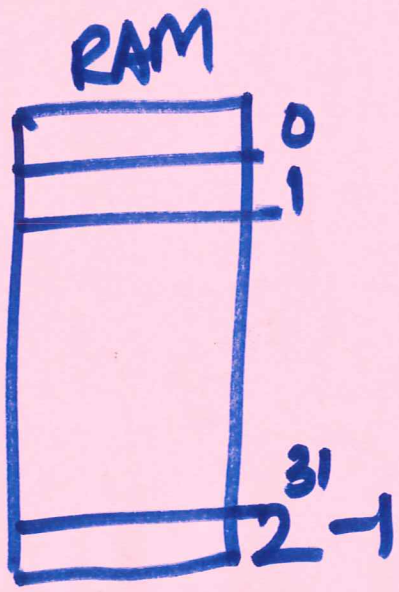
Physical Addr space

Main Memory size

eg. 2 GB

{0, 1, 2 . . . - M}

$2^{31}-1$



2^{10} = KB

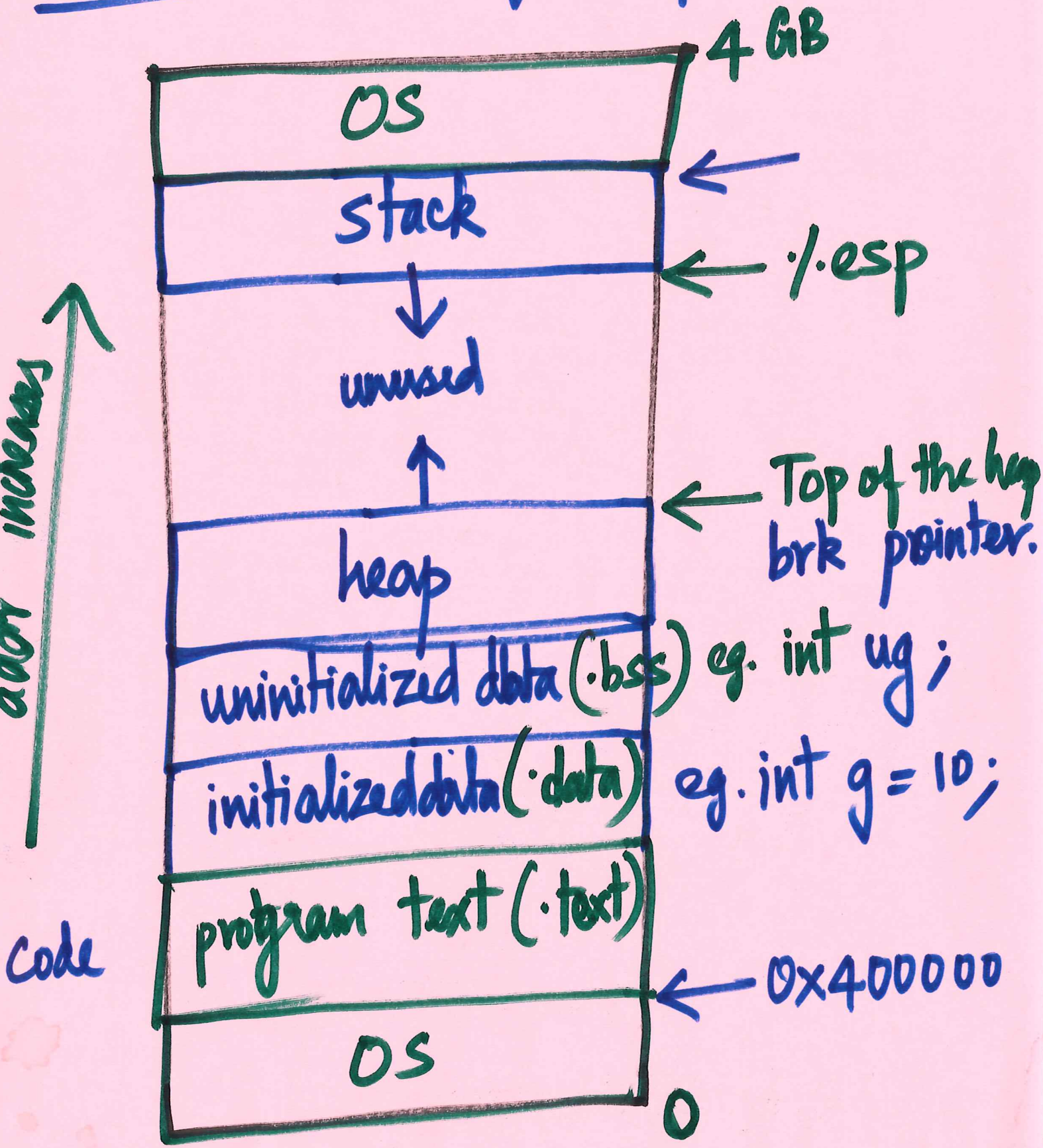
2^{20} = MB

2^{30} = GB

2GB $= 2 \times 2^{30}$ bytes

$= 2^{31}$ bytes.

Address space of a process



Functions to access the heap

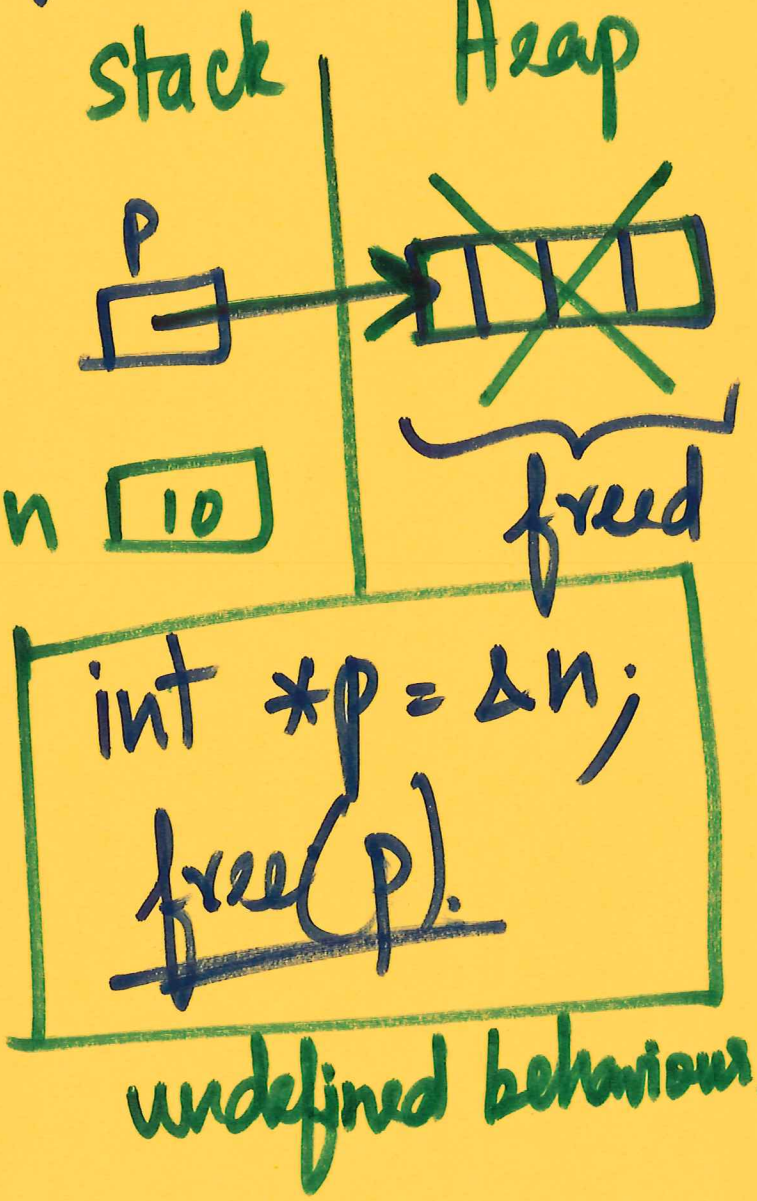
1. malloc
2. free.

```
p = malloc(4);
```

```
void free(void *ptr);
```

```
free(p);
free(p);
```

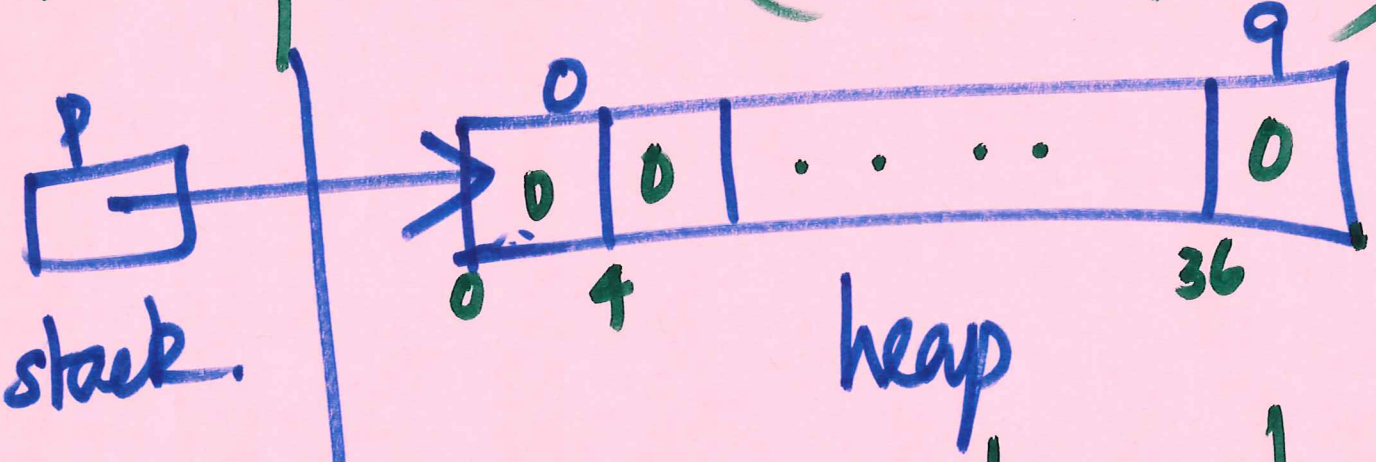
"nasal demons"



3. calloc

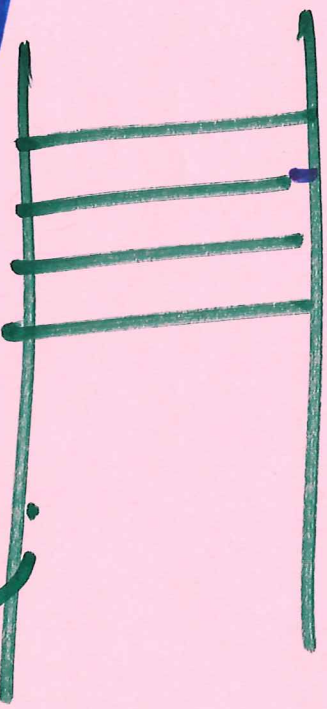
```
void *calloc(size_t nmemb,
             size_t size);
```

```
int *p = calloc(10, sizeof(int));
```



4. realloc

```
void *realloc(void *ptr,
              size_t size);
```

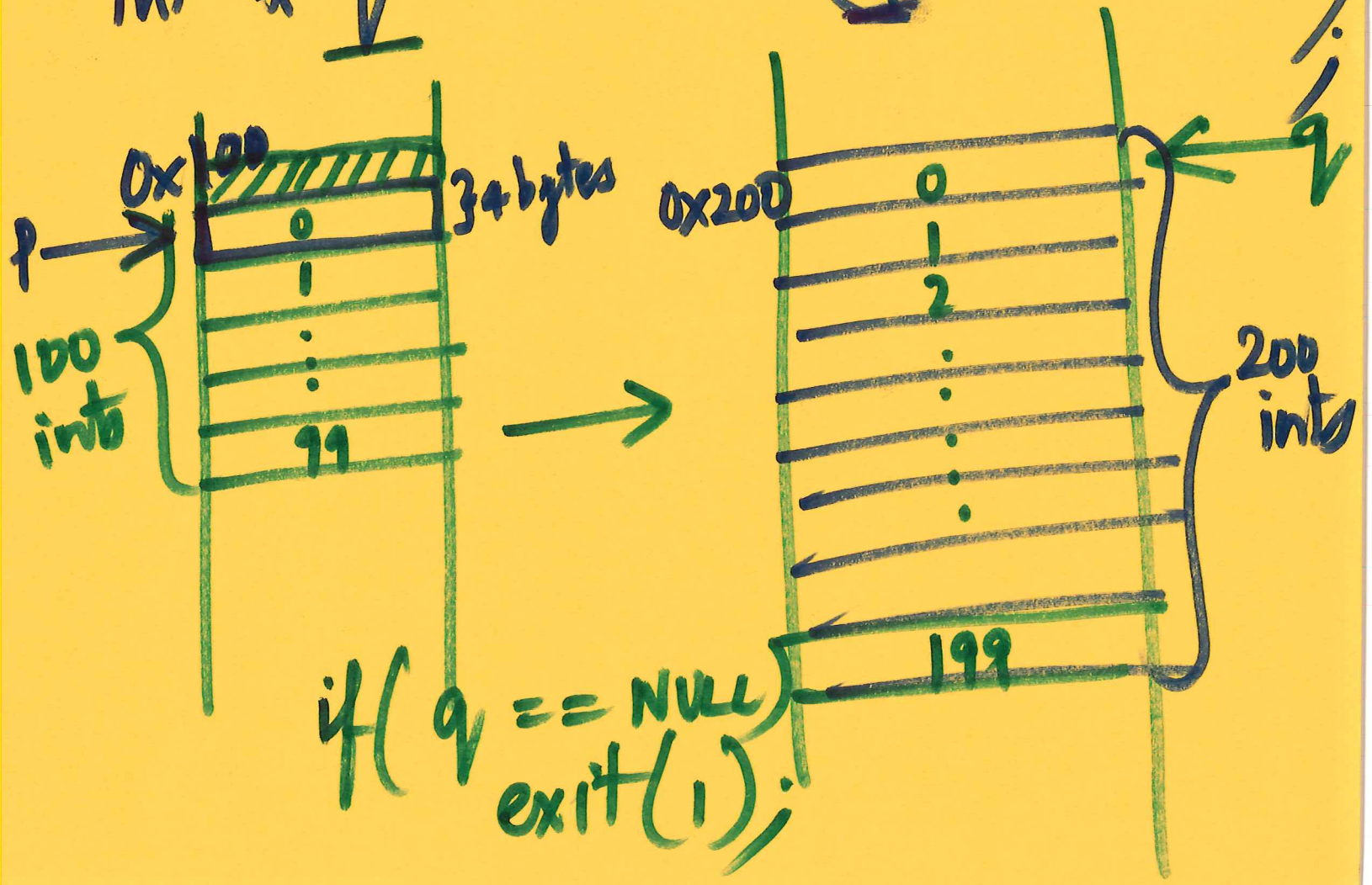


```
int *p = malloc(100 * sizeof(int));
```

⋮

if we want to increase/decrease memory in the heap.

```
int *q = realloc(p, 200*sizeof(int));
```



realloc

if (p == NULL)

⇒ malloc

if size == 0, ptr != NULL } ⇒ free

#include <unistd.h>

int brk(void * addr);

sbrk