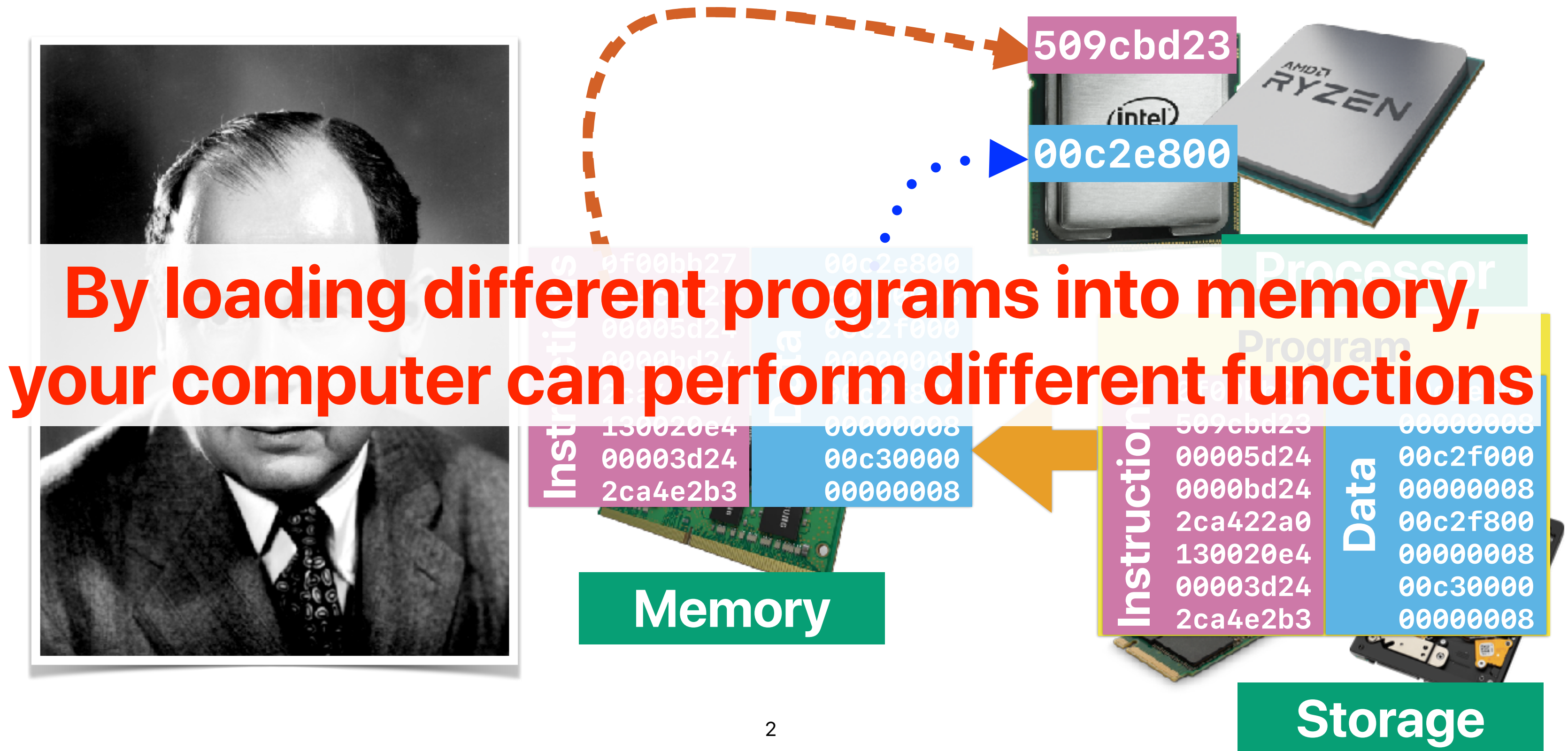


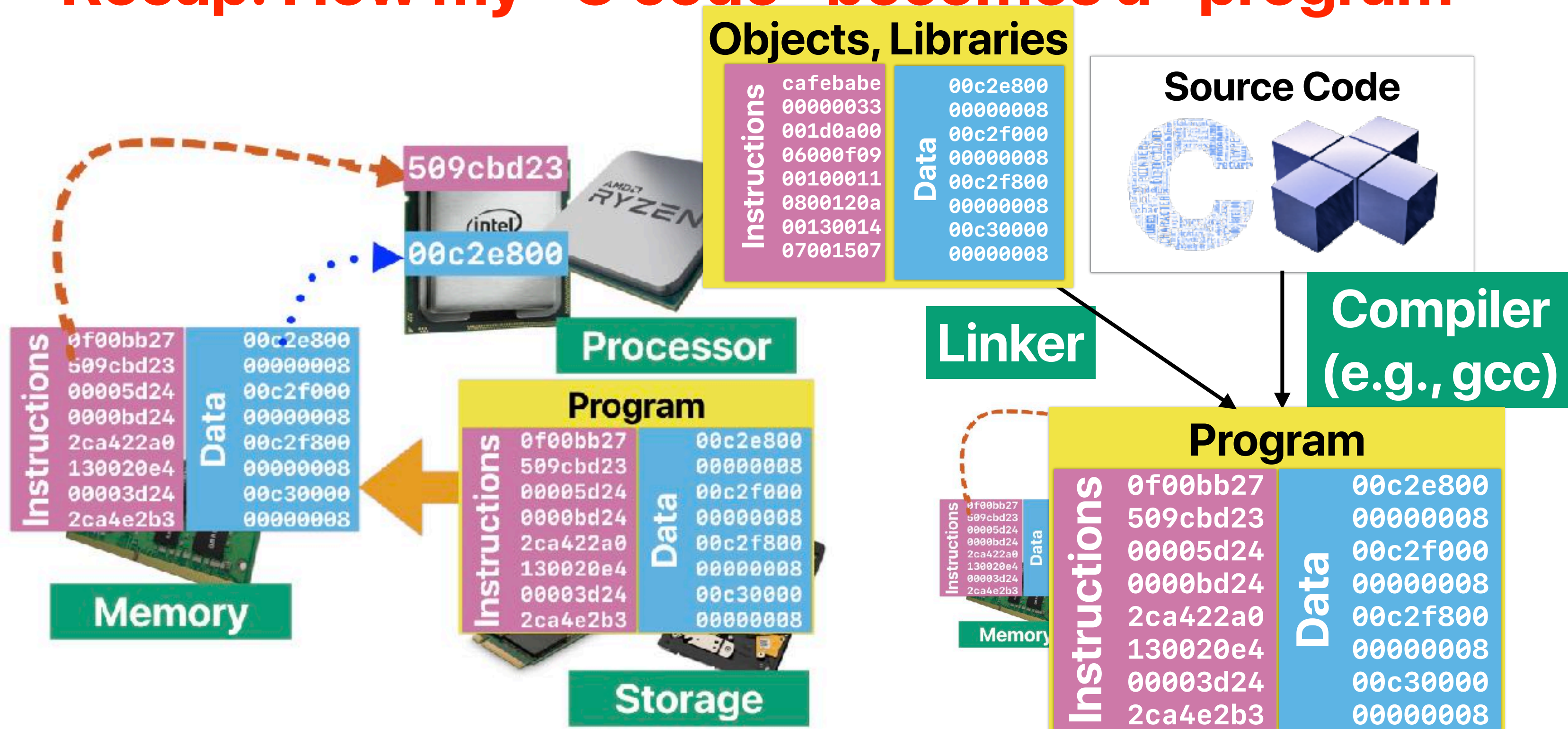
# Performance (I): The Basics

Hung-Wei Tseng

# Recap: von Neumann Architecture

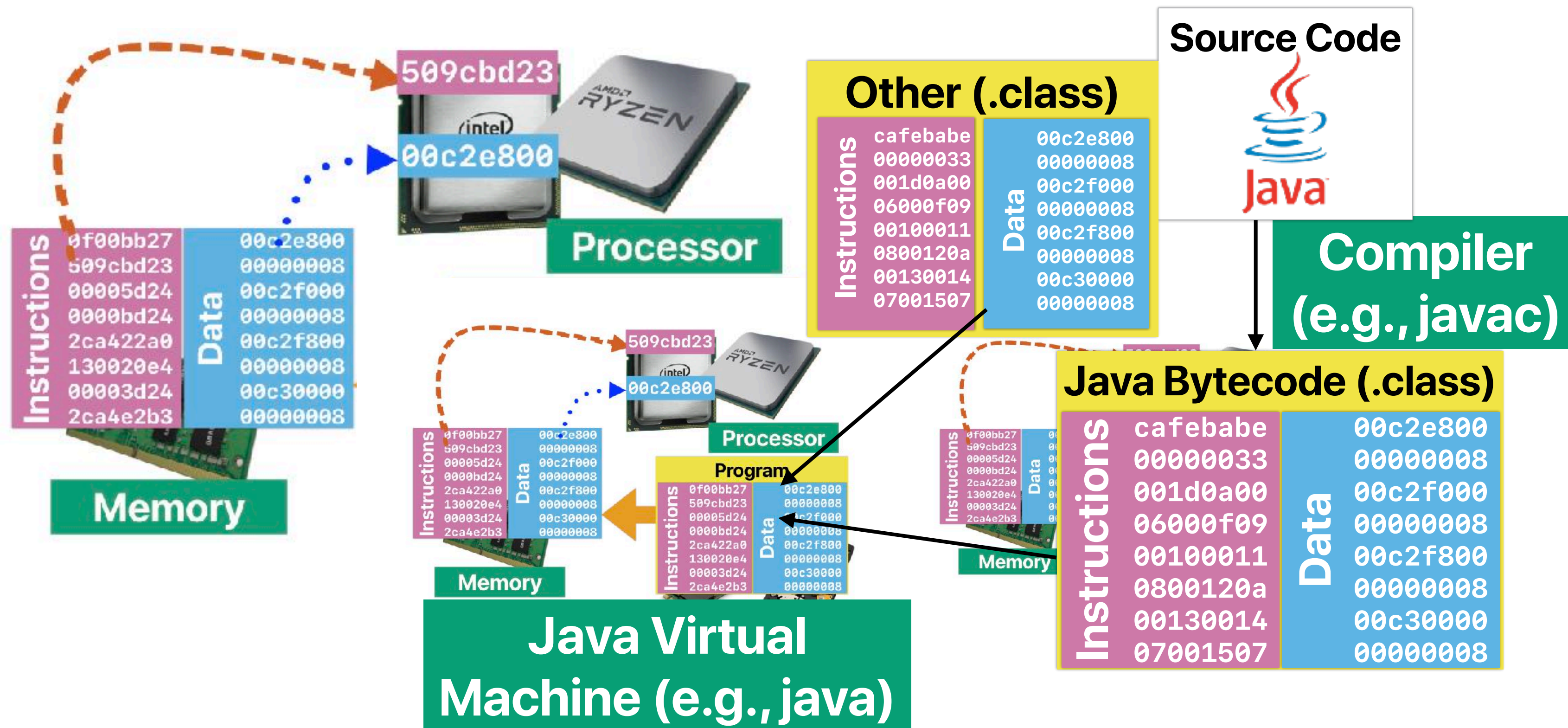


# Recap: How my "C code" becomes a "program"

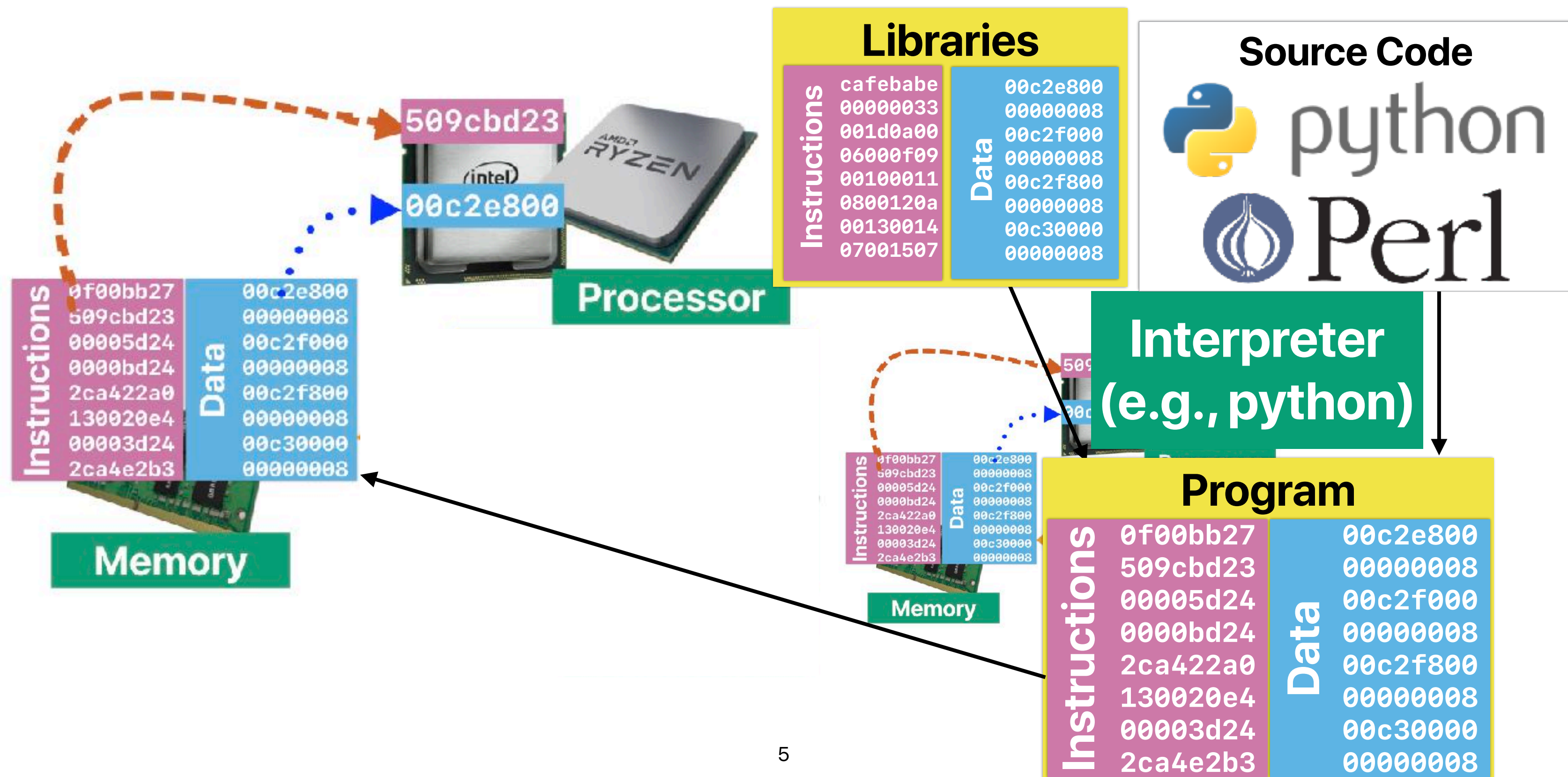




# Recap: How my "Java code" becomes a "program"



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



# Outline

- Definition of "Performance"
- What affects each factor in "Performance Equation"

# Definition of “Performance”



# Peer instruction

- Before the lecture — You need to complete the required **reading**
- During the lecture — I'll bring in activities to ENGAGE you in exploring your understanding of the material
  - Popup questions
  - Individual **thinking** — use polls in Zoom to express your opinion
  - Group **discussion**
    - Breakout rooms based on your residential colleges!
    - Use polls in Zoom to express your group's opinion
  - Whole-classroom **discussion** — we would like to hear from you

**Read**

**Think**

**Discuss**



# CPU Performance Equation (X)

- Assume that we have an application composed with a total of **50000000000** instructions, in which **20%** of them are "Type-A" instructions with an average **CPI of 8** cycles, **20%** of them are "Type-B" instructions with an average **CPI of 4** cycles and **the rest** instructions are "Type-C" instructions with average **CPI of 1** cycle. If the processor runs at **3 GHz**, how long is the execution time?

- A. 3.67 sec
- B. 5 sec
- C. 6.67 sec
- D. 15 sec
- E. 45 sec

**Now, open the png file sent through the chat (you cannot access it after we're in break-out rooms)**

**After entering the break-out room, elect someone as your scribe — who will be responsible for express the thoughts/answers in your group today when your team is selected!**

**Once your group reach a consensus, go back to  
the main lobby and vote!**

**Now — let's try to wrap up everything in 3  
minutes!**



# CPU Performance Equation (X)



- Assume that we have an application composed with a total of **5000000000** instructions, in which **20%** of them are "Type-A" instructions with an average **CPI of 8** cycles, **20%** of them are "Type-B" instructions with an average **CPI of 4** cycles and **the rest** instructions are "Type-C" instructions with average **CPI of 1** cycle. If the processor runs at **3 GHz**, how long is the execution time?
- A. 3.67 sec
  - B. 5 sec
  - C. 6.67 sec
  - D. 15 sec
  - E. 45 sec

# CPU Performance Equation

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

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

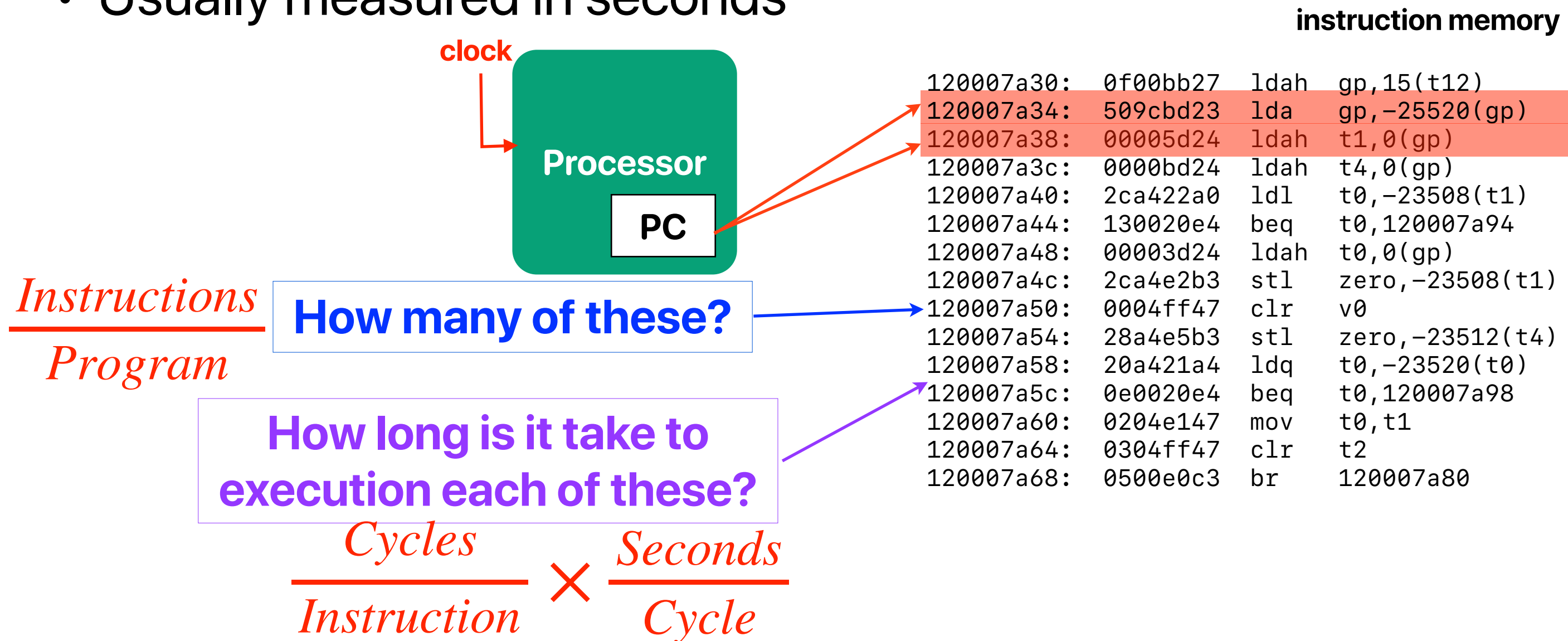
$$ET = IC \times CPI \times CT$$

$$1GHz = 10^9 Hz = \frac{1}{10^9} sec\ per\ cycle = 1\ ns\ per\ cycle$$

*Frequency(i.e., clock rate)*

# Execution Time

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



# Performance Equation (X)

- Assume that we have an application composed with a total of **50000000000** instructions, in which **20%** of them are "Type-A" instructions with an average **CPI of 8** cycles, **20%** of them are "Type-B" instructions with an average **CPI of 4** cycles and **the rest** instructions are "Type-C" instructions with average **CPI of 1** cycle. If the processor runs at **3 GHz**, how long is the execution time?

A. 3.67 sec

B. 5 sec

C. 6.67 sec

D. 15 sec

E. 45 sec

$$ET = (5 \times 10^9) \times (20\% \times 8 + 20\% \times 4 + 60\% \times 1) \times \frac{1}{3 \times 10^{-9}} \text{sec} = 5$$

**average CPI**

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



# Speedup of Y over X

- Consider the same program on the following two machines, X and Y. By how much Y is faster than X?

	Clock Rate	Instructions	Percentage of Type-A	CPI of Type-A	Percentage of Type-B	CPI of Type-B	Percentage of Type-C	CPI of Type-C
Machine X	3 GHz	5000000000	20%	8	20%	4	60%	1
Machine Y	5 GHz	5000000000	20%	13	20%	4	60%	1

- A. 0.2
- B. 0.25
- C. 0.8
- D. 1.25
- E. No changes



# Speedup of Y over X

- Consider the same program on the following two machines, X and Y. By how much Y is faster than X?

	Clock Rate	Instructions	Percentage of Type-A	CPI of Type-A	Percentage of Type-B	CPI of Type-B	Percentage of Type-C	CPI of Type-C
Machine X	3 GHz	5000000000	20%	8	20%	4	60%	1
Machine Y	5 GHz	5000000000	20%	13	20%	4	60%	1

- A. 0.2
- B. 0.25
- C. 0.8
- D. 1.25
- E. No changes

# Speedup

- The relative performance between two machines, X and Y. Y is  $n$  times faster than X

$$n = \frac{\textit{Execution Time}_X}{\textit{Execution Time}_Y}$$

- The speedup of Y over X

$$\textit{Speedup} = \frac{\textit{Execution Time}_X}{\textit{Execution Time}_Y}$$

# Speedup of Y over X

- Consider the same program on the following two machines, X and Y. By how much Y is faster than X?

	Clock Rate	Instructions	Percentage of Type-A	CPI of Type-A	Percentage of Type-B	CPI of Type-B	Percentage of Type-C	CPI of Type-C
Machine X	3 GHz	5000000000	20%	8	20%	4	60%	1
Machine Y	5 GHz	5000000000	20%	13	20%	4	60%	1

A. 0.2  $ET_Y = (5 \times 10^9) \times (20\% \times 13 + 20\% \times 4 + 60\% \times 1) \times \frac{1}{5 \times 10^{-9}} \text{sec} = 4$

B. 0.25  $Speedup = \frac{Execution\ Time_X}{Execution\ Time_Y}$

C. 0.8

D. 1.25

$$= \frac{5}{4} = 1.25$$

E. No changes



# **What Affects Each Factor in Performance Equation**

# 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



# 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

# 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];
    }
}
```

- 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];
    }
}
```

$O(n^2)$

Same

Same

???

Complexity

Instruction Count?

Clock Rate

CPI

$O(n^2)$

Same

Same

???

# 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

# 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



# 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'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

# 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
```

```
To read
```

# 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

# 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



# How programming languages affect perform

- 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

# 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

# Announcement

- 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
- Check our website for slides, iLearn for quizzes/assignments, piazza for discussions
  - Assignment #1 due 10/19
  - Assignments SHOULD BE done individually
  - We will drop your least performing assignment as well
  - Attendance counts as one assignment

# Computer Science & Engineering

# 203

# つづく

