Common memory leak from Lab 2.

```c
struct vspace new;

// initialize & set up new vspace;
1 vspace init (&p→vspace);
2 vspace copy (&p→vspace, &new);
3 vspace install (p);
4 vspace free (&new);
```

What happens to your current page table & data?

This makes a deep copy of the new address vspace, is that what we want?

```c
struct vregion {
    enum vr_direction dir; // direction of growth
    uint64_t va_base; // base of the region
    uint64_t size; // size of region in bytes
    struct vpi page *pages; // pointer to array of page_infos
};

struct vspace {
    struct vregion regions[NREGIONS]; // the regions for a process' virtual space
    pm4e_t* pgtbl; // process' page table
};
```

vspace just stores pointers to things...
swap from the old vspace to new one

```c
exec() {
    struct vspace new;
    // initialize & set up new vspace:
    struct vspace old = p->vspace; // saves old vspace struct, has references to old page table & bookkeeping data
    p->vspace = new;
    vspaceinstall(p); // switch to new
    vspacefree(&old); // safe to free old vspace.
}
```
Solid State Drive

Disk: Sectors (512 bytes)

SSD: Page (2k-4k) read/write unit

Erasure Block (1MB-8MB), erase unit

Erase: Erase all pages in the block (set all bits to 1)

Write: Write Os

Initially (Invalid)

Erase (block) (Erased)

Write (page) (page written = Valid, other pages = Erased)

Read (page)
Performance

→ read (page level): good (~10us) for sequential and random read
→ write (page level): write to an erased page (~100us)
   / program
→ erase (block level): takes a few ms

On erase, what happens to existing data in the block?

Flash Translation Layer

→ logical to physical block #
  ↓ what we give to client (kernel)

Block 1

1
aa
alen

2
bb
llen

3
cc
elen

→ program/erase so many times before the block becomes unusable

→ wear leveling, spread writes across blocks & pages

Block 2

1
aa
alen

2
bb
elen

3
ee
elen

→ can only program/erase

On erase, what happens to existing data in the block?

1
aa
alen

2
bb
llen

3
cc
elen

⇒

1
aa
alen

2
bb
elen

3
ee
elen

to overwrite page 3 we need to erase block 1

copy existing data to erased pages elsewhere, erase block and
then write to block 1 pages

not the most efficient approach, had to move data around multiple times

www.pagegweneea
Performance Metrics

→ I/O operations per second (IOPS)
→ I/O request: size (# of bytes to read/write)
  access patterns matter! (workload)
  → simple average IOPS
  \[
  \text{IOPS} = \frac{1}{\text{average request latency}}
  \]
  for disk: \[\frac{1}{14\text{ ms}} \approx 70 \text{ IOPS}\]

→ actual IOPS
  \[
  \text{IOPS} = \frac{\text{# of I/O requests}}{\text{total time to complete the requests}}
  \]
  for 10 sequential reads on disk: \[\frac{10}{14\text{ ms}} \approx 700 \text{ IOPS}\]
Filesystems

- Abstraction for storage devices
- Names & data (files, directories)
- Manages storage devices & offers richer APIs

Data vs Metadata

File System API and Performance

Device Access