ALLEGHENY COLLEGE

Operating Systems:
Chapter 3
Virtual Memory-Space
CS400

Week 5: 12th Feb

Spring 2020

Oliver BONHAM-CARTER

Limiting Factor

- Limiting factor: Memory is a critical resource
- Memory needed for running applications, carry out the operations.
- Common questions
  - *How can I load all of my software into memory?*
  - *How quickly is my memory being used up?*
  - *How can I make better software to use my memory more efficiently / effectively?*

ALLEGHENY COLLEGE

Memory Usage: Linux

- MacOS Usage
  - Storage: 65GBs taken up by OS on my MacBook Air
  - A tab on my Chrome browser uses about 45MB
- *How much memory on Linux are you using now?*
- `free -h`

```
             total       used       free     shared    buffers     cached
Mem:          3699       2896        802          0        247       1120
-/+ buffers/cache:       1528       2170
Swap:         1905         62       1843
```

ALLEGHENY COLLEGE

Your Own Memory Usage

- Mem: used is your total used memory.
- -/+ buffers/cache: used is your total used memory minus buffers and cache.
- I am using 1528 MB and have 2170 MB free.

```
             total       used       free     shared    buffers     cached
Mem:          3699       2896        802          0        247       1120
-/+ buffers/cache:       1528       2170
Swap:         1905         62       1843
```

ALLEGHENY COLLEGE

Your Own Memory Usage

- Another way to watch a process for its memory usage:
  - `ps u -p 31730 | awk '{sum=sum+$6}; END {print sum/1024}'`
- The 31730 is the process id.

```
PID    COMMAND      %CPU TIME     #TH  #WQ  #PORT MEM    PURG   CMPRS  PGRP  PPID  STATE
31732  ReportCrash  0.0  00:00.67 7    3    69    4132K  0B     0B     31732 1     sleeping
31730  top          4.1  00:00.68 1/1  0    20    2564K  0B     0B     31730 31485 running
31656  Google Chrom 4.7  00:27.28 14   0    126   102M   0B     0B     26943 26943 sleeping
31513  mdworker     0.0  00:00.04 3    0    50    1428K  0B     0B     31513 1     sleeping
```

ALLEGHENY COLLEGE

Python's Memory Usage?

- Run Python in a shell
- Open a new shell
- Run command
- `ps aux | grep python | awk '{sum=sum+$6}; END {print sum/1024 " MB"}'`
- Outputs the number of MBs used by Python program (or the program you checked)

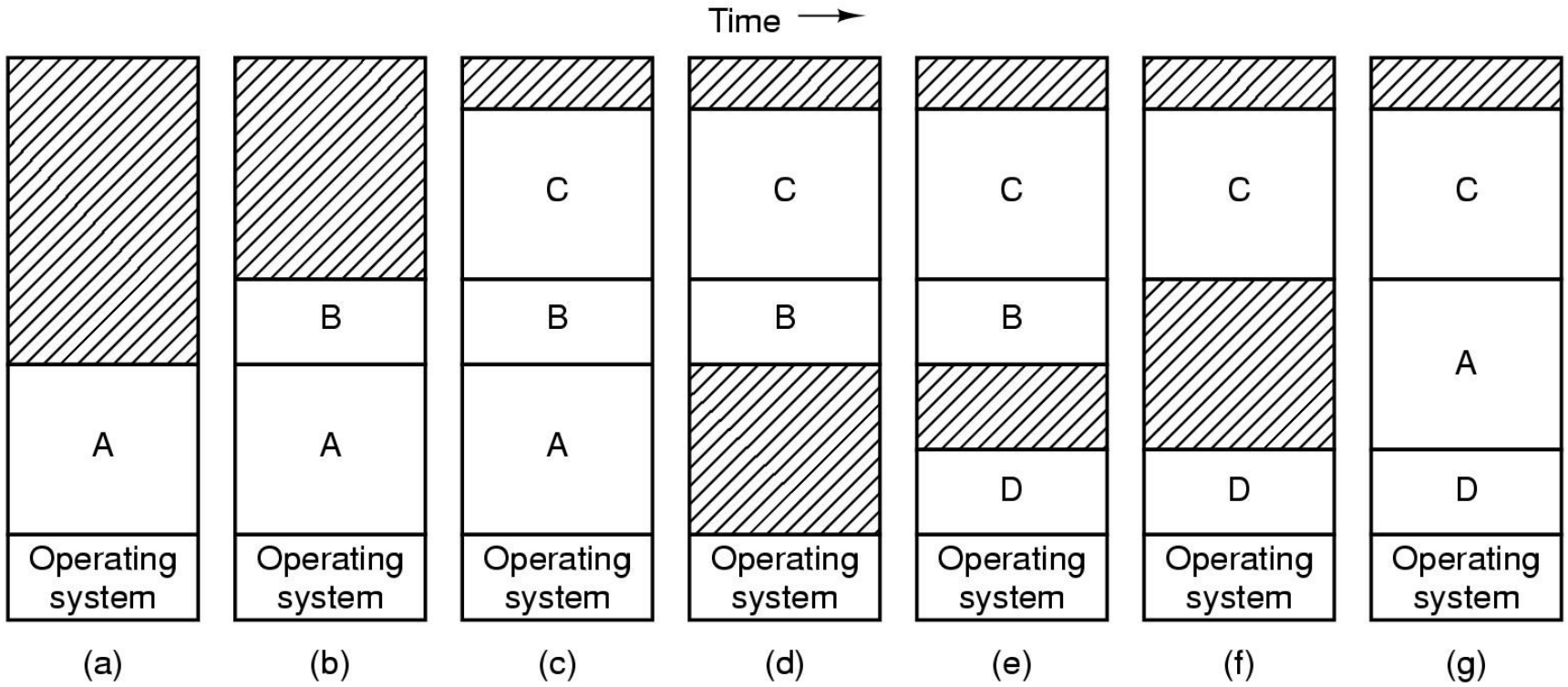
How Can I Use More Memory?

- Why do we care about loading more into our memory banks?

- Swap files
  - Disadvantage: Slow since we are using memory in storage space
- Virtual memory
  - However, parts of the program may actually be running in the main memory

Swapping

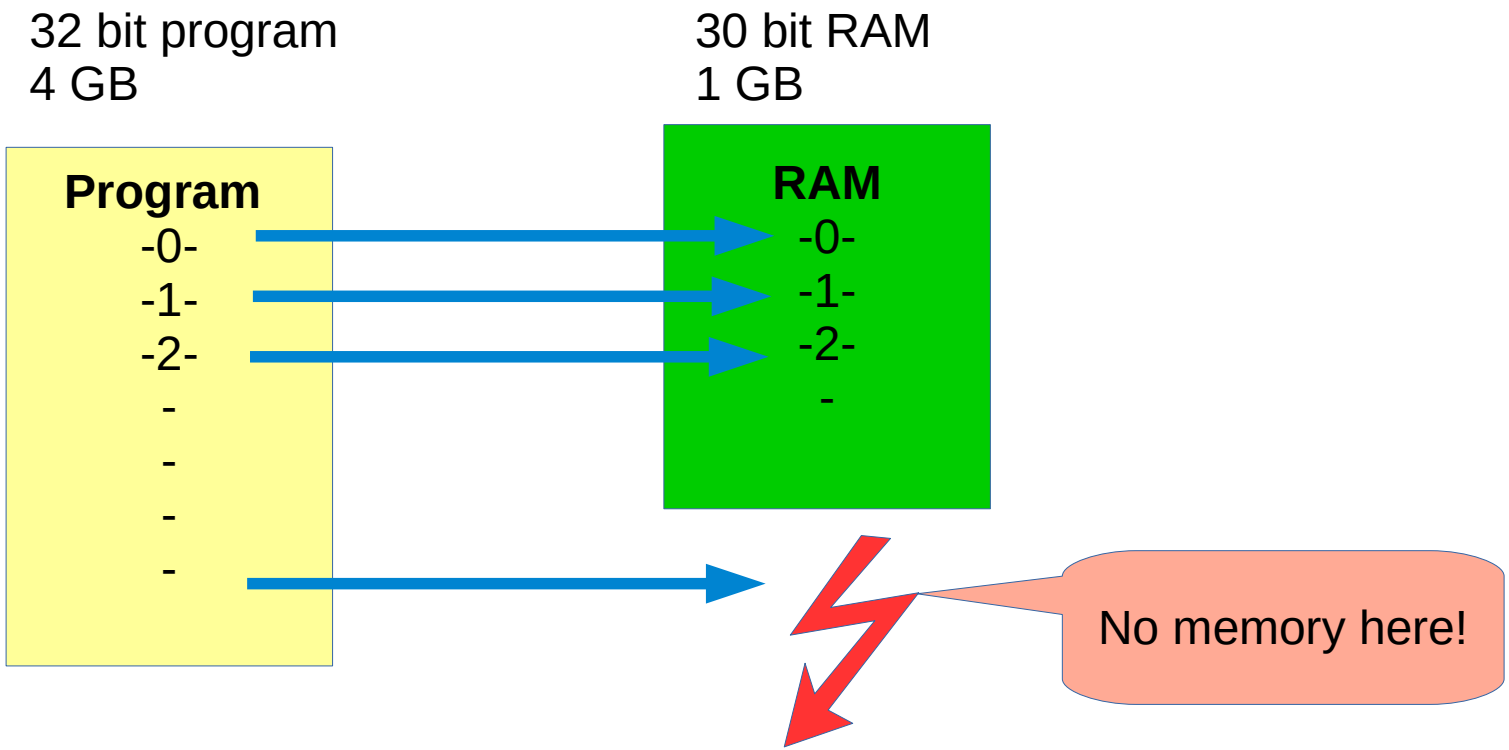
- What does swapping programs look like?
- Programs are placed into openings when other non-active ones are removed
- This is slow going!



ALLEGHENY COLLEGE

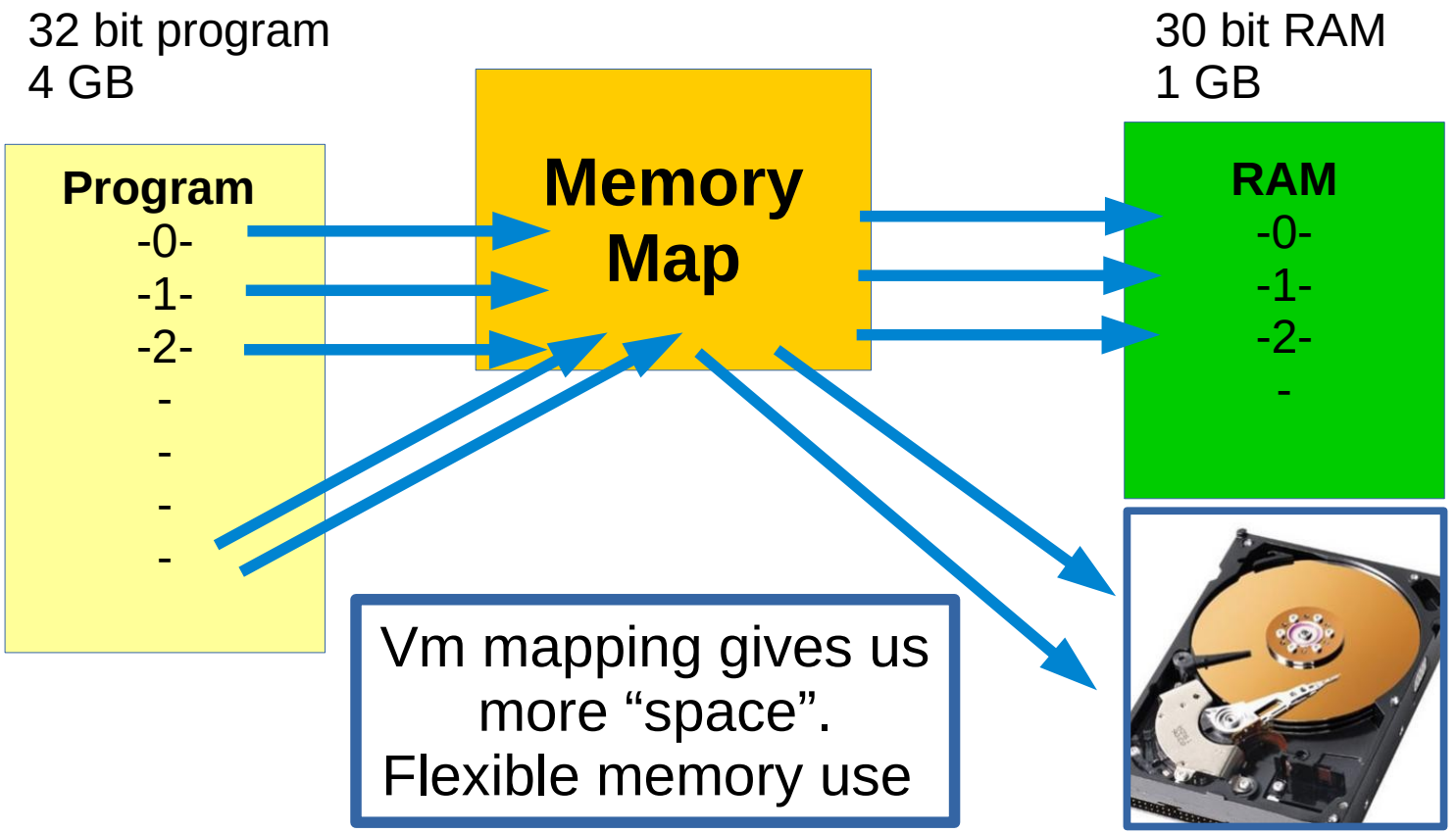
Intro to Virtual Memory

- **No abstraction**: program addresses = RAM addresses
- Crashes possible if we try to map to non-existent RAM



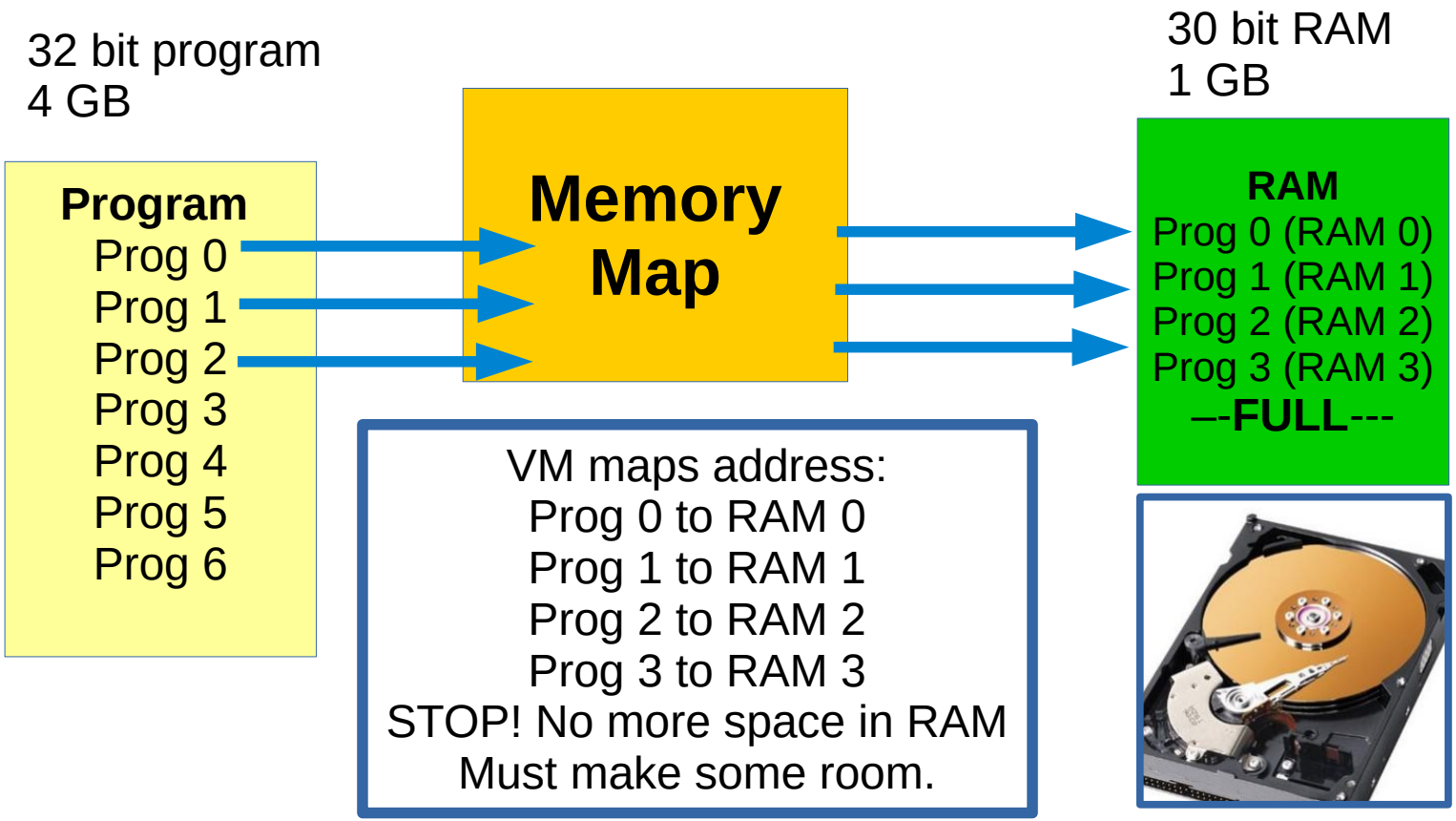
Intro to Virtual Memory

- **Virtual Memory**: program addresses **MAP** to RAM addresses
- Crashes possible if we try to map to non-existent RAM



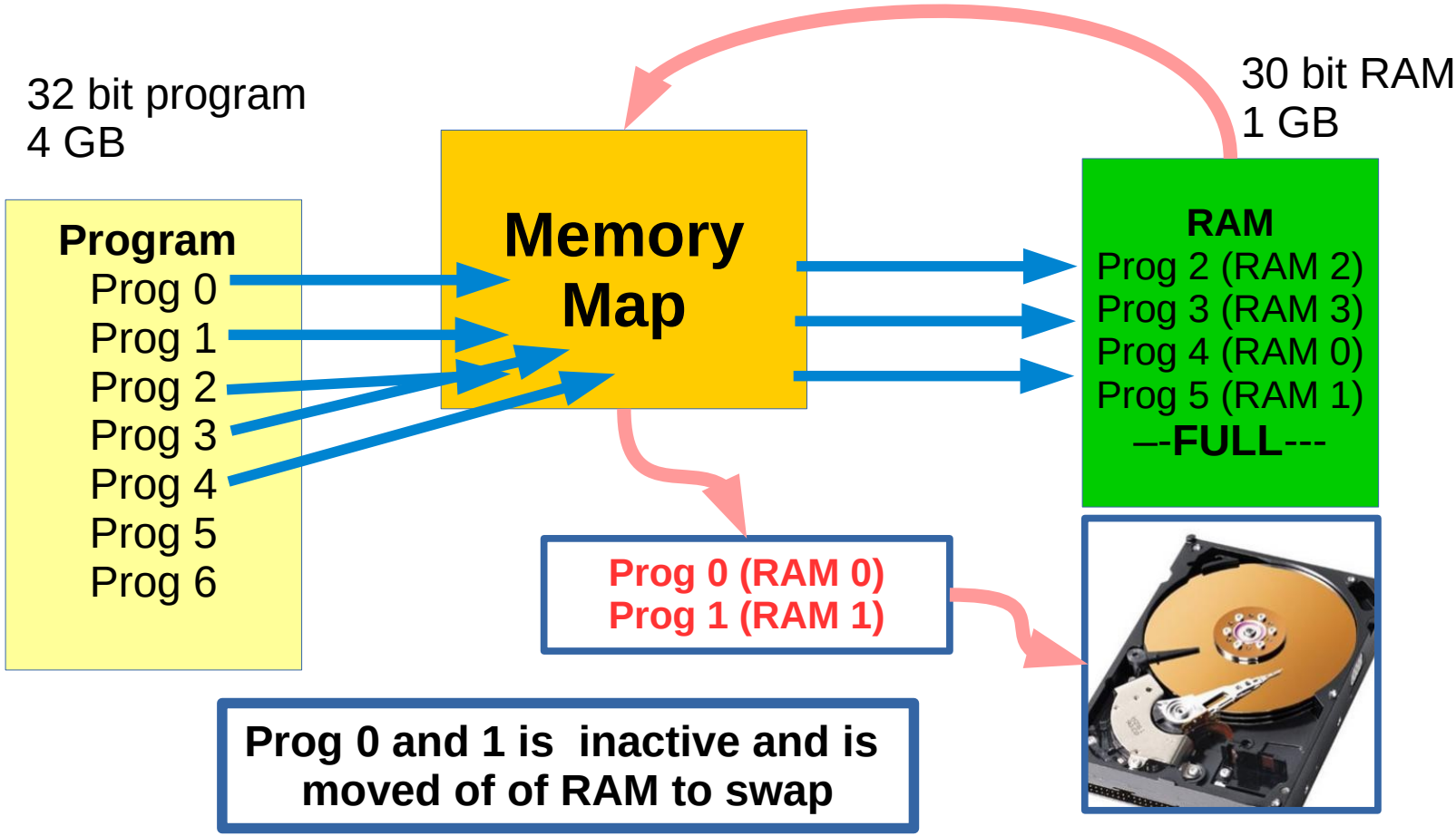
Intro to Virtual Memory

- **Virtual Memory**: program addresses **MAP** to RAM addresses
- Program loads into RAM via memory mapping.



Intro to Virtual Memory

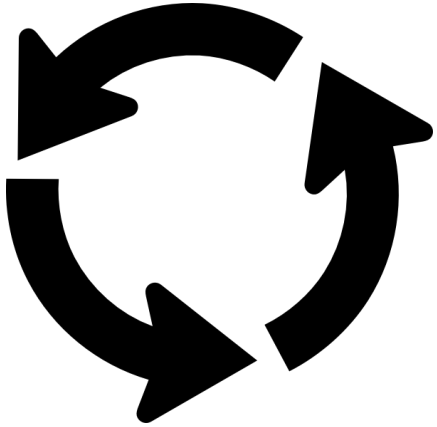
- **Virtual Memory**: program addresses **MAP** to RAM addresses
- Program loads into RAM via memory mapping.



Illusion of Extra Memory

- The mapping system places inactive data on the disk.
- Eventually, that data may become active and has to be processed
- Data may be loaded back into RAM
  - But first, something else in RAM will first have to be moved to make room for the incoming data ...

RAM
Prog 2 (RAM 2)
Prog 3 (RAM 3)
Prog 4 (RAM 0)
Prog 5 (RAM 1)
–-**FULL**---



ALLEGHENY COLLEGE

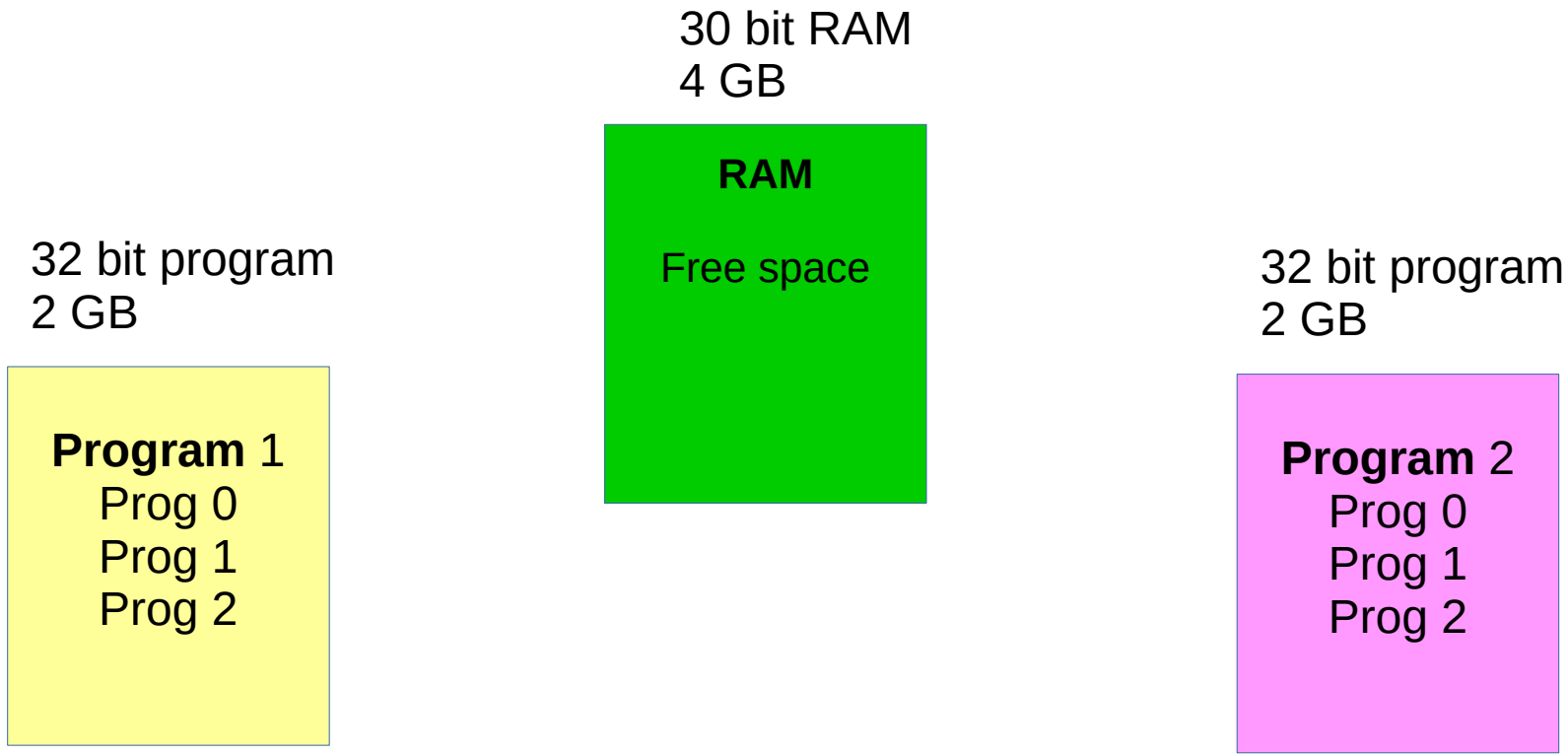
Performance Evaluation

- Virtual memory = extra memory to use for computation
- Solutions to memory limitations, right?
- How is system performance when using VM?
- How does the OS handle all the disk writes?
- What if another application tries to use disk?

ALLEGHENY COLLEGE

Fitting It All In?

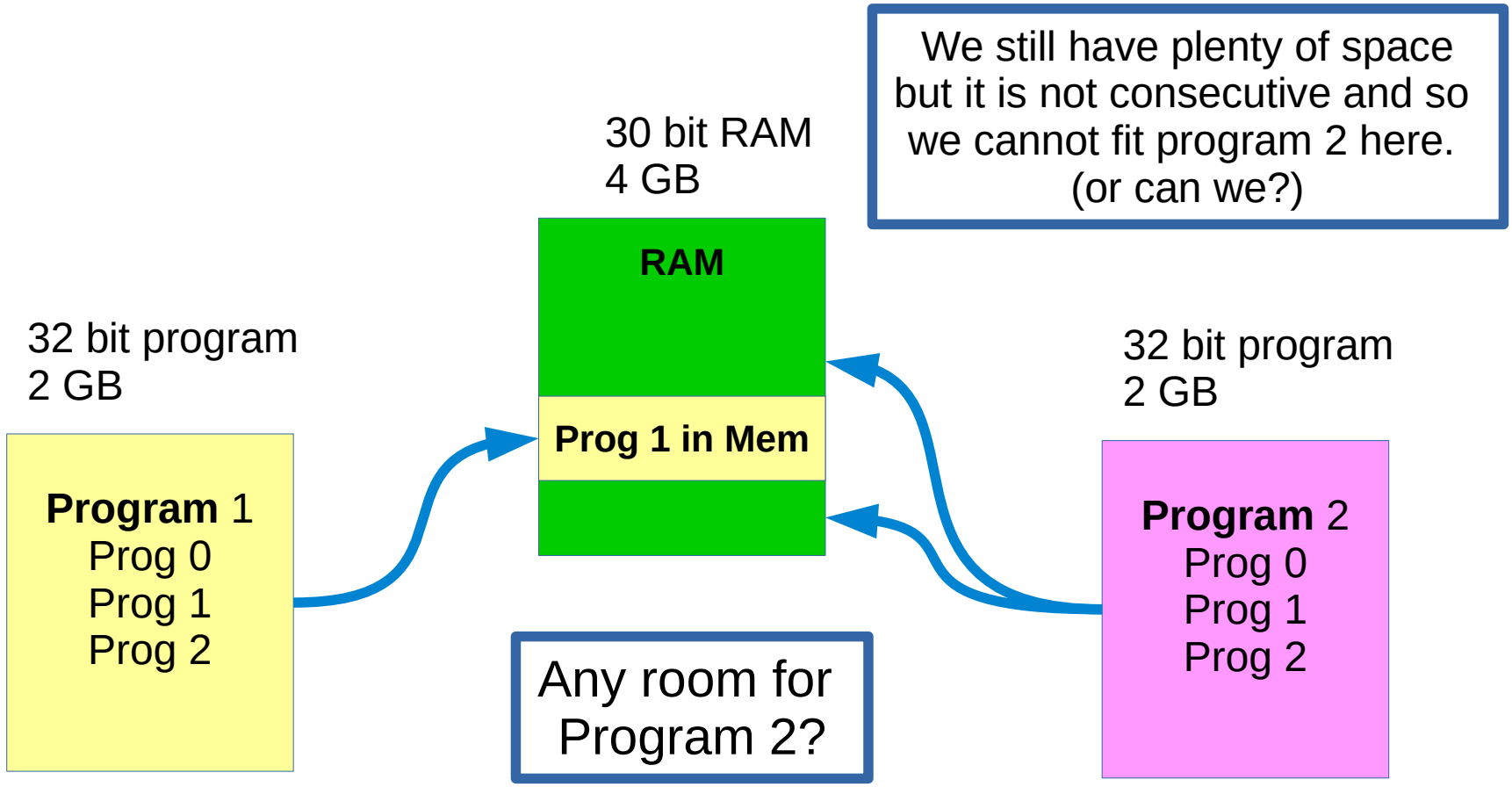
- Two programs to load into a memory space.
- No prior permissions; both load into where-ever they can.



ALLEGHENY COLLEGE

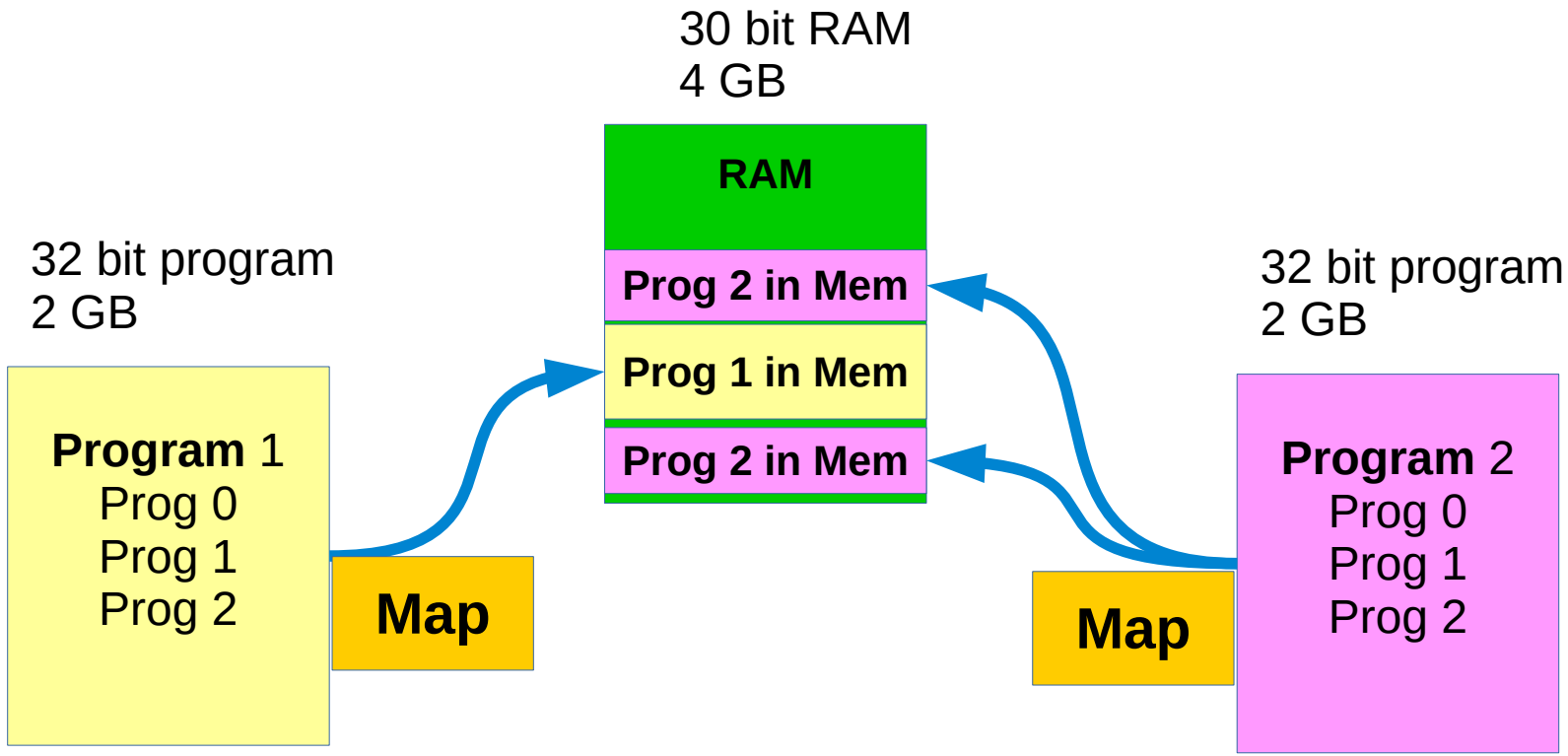
Fitting It All In?

- RAM Memory is cut in half.
- Program 2 cannot load in its entirety.



Performance Evaluation

- VM works very well when slotting data from different programs into memory “holes.”
- Make better use of memory; no empty slots.

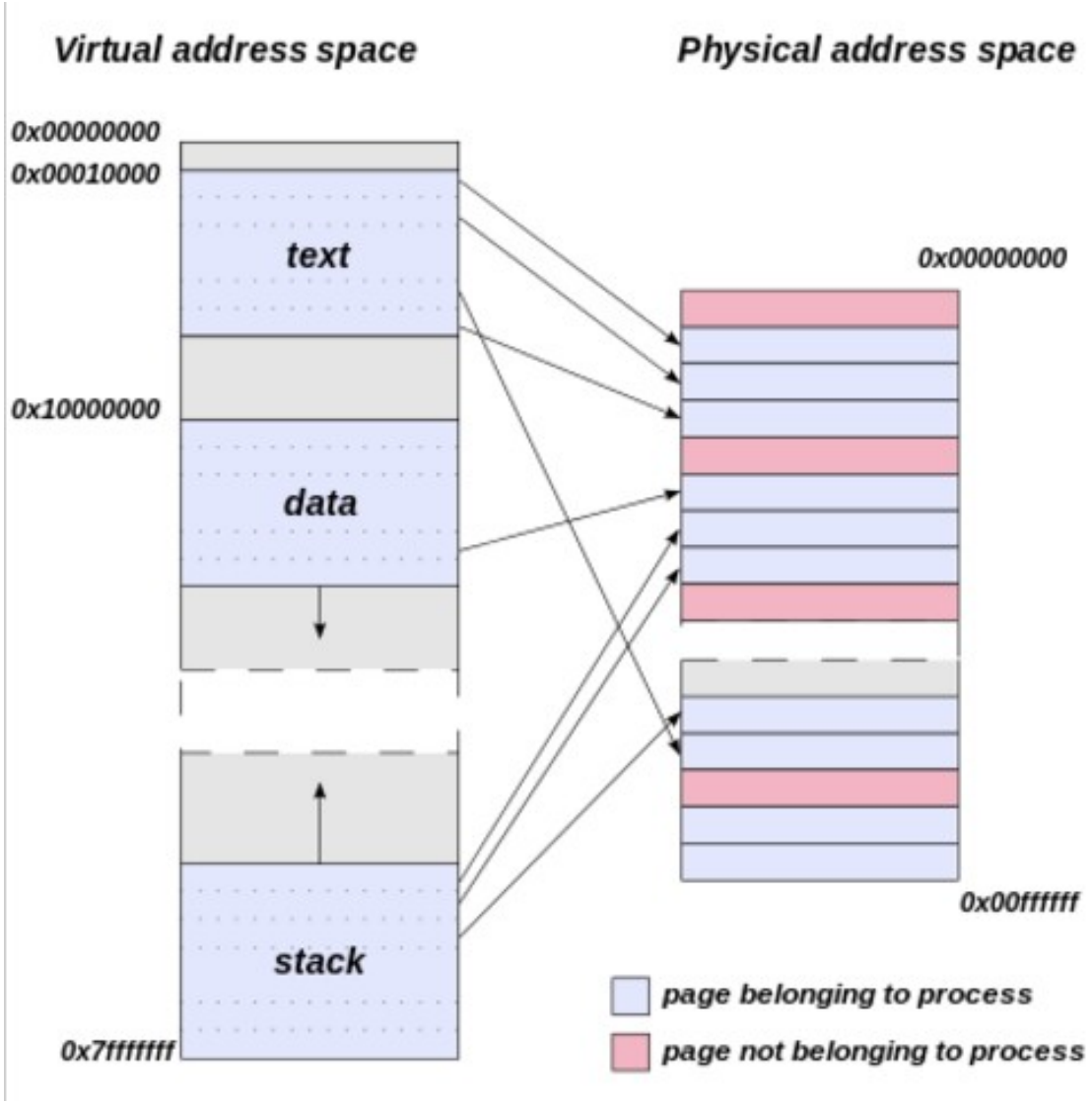


Two Maps For Two Programs

- Program 1 and program 2 both have own memory-mapping system, discrete and non-overlapping
  - Mapping provides memory addresses for each program.
- Each program has a mapping system which works around the mapping systems of other programs. (Hence, memory management).
- Question: With this model, does one program know of the existence of another?
- (and why do we care?)

Fragmented Memory

- Virtual address spaces map in an ad-hoc way to seemingly any free-standing space.
- Every process is given the impression that it is working with large, contiguous sections of memory

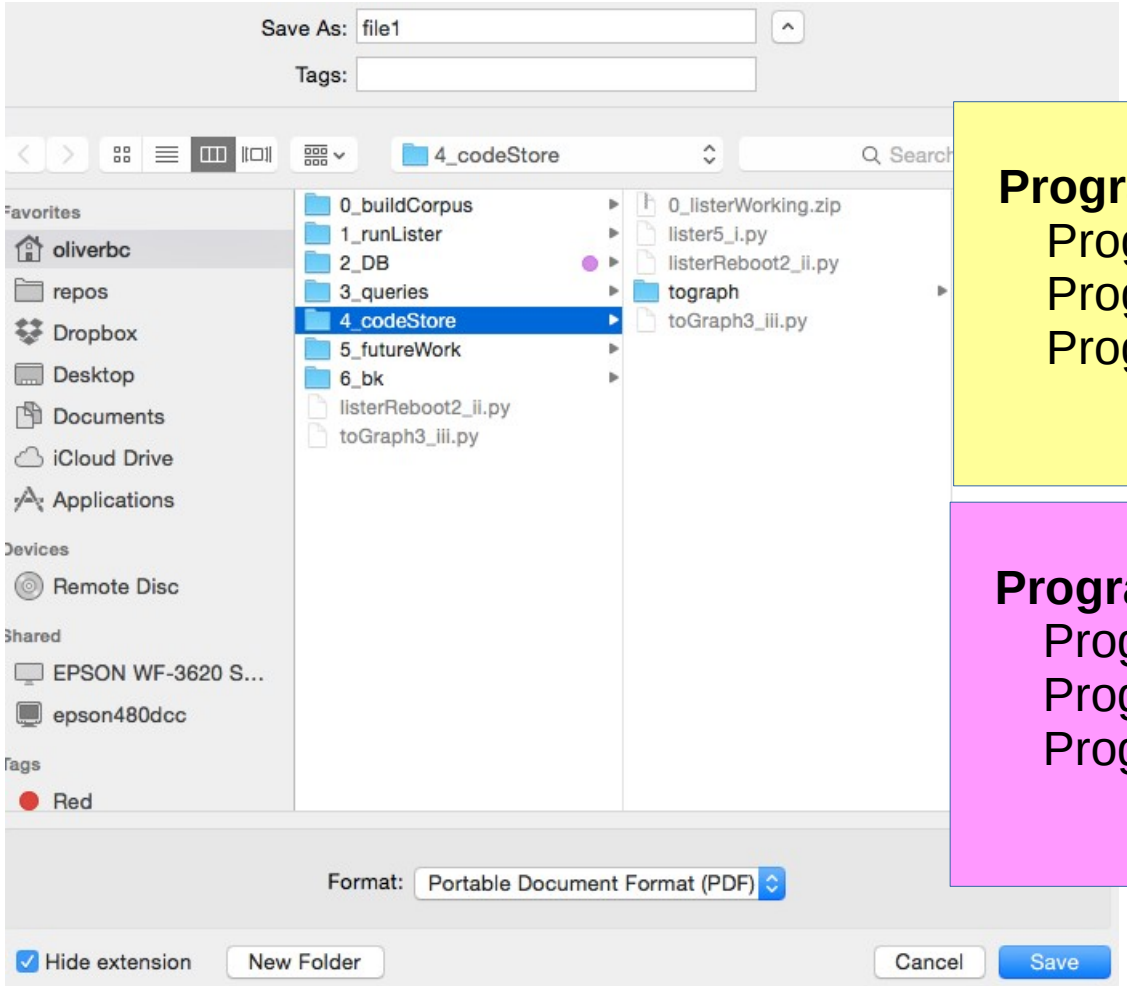


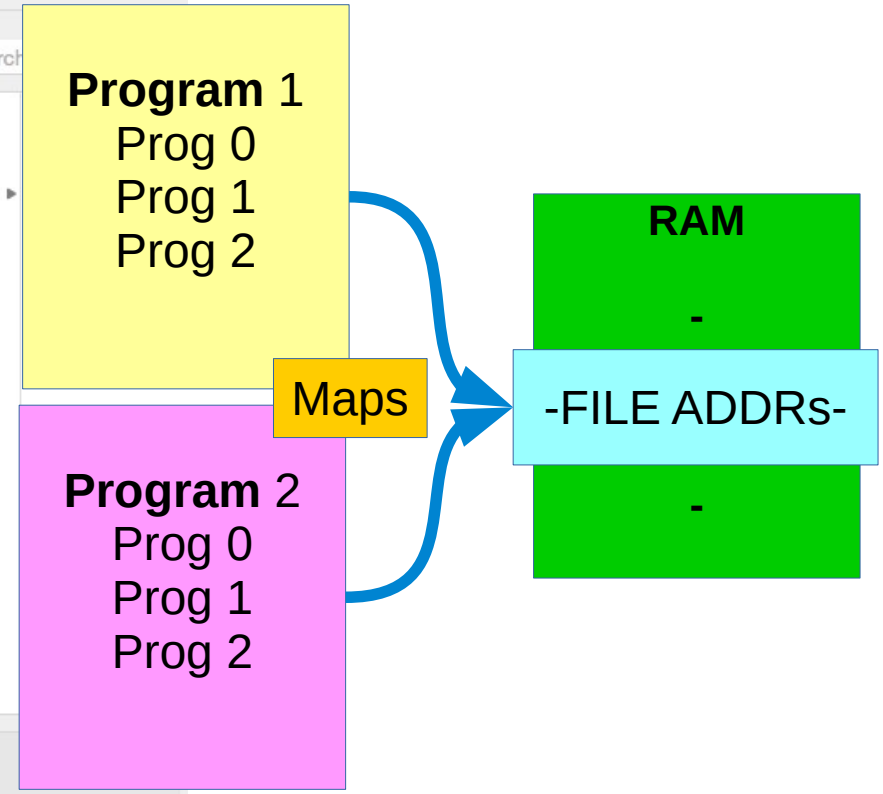
Sharing Memory

- Can you think of a reason why you would *ever* want to share data (a memory space) between an arbitrary number of programs?
- What about saving files.
- Each program that saves files will have to know what the directory structure looks like.
  - Files
  - Directories
  - Libraries
  - Drivers
- The OS shares these addresses with all programs.

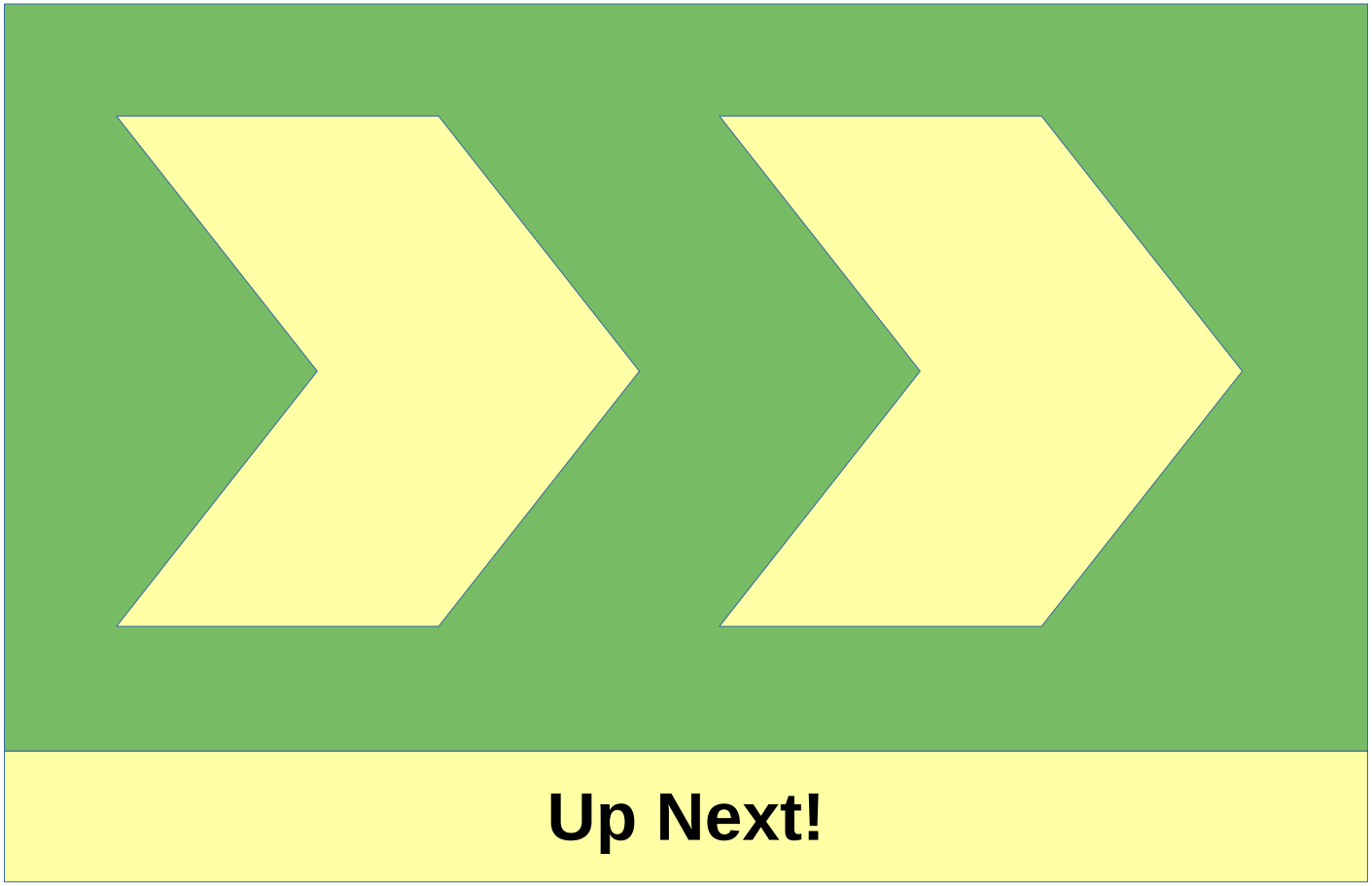
Sharing is Caring

- Sharing File addresses by OS





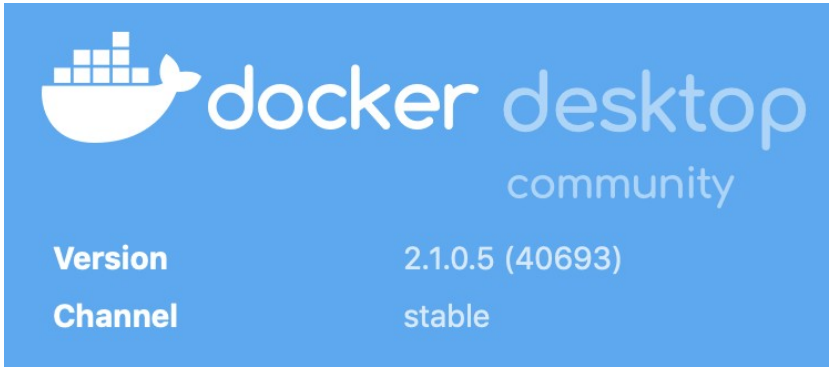
Let's Code!



ALLEGHENY COLLEGE

Commands to Run
From (Linux) Bash

- Build the container :
  - docker build -t gccdev .
- Run the container :
  - docker run -it gccdev
- Mount local drive and run container :
  - docker run -it --mount type=bind,source=$PWD,target=/home/gccdev gccdev



Note: the directory where you run this becomes your local directory in the container.