Virtual Memory

- OS must manage physical memory
  - One single physical resource
  - All programs want to use same addresses
    - Stack 0xff...
    - Code 0x80...
  - Portability

- What if physical memory is smaller than expected?
  - Make memory seem larger than it is (swapping)

- Protection
  - Isolates your data
  - Protects other's data

- Sharing between processes
1) how VM works
2) how VM is actually implemented
3) speeding up VM (caching)
4) memory management user-level (malloc)
5) code + memory (load + link)

---

Physical addressing

CPU

1d 0x100

memory

0x1000
Virtual addressing

MMU - Memory Management Unit

address space: Set of addresses (usually linear)

Virtual address space $\mathbb{Z}_0^{N-1}$
Physical address space $\mathbb{Z}_0^{M-1}$

Program's address space → memory the program thinks it can access
Virtual address space
```c
int a = 12;
&a -> What's possible here?
```

Diagram:

```
P1
Virtual address space

Stack

heap

code

D

P2 Virtual address space

Stack

head

code

Map VA to PA

Translation

L7 Conversion from virt. to phys
```
Translation table

P1: 0x80000000 → 0x30000000 1,000,000
P2: 0x80000000 → 0x10000000 1,500,000

Segmentation →

Segmentation leads to fragmentation solution? →