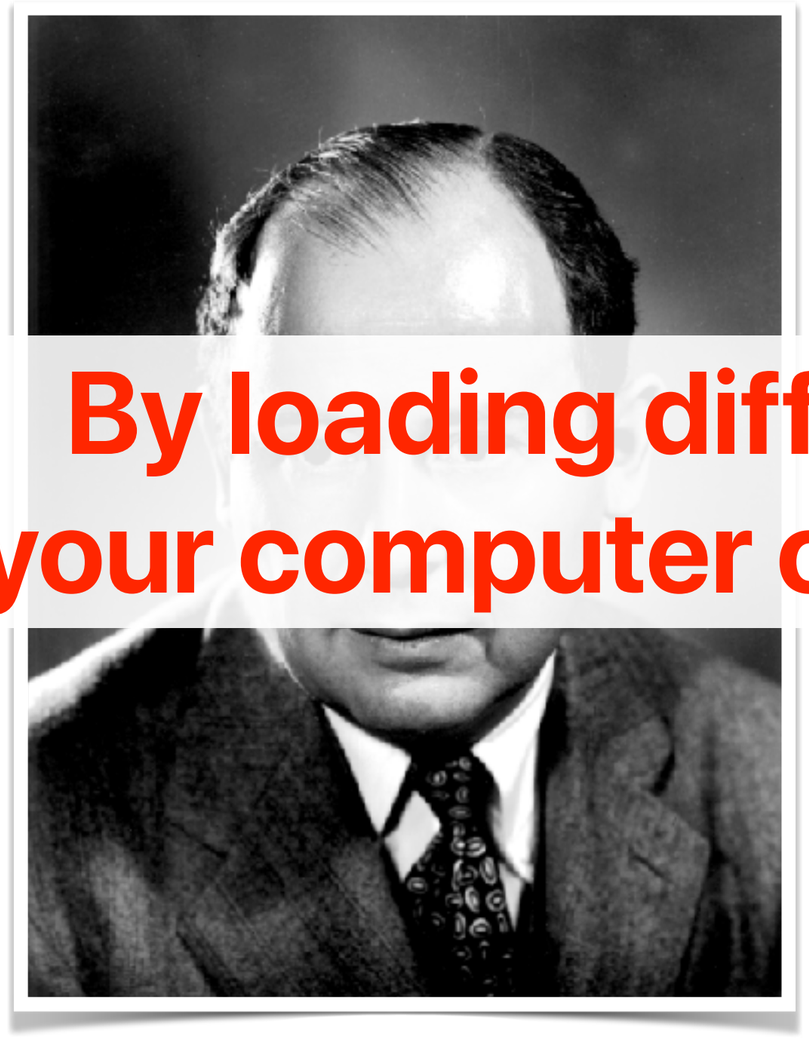
Performance (2): What matters?

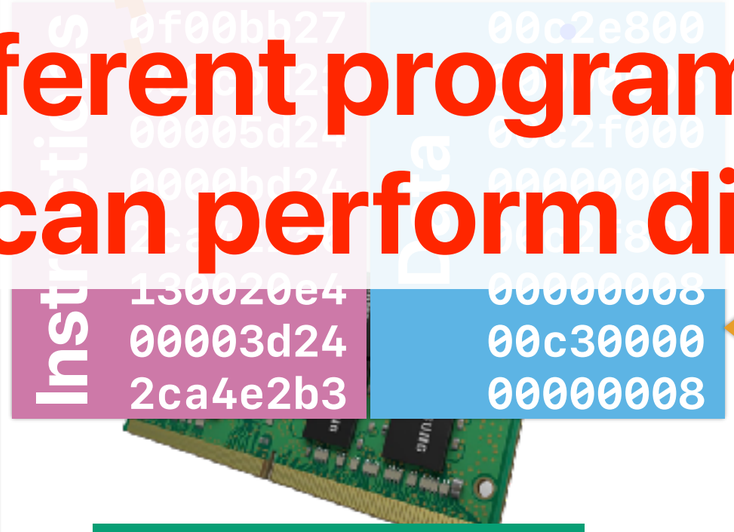
Hung-Wei Tseng

Recap: von Neumman Architecture

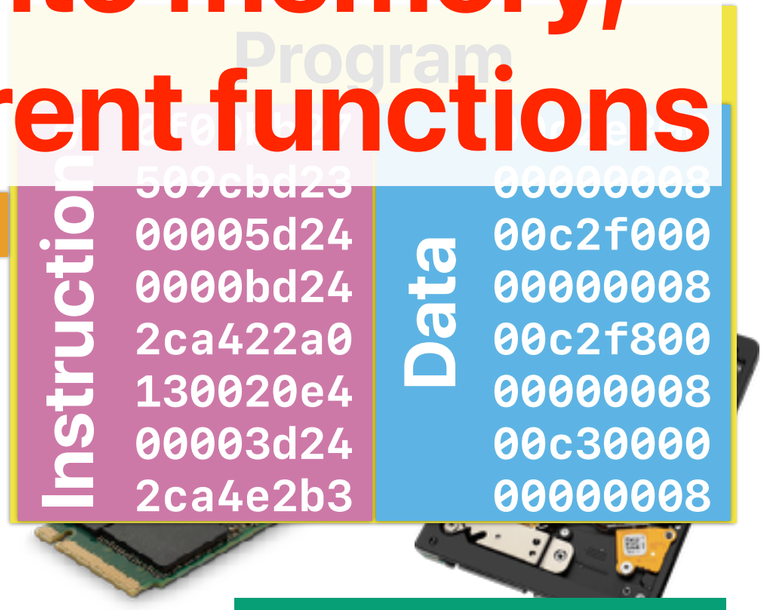


Processor

By loading different programs into memory, your computer can perform different functions



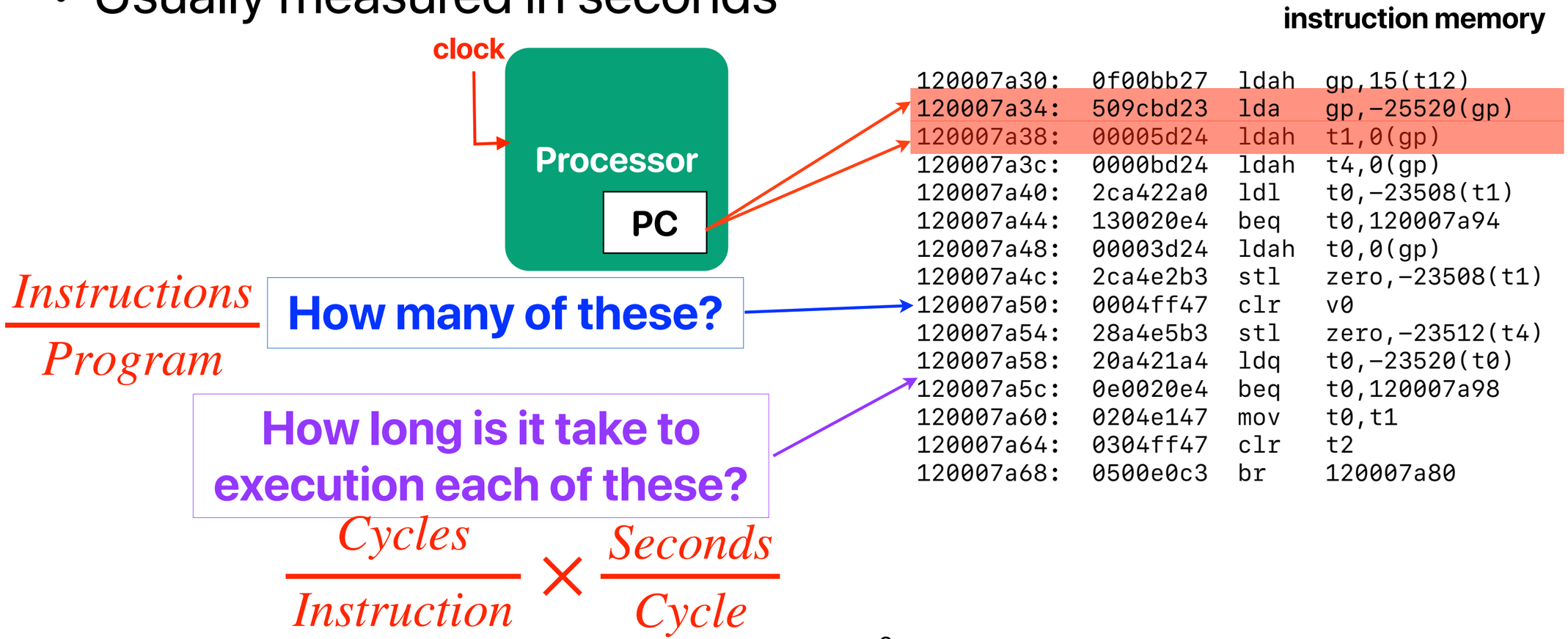
Memory



Storage

Recap: Execution Time

- The simplest kind of performance
- Shorter execution time means better performance
- Usually measured in seconds



Recap: Performance & Speedup

$$Performance = \frac{1}{Execution\ Time}$$

$$Execution\ Time = \frac{Instructions}{Program} \times \frac{Cycles}{Instruction} \times \frac{Seconds}{Cycle}$$

$$ET = IC \times CPI \times CT$$

$$Speedup = \frac{Execution\ Time_X}{Execution\ Time_Y}$$

3:00

If you can change just “one thing” in your life routine, what’s that going to be & why?

Outline

- What affects each factor in “Performance Equation”
- Amdahl’s Law

What Affects Each Factor in Performance Equation (Cont.)

How programmer affects performance?

- Performance equation consists of the following three factors
  1. IC
  2. CPI
  3. CT

  How many can a **programmer** affect?

  A. 0
  B. 1
  C. 2
  D. 3

Use “performance counters” to figure out!

- Modern processors provides performance counters
  - instruction counts
  - cache accesses/misses
  - branch instructions/mis-predictions
- How to get their values?
  - You may use “perf stat” in linux
  - You may use Instruments —> Time Profiler on a Mac
  - Intel’s vtune — only works on Windows w/ intel processors
  - You can also create your own functions to obtain counter values

Demo — programmer & performance

A

```
for(i = 0; i < ARRAY_SIZE; i++)
{
  for(j = 0; j < ARRAY_SIZE; j++)
  {
    c[i][j] = a[i][j]+b[i][j];
  }
}
```

B

```
for(j = 0; j < ARRAY_SIZE; j++)
{
  for(i = 0; i < ARRAY_SIZE; i++)
  {
    c[i][j] = a[i][j]+b[i][j];
  }
}
```

$O(n^2)$	Complexity	$O(n^2)$
Same	Instruction Count?	Same
Same	Clock Rate	Same
Better	CPI	Worse

Programmer's impact

- By adding the "sort" in the following code snippet, what the programmer changes in the performance equation to achieve **better** performance?

```
std::sort(data, data + arraySize);

for (unsigned c = 0; c < arraySize*1000; ++c) {
    if (data[c%arraySize] >= INT_MAX/2)
        sum ++;
}
}
```

A. CPI

B. IC ← **programmer changes IC as well, but not in the positive direction**

C. CT

D. IC & CPI

E. CPI & CT

Programmers can also set the cycle time

https://software.intel.com/sites/default/files/comment/1716807/how-to-change-frequency-on-linux-pub.txt

```
==========================================
Subject: setting CPU speed on running linux system


If the OS is Linux, you can manually control the CPU speed by reading and writing some virtual files in the "/proc"

1.) Is the system capable of software CPU speed control?
If the "directory" /sys/devices/system/cpu/cpu0/cpufreq exists, speed is controllable.
-- If it does not exist, you may need to go to the BIOS and turn on EIST and any other C and P state control and vi



2.) What speed is the box set to now?
Do the following:
$ cd /sys/devices/system/cpu
$ cat ./cpu0/cpufreq/cpuinfo_max_freq
3193000
$ cat ./cpu0/cpufreq/cpuinfo_min_freq
1596000

3.) What speeds can I set to?
Do
$ cat /sys/devices/system/cpu/cpu0/cpufreq/scaling_available_frequencies
It will list highest settable to lowest; example from my NHM "Smackover" DX58SO HEDT board, I see:
3193000 3192000 3059000 2926000 2793000 2660000 2527000 2394000 2261000 2128000 1995000 1862000 1729000 159600
You can choose from among those numbers to set the "high water" mark and "low water" mark for speed.  If you set "h


4.) Show me how to set all to highest settable speed!
Use the following little sh/ksh/bash script:
$ cd /sys/devices/system/cpu  # a virtual directory made visible by device drivers
$ newSpeedTop=`awk '{print $1}' ./cpu0/cpufreq/scaling_available_frequencies`
$ newSpeedLow=$newSpeedTop  # make them the same in this example
$ for c in ./cpu[0-9]* ; do
>   echo $newSpeedTop >${c}/cpufreq/scaling_max_freq
>   echo $newSpeedLow >${c}/cpufreq/scaling_min_freq
> done
$

5.) How do I return to the default - i.e. allow machine to vary from highest to lowest?
Edit line # 3 of the script above, and re-run it.  Change the line:
$ newSpeedLow=$newSpeedTop  # make them the same in this example
```

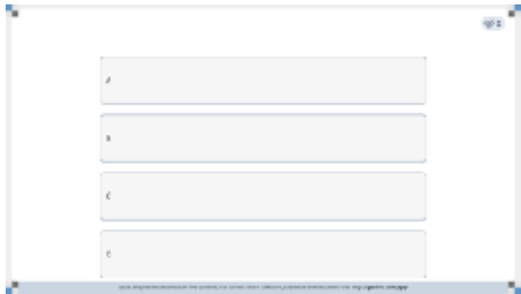
https://www.pollev.com/hungweitseng close in 1:30

How programming languages affect performance

- Performance equation consists of the following three factors
  ① IC
  ② CPI
  ③ CT

How many can the **programming language** affect?

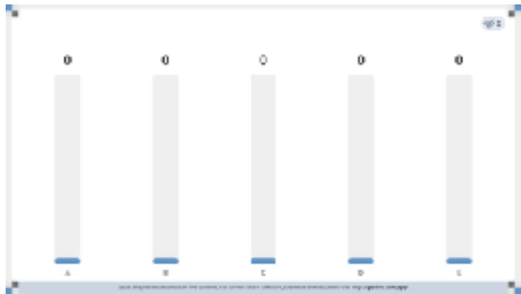
A. 0
B. 1
C. 2
D. 3



https://www.pollev.com/hungweitseng close in 1:30

Programming languages

- Which of the following programming language needs to highest instruction count to print “Hello, world!” on screen?
  A. C
  B. C++
  C. Java
  D. Perl
  E. Python



Programming languages

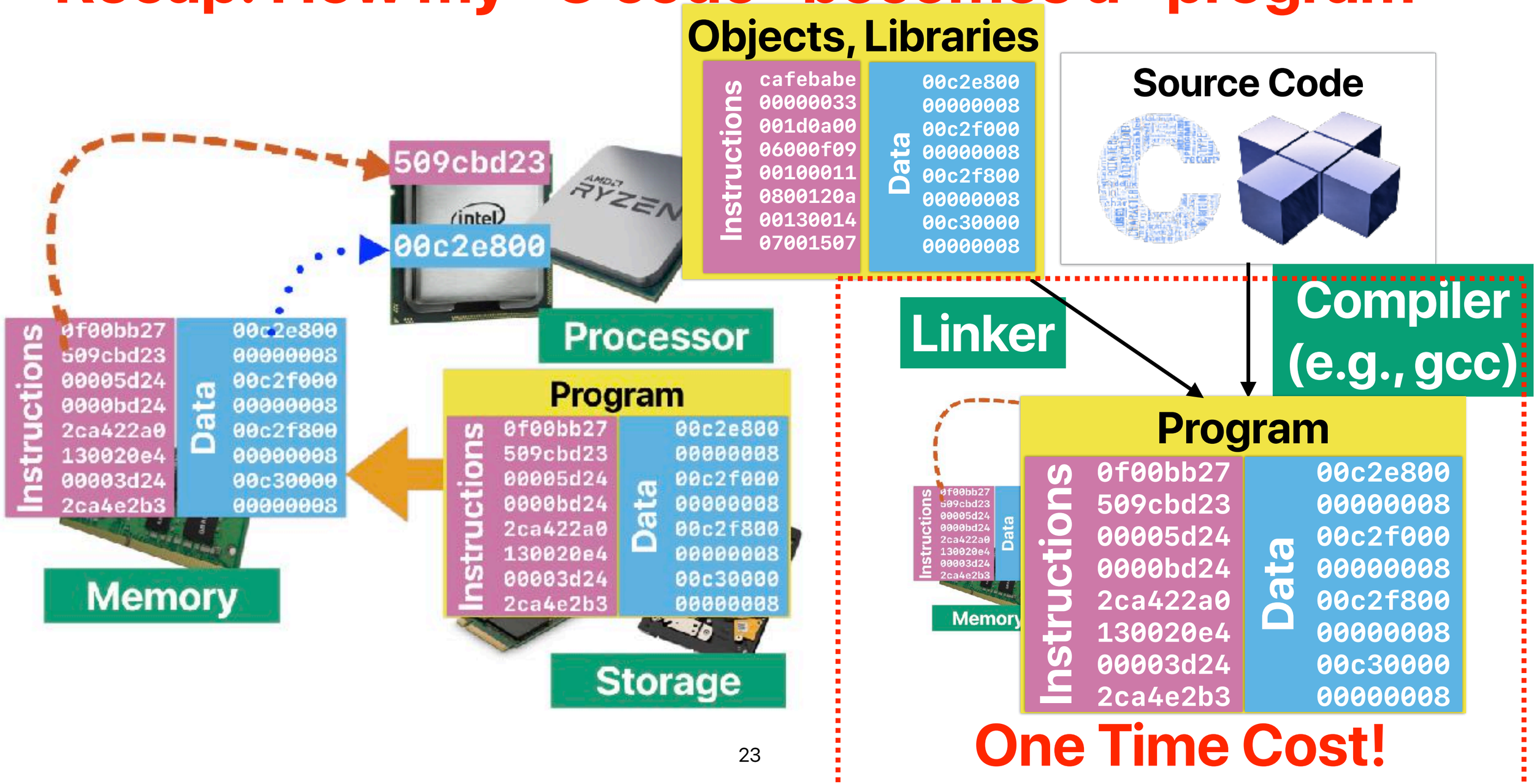
- How many instructions are there in “Hello, world!”

	Instruction count	LOC	Ranking
C	600k	6	1
C++	3M	6	2
Java	~145M	8	5
Perl	~12M	4	3
Python	~33M	1	4
GO (Interpreter)	~1200M	1	6
GO (Compiled)	~1.7M	1	
Rust	~1.4M	1	

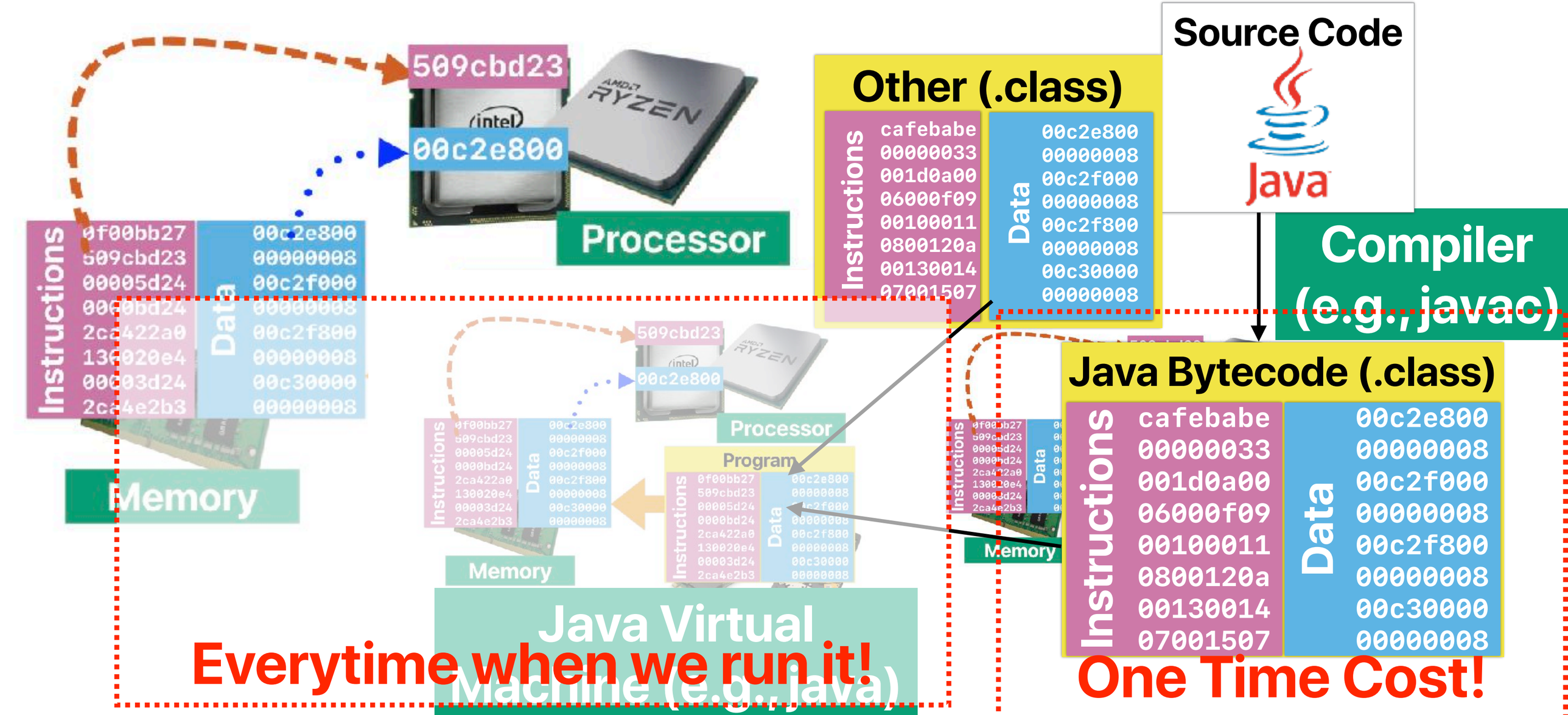
Programming languages

- Which of the following programming language needs to highest instruction count to print “Hello, world!” on screen?
  A. C
  B. C++
  C. Java
  D. Perl
  E. Python

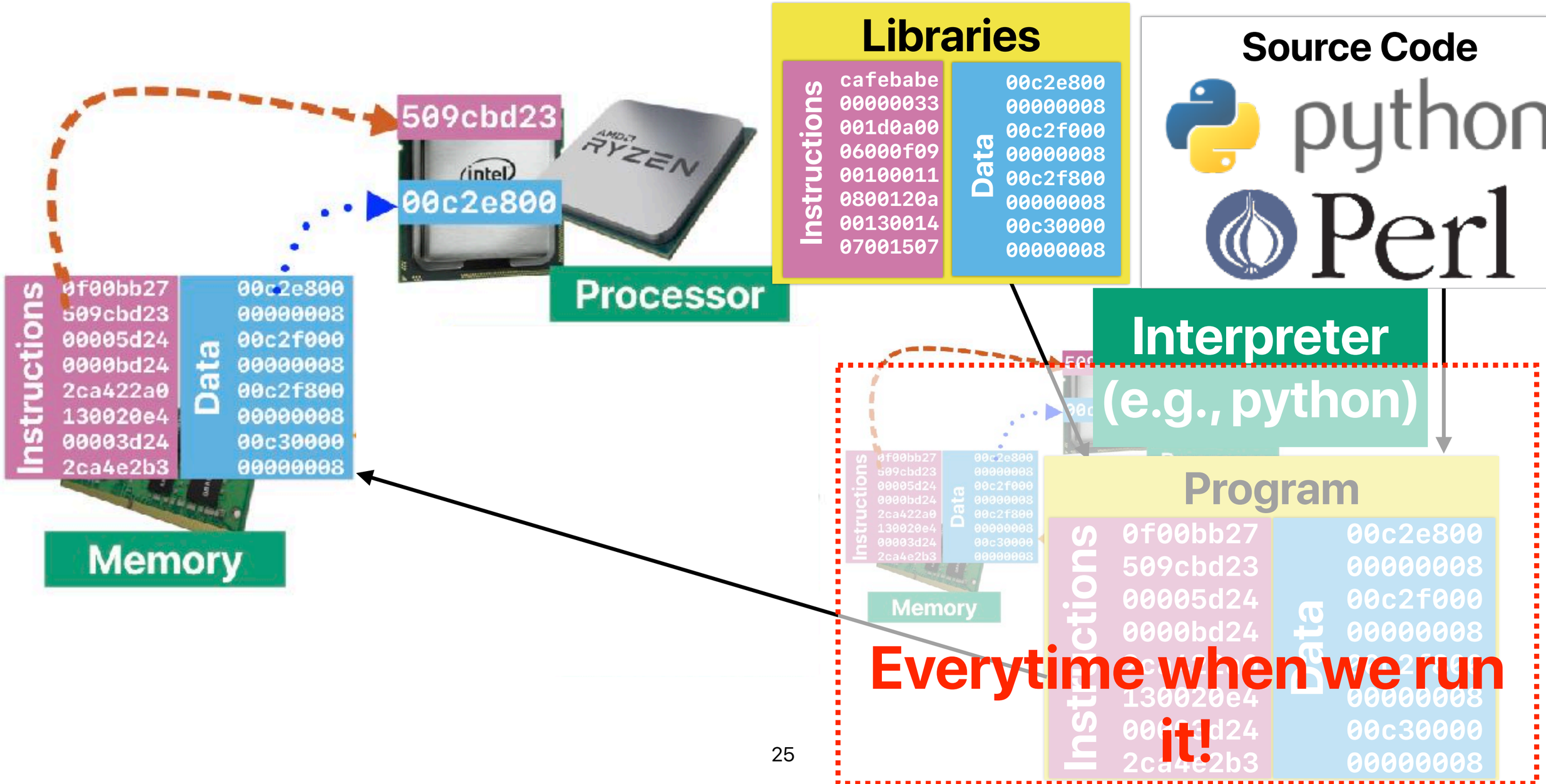
Recap: How my "C code" becomes a "program"



Recap: How my "Java code" becomes a "program"



Recap: How my "Python code" becomes a "program"



How programming languages affect performance

- Performance equation consists of the following three factors
  1. IC
  2. CPI
  3. CT

Programmer uses programming languages to create library/ programs that changes the CT, not the programming language itself makes the change

How many can the **programming language** affect?

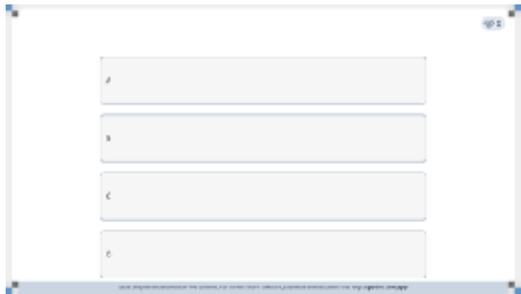
A. 0
B. 1
C. 2
D. 3

How compilers affect performance

- Performance equation consists of the following three factors
  ① IC
  ② CPI
  ③ CT

  How many can the **compiler** affect?

  A. 0
  B. 1
  C. 2
  D. 3



How compilers affect performance

- Performance equation consists of the following three factors
  1. IC
  2. CPI
  3. CT

Compiler is less effective than programmer if the programmer wisely tweaked the code based on architecture!

How many can the **compiler** affect?

A. 0
B. 1
C. 2
D. 3

Revisited the demo with compiler optimizations!

- gcc has different optimization levels.
  - -O0 — no optimizations
  - -O3 — typically the best-performing optimization

A

```
for(i = 0; i < ARRAY_SIZE; i++)
{
  for(j = 0; j < ARRAY_SIZE; j++)
  {
    c[i][j] = a[i][j]+b[i][j];
  }
}
```

B

```
for(j = 0; j < ARRAY_SIZE; j++)
{
  for(i = 0; i < ARRAY_SIZE; i++)
  {
    c[i][j] = a[i][j]+b[i][j];
  }
}
```

How about complexity?

How about “computational complexity”

- Algorithm complexity provides a good estimate on the performance if —
  - Every instruction takes exactly the same amount of time
  - Every operation takes exactly the same amount of instructions

These are unlikely to be true

Summary of CPU Performance Equation

$$Performance = \frac{1}{Execution\ Time}$$

$$Execution\ Time = \frac{Instructions}{Program} \times \frac{Cycles}{Instruction} \times \frac{Seconds}{Cycle}$$

$$ET = IC \times CPI \times CT$$

- IC (Instruction Count)
  - ISA, Compiler, algorithm, programming language, **programmer**
- CPI (Cycles Per Instruction)
  - Machine Implementation, microarchitecture, compiler, application, algorithm, programming language, **programmer**
- Cycle Time (Seconds Per Cycle)
  - Process Technology, microarchitecture, **programmer**

Amdahl's Law — and It's Implication in the Multicore Era

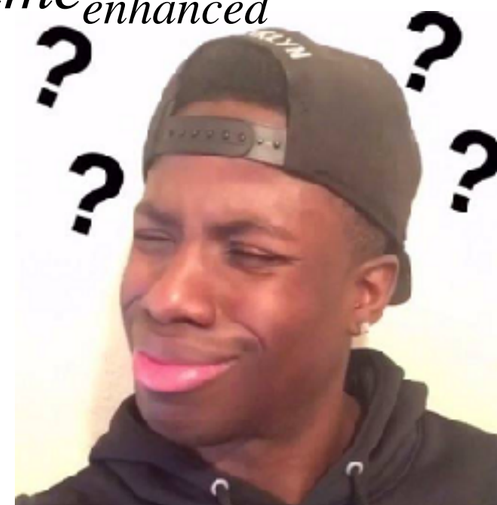
Amdahl's Law

$$Speedup_{enhanced}(f, s) = \frac{1}{(1 - f) + \frac{f}{s}}$$

f — The fraction of time in the original program

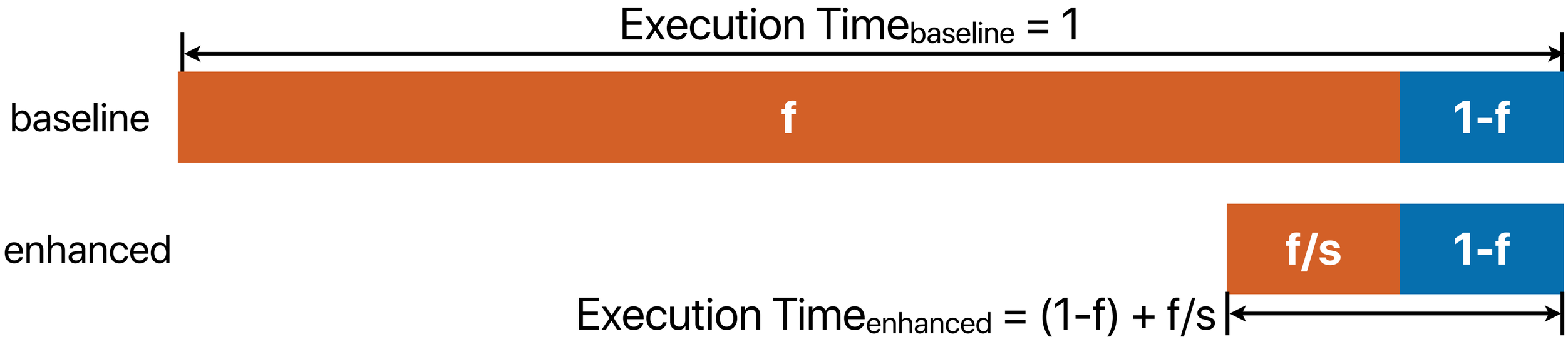
s — The speedup we can achieve on f

$$Speedup_{enhanced} = \frac{Execution\ Time_{baseline}}{Execution\ Time_{enhanced}}$$



Amdahl's Law

$$Speedup_{enhanced}(f, s) = \frac{1}{(1 - f) + \frac{f}{s}}$$

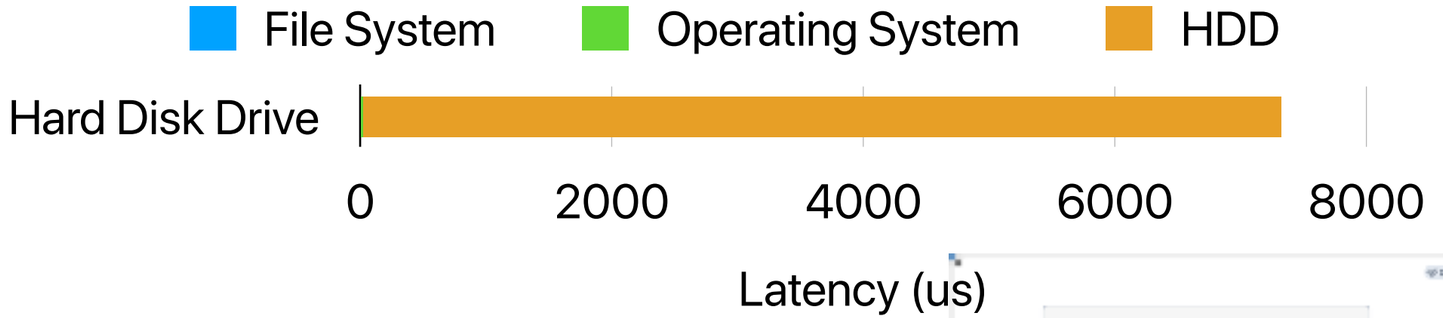


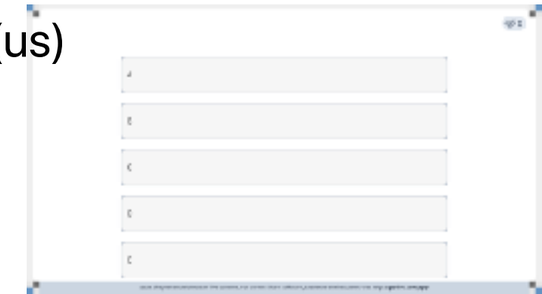
$$Speedup_{enhanced} = \frac{Execution\ Time_{baseline}}{Execution\ Time_{enhanced}} = \frac{1}{(1 - f) + \frac{f}{s}}$$

https://www.pollev.com/hungweitseng close in 1:30

Practicing Amdahl's Law

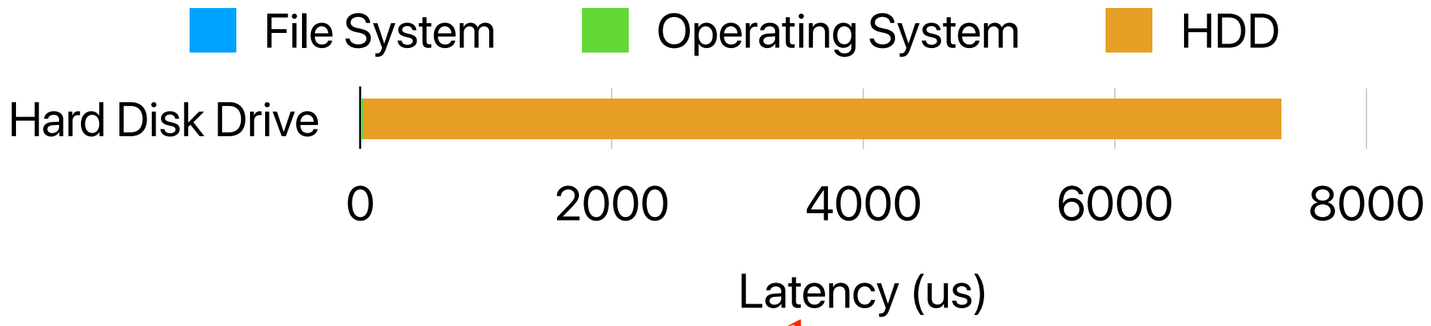
- Final Fantasy XV spends lots of time loading a map — within which period that 95% of the time on the accessing the H.D.D., the rest in the operating system, file system and the I/O protocol. If we replace the H.D.D. with a flash drive, which provides 100x faster access time. By how much can we speed up the map loading process?
  - A. ~7x
  - B. ~10x
  - C. ~17x
  - D. ~29x
  - E. ~100x





Practicing Amdahl's Law

- Final Fantasy XV spends lots of time loading a map — within which period that 95% of the time on the accessing the H.D.D., the rest in the operating system, file system and the I/O protocol. If we replace the H.D.D. with a flash drive, which provides 100x faster access time. By how much can we speed up the map loading process?
  - A. ~7x
  - B. ~10x
  - C. ~17x
  - D. ~29x
  - E. ~100x



$$Speedup_{enhanced}(95\,\%, 100) = \frac{1}{(1 - 95\%) + \frac{95\,\%}{100}} = 16.81\times$$

Amdahl's Law on Multiple Optimizations

- We can apply Amdahl's law for multiple optimizations
- These optimizations must be dis-joint!
  - If optimization #1 and optimization #2 are dis-joint:

$$Speedup_{enhanced}(f_{Opt1}, f_{Opt2}, s_{Opt1}, s_{Opt2}) = \frac{1}{(1 - f_{Opt1} - f_{Opt2}) + \frac{f\_Opt1}{s\_Opt1} + \frac{f\_Opt2}{s\_Opt2}}$$

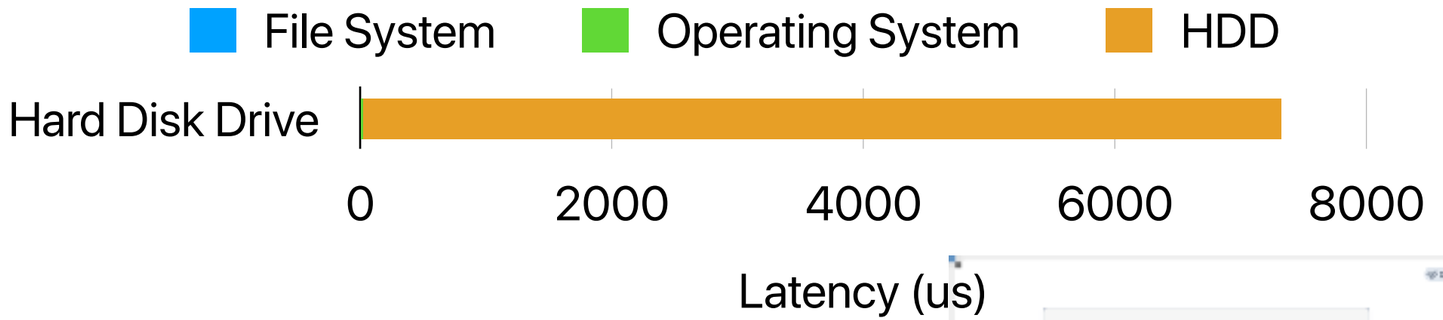
  - If optimization #1 and optimization #2 are not dis-joint:

$$Speedup_{enhanced}(f_{OnlyOpt1}, f_{OnlyOpt2}, f_{BothOpt1Opt2}, s_{OnlyOpt1}, s_{OnlyOpt2}, s_{BothOpt1Opt2}) = \frac{1}{(1 - f_{OnlyOpt1} - f_{OnlyOpt2} - f_{BothOpt1Opt2}) + + \frac{f\_BothOpt1Opt2}{s\_BothOpt1Opt2} + \frac{f\_OnlyOpt1}{s\_OnlyOpt1} + \frac{f\_OnlyOpt2}{s\_OnlyOpt2}}$$

Practicing Amdahl's Law (2)

- Final Fantasy XV spends lots of time loading a map — within which period that 95% of the time on the accessing the H.D.D., the rest in the operating system, file system and the I/O protocol. If we replace the H.D.D. with a flash drive, which provides 100x faster access time and a better processor to accelerate the software overhead by 2x. By how much can we speed up the map loading process?

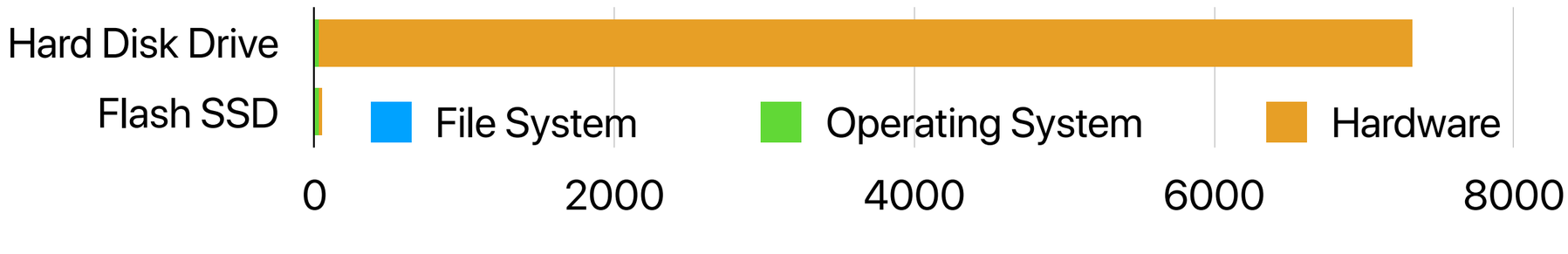
  A. ~7x
  B. ~10x
  C. ~17x
  D. ~29x
  E. ~100x



Practicing Amdahl's Law (2)

- Final Fantasy XV spends lots of time loading a map — within which period that 95% of the time on the accessing the H.D.D., the rest in the operating system, file system and the I/O protocol. If we replace the H.D.D. with a flash drive, which provides 100x faster access time and a better processor to accelerate the software overhead by 2x. By how much can we speed up the map loading process?

A. ~7x
B. ~10x
C. ~17x
D. ~29x
E. ~100x



$$Speedup_{enhanced}(95\%, 5\%, 100, 2) = \frac{1}{(1 - 95\% - 5\%) + \frac{95\%}{100} + \frac{5\%}{2}} = 28.98\times$$

Announcement

- Assignment 1 released. Due Sunday midnight.
  - Jupyter notebook based template, export as PDF when you turn in
- Reading quiz due next Monday before the lecture
  - We will drop two of your least performing reading quizzes
  - You have two shots, both unlimited time
- Course resources
  - Slides/Assignments: https://www.escalab.org/classes/cse142-2023su/
  - Reading quizzes & grading: https://canvas.ucsd.edu/courses/48362
  - Lecture recordings: https://www.youtube.com/c/ProfUsagi/playlists
  - Turnin assignments: https://www.gradescope.com/courses/564382
  - Discussion: https://piazza.com/class/lkwy1ccytqc3ct

Computer
Science &
Engineering

142

つづく