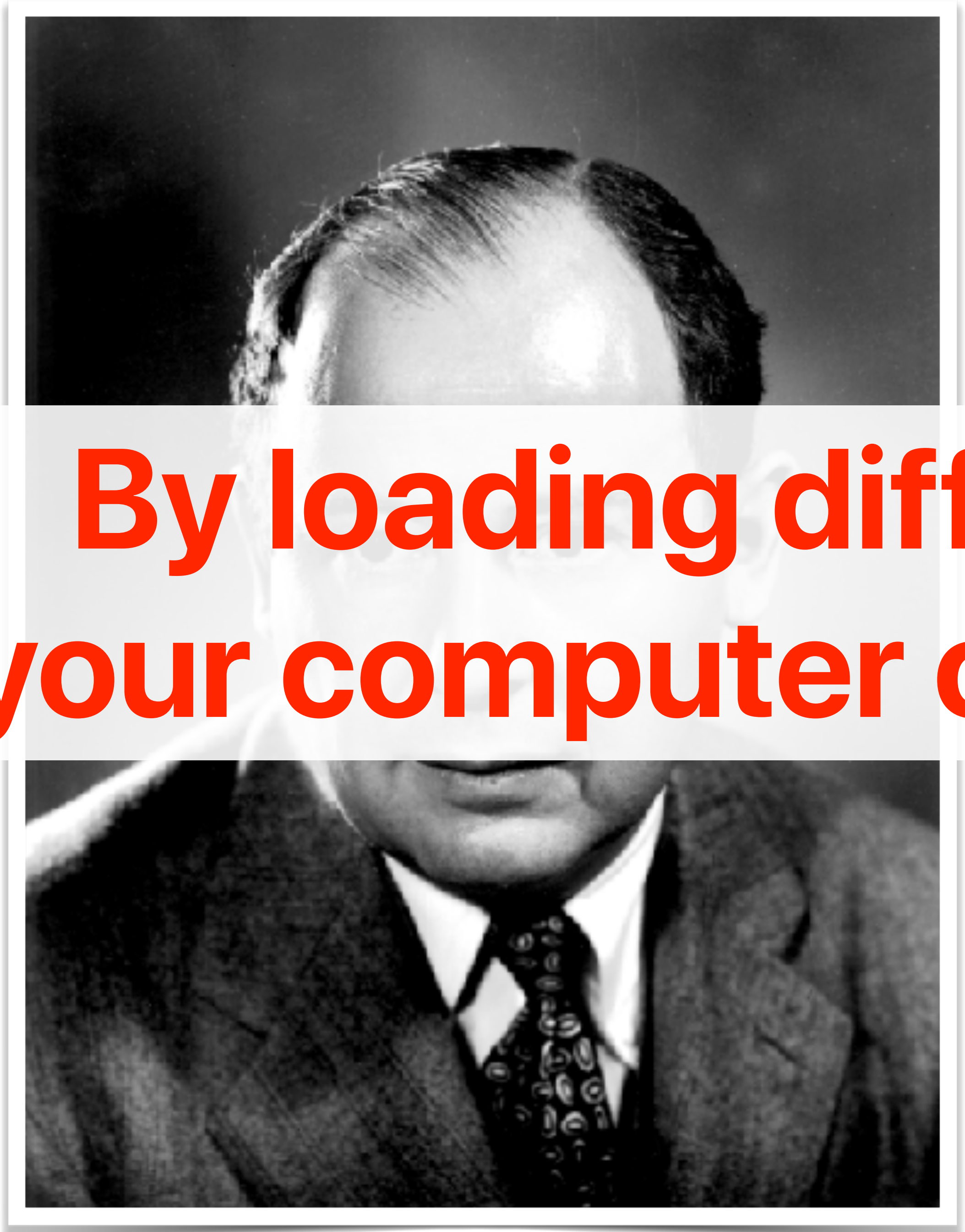
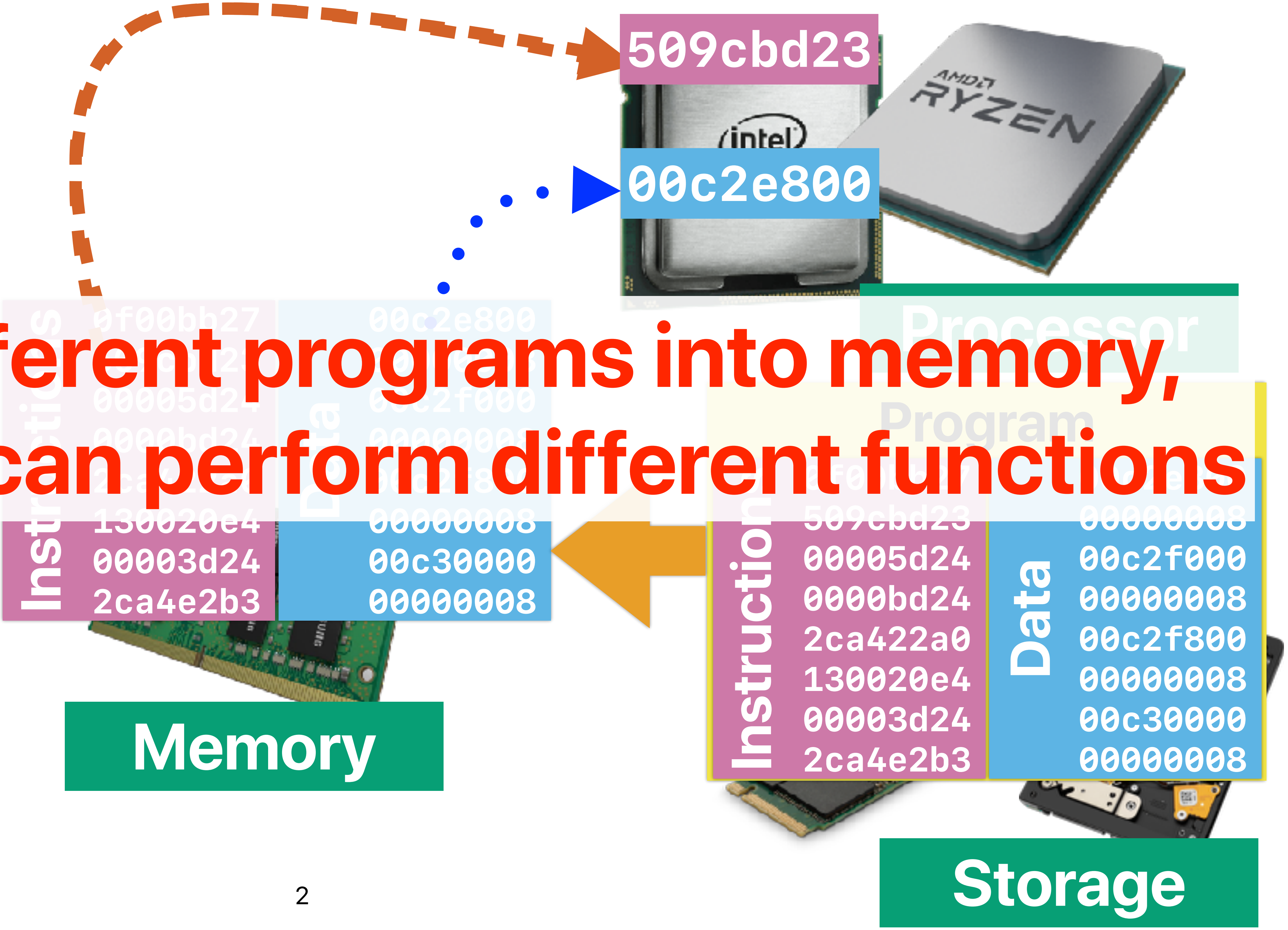
Performance (2): Who can make it better?

Hung-Wei Tseng

Recap: von Neumman Architecture





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

Recap: Important performance metrics

- End-to-end latency — how much **time** the program/operation takes from the beginning to the end
- Response time — how much **time** the user starts to feel the program is running/finishing
- Throughput/bandwidth — the average amount of work/data can the program/system deliver within the execution **time**
- Energy consumption — the aggregated power during the execution **time**
- Cost of operation — the amount of money necessary for finishing an operation (related to **time**)
- Quality of results — the human perception of the execution result
- Power consumption — the heat generation produced by the circuit

Recap: definition of performance and how to be fair

- Latency (ET) is the most fundamental performance metric
- Instruction count, cycles per instruction, cycle time define the latency of execution on CPUs — $ET = IC \times CPI \times CT$
- As $TFLOPS = \frac{\cancel{IC} \times \%\ of\ floating\ point\ instructions \times 10^{-12}}{\cancel{IC} \times CPI \times CT}$ crosses out the effect of IC, FLOS may not be ideal for cases where changes of infrastructure or the amount of computation occur

3:00

What’s your favorite programming language and why?

Outline

- What affects performance and to what degree
  - You
  - Programming languages
  - Compilers
- Why computational complexity fails on real machines

What Affects Each Factor in Performance Equation

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

What can programmers affect?

- 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

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** $O(n^2)$

Instruction Count?

Clock Rate

CPI

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];
  }
}
```

How many of the following make(s) the performance different between version A & version B?

① IC
② CPI
③ CT

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

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];
  }
}
```

A		B
$O(n^2)$	Complexity	$O(n^2)$
Same	Instruction Count?	Same
Same	Clock Rate	Same
???	CPI	???

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];
  }
}
```

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

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];
  }
}
```

How many of the following make(s) the performance different between version A & version B?

① IC
② CPI
③ CT

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

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

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
C. CT
D. IC & CPI
E. CPI & CT

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
C. CT
D. IC & CPI
E. CPI & CT

programmer changes IC as well, but not in the positive direction

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
```

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

Takeaways: What matters?

- Programmers can control all three factors in the classic performance equation

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

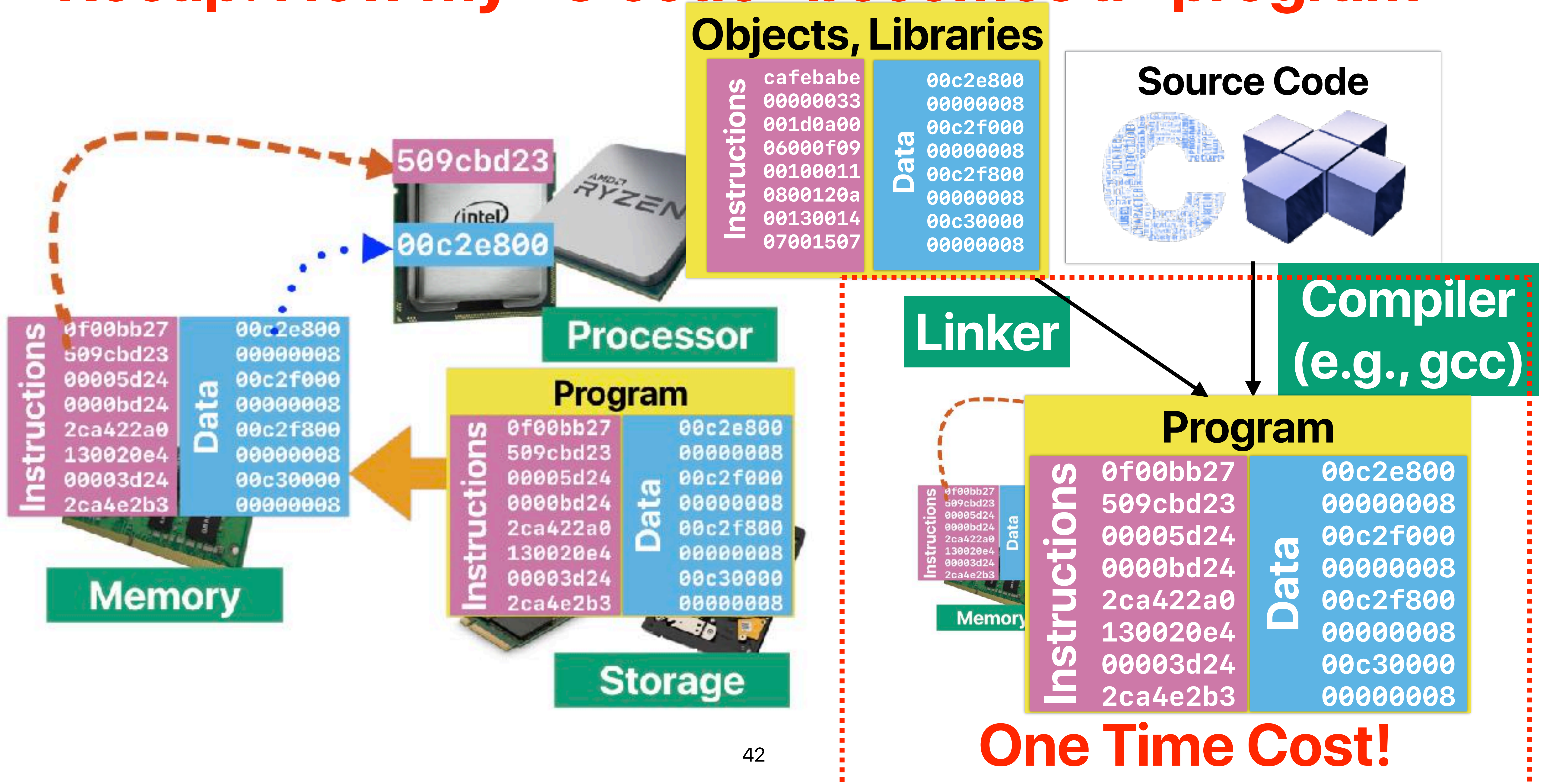
- How many instructions are there in "Hello, world!"

	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	

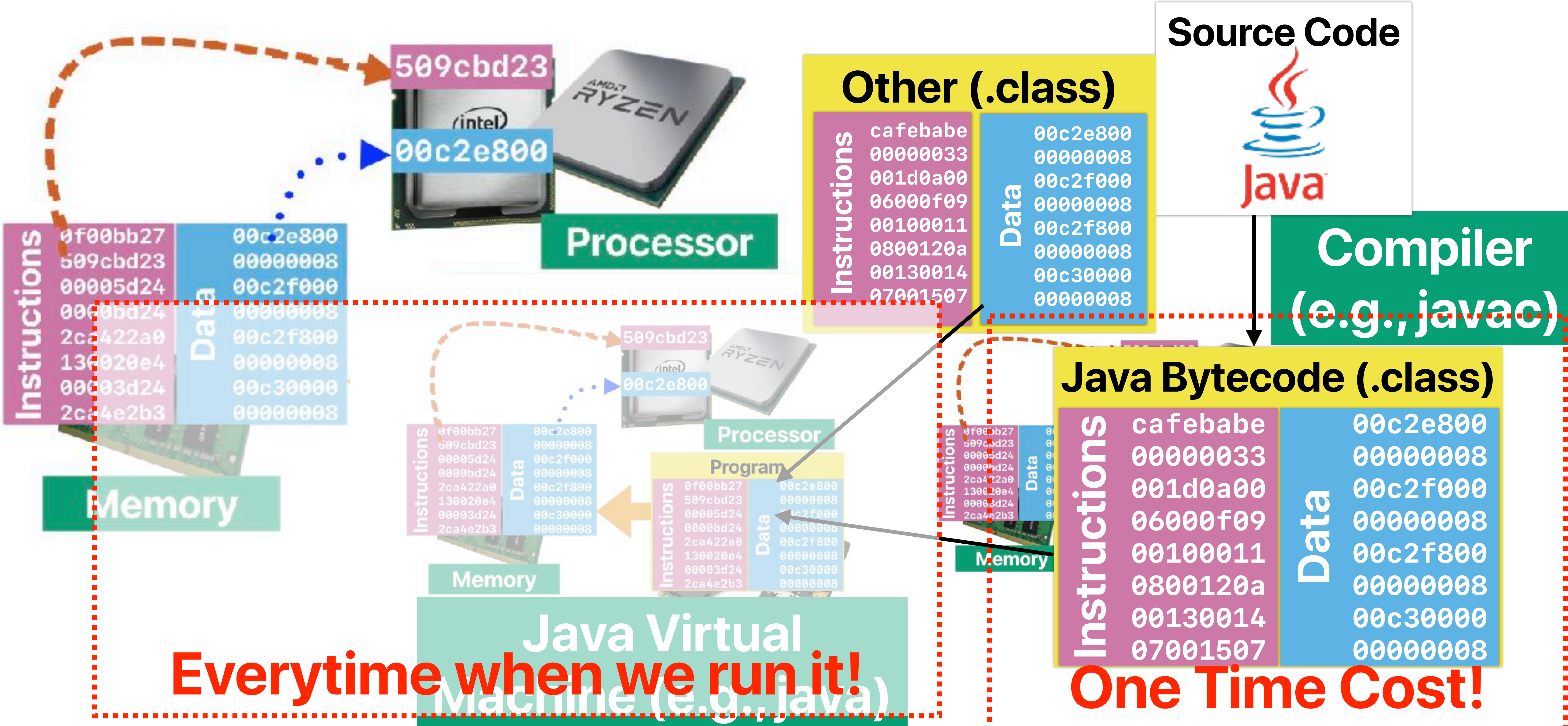
Programming languages

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

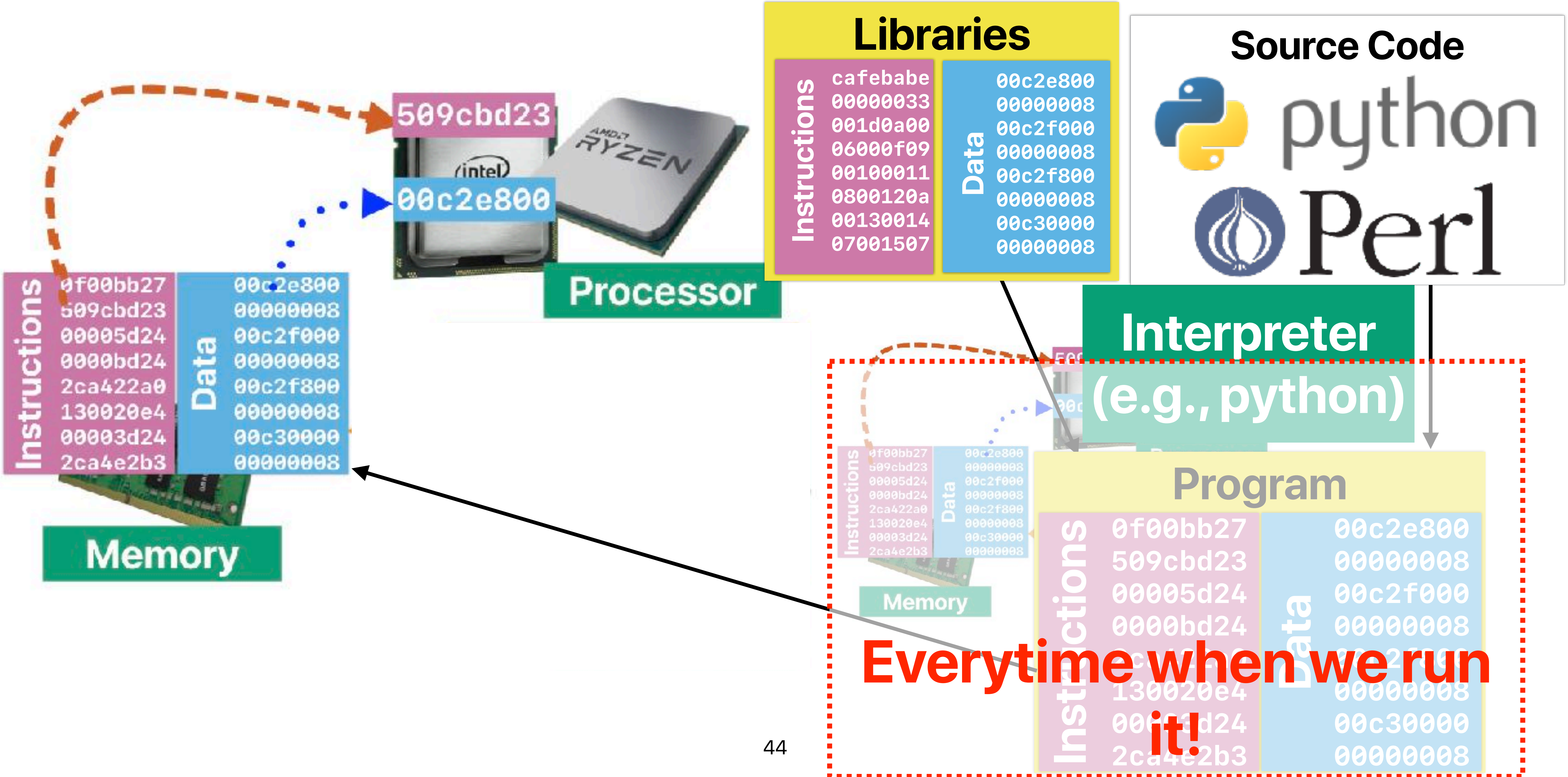
Recap: How my "C code" becomes a "program"



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

Takeaways: What matters?

- Programmers can control all three factors in the classic performance equation
- Different programming languages can generate machine operations with different orders of magnitude performance — programmers need to make wise choice of that!

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

How compilers affect performance

- If we apply compiler optimizations for both code snippets **A** and **B**, how many of the following can we expect?
  ① Compiler optimizations can reduce IC for both
  ② Compiler optimizations can make the CPI lower for both
  ③ Compiler optimizations can make the ET lower for both
  ④ Compiler optimizations can transform code B into code A

A. 0
B. 1
C. 2
D. 3
E. 4

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 compilers affect performance

- If we apply compiler optimizations for both code snippets **A** and **B**, how many of the following can we expect?
  ① Compiler optimizations can reduce IC for both
  Compiler can apply loop unrolling, constant propagation naively to reduce IC
  ② Compiler optimizations can make the CPI lower for both
  Reduced IC does not necessarily mean lower CPI — compiler may pick one longer instruction to replace a few shorter ones
  ③ Compiler optimizations can make the ET lower for both
  Compiler cannot guarantee the combined effects lead to better performance!
  ④ Compiler optimizations can transform code B into code A
  Compiler will not significantly change programmer's code since compiler cannot guarantee if doing that would affect the correctness

A. 0

B. 1

C. 2

D. 3

E. 4

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];
  }
}
```

Takeaways: What matters?

- Programmers can control all three factors in the classic performance equation
- Different programming languages can generate machine operations with different orders of magnitude performance — programmers need to make wise choice of that!
- Compiler optimization can help — only if programmers write code in a way facilitating optimizations!

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**

Takeaways: What matters?

- Programmers can control all three factors in the classic performance equation
- Different programming languages can generate machine operations with different orders of magnitude performance — programmers need to make wise choice of that!
- Compiler optimization can help — only if programmers write code in a way facilitating optimizations!
- Complexity does not provide good assessment on real machines due to the idealized assumptions

Announcement

- Reading quiz due this Thursday before the lecture — we will drop two of your least performing reading quizzes
- Check our website for slides, Gradescope for assignments, piazza for discussions
  - Don't forget to check your access to escalab.org/datahub and piazza
  - check your grades on escalab.org/my_grades
- Youtube channel for lecture recordings: https://www.youtube.com/c/ProfUsagi/playlists

Computer
Science &
Engineering

つづく