Lecture 1

Entire course is online: <https://learn.bcit.ca/d2l/home/742403>

Jason puts things under the Content section

Quiz every lesson: Activities->Quizzes based on previous week

Homework sometimes: Activities->Assignments

Recorded lectures: Activities->Virtual Classroom

Jason\_Harrison@bcit.ca

How to do well in this course:

* Focus on math and diagrams
* Read ahead of time
* Don’t fall behind
* Ask questions, especially during class

Typical class: Q&A, lesson, quiz

Official grades are at my.bcit.ca

Course outline: <https://www.bcit.ca/outlines/20212057039>

Focus on these diagrams from the textbook:

Chapter 1: pages 5, 18, 50\*

Chapter 2: 57, 66, 75\*, 82

Chapter 3: 161\*, 164, 165, 167, 173, 176, 193, 195

Chapter 4: 245, 292, 307\*\*

Chapter 5: 366

The final exam covers material only after the midterm.

Fast computer: more than one core, fast clock speed cpu, fast frequency RAM, motherboard with lots of threads, high bus speed.

Your computer is the result of a lot of compromises: we want simplicity, inexpensive, backwards compatibility.

John von Neumann. Work with a partner and come back and tell me about JvN.

DEFINITIONS:

Bit 0 or 1 1b = 1 bit

Byte 8-bit word 1B = 1 Byte

Word 64 bits in size probably; it depends on the size of a register

Register cpu memory: by definition it stores one word (the word could be an address, an instruction, a operand for an instruction, etc…)

32 bits versus 64 bits: 2^64 = 2^32 \* 2^32

How much larger is 1000000 compared to 1000? 1000

CPU central processing unit: FDE cycle: fetches, decodes, and executes instructions

Memory main memory aka RAM

Bus common electrical pathway along which data is transferred among devices

2^7 = 128

2^3 = 8

2^10 = 1024 aka 1k

2^17 = 2^7 \* 2^10 = 128k

2^20 = 1M

2^30 = 1G

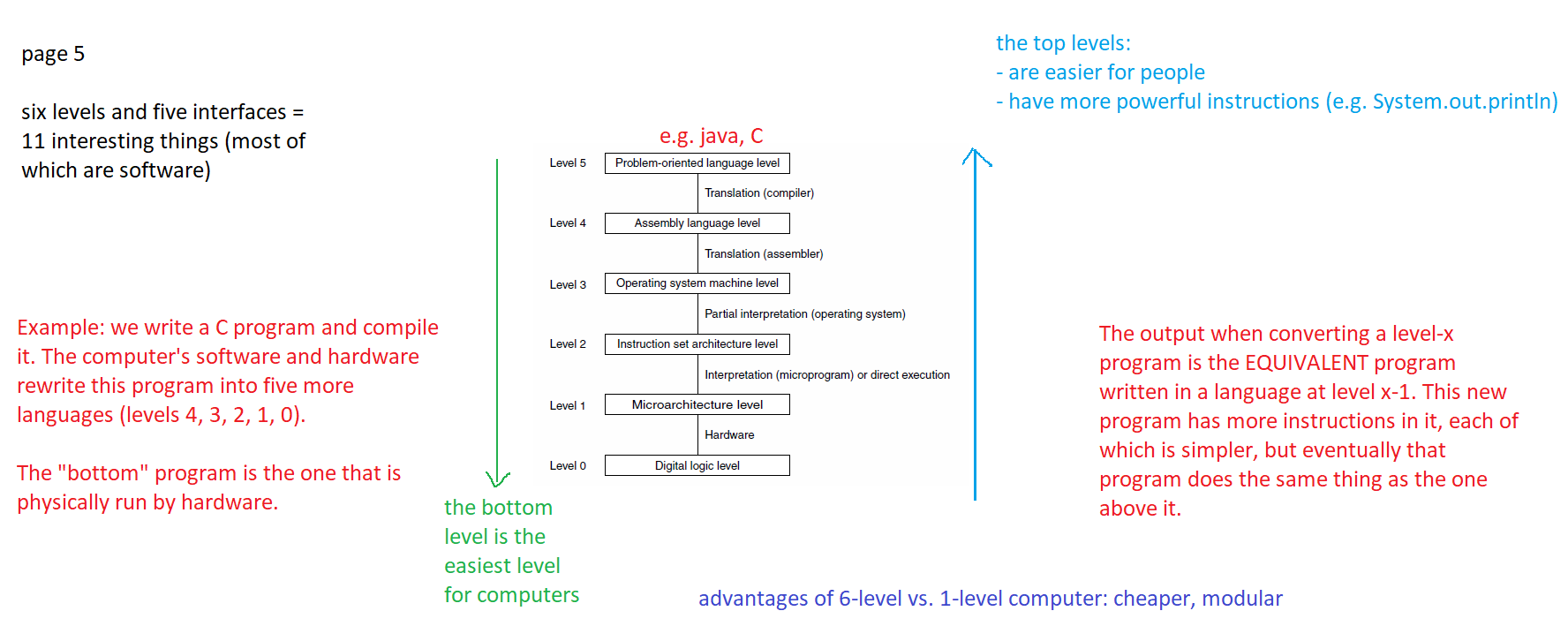
2^40 = 1T

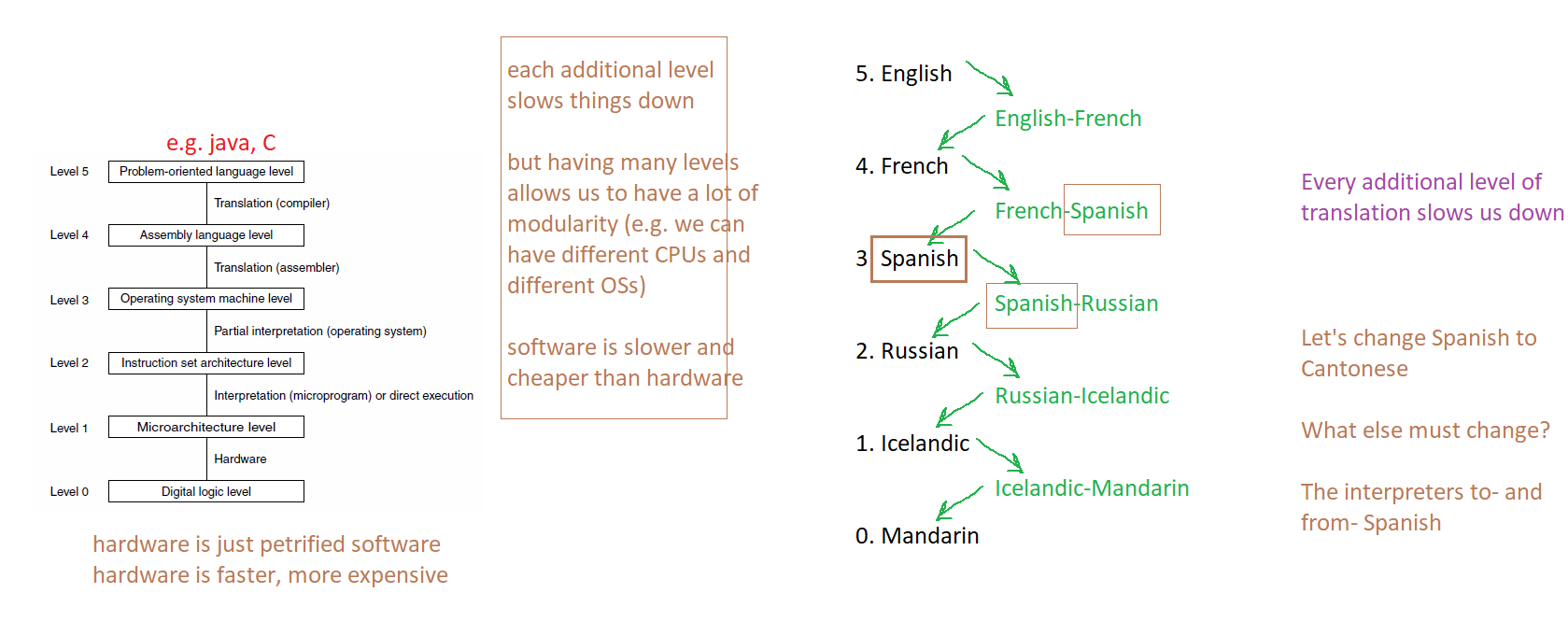
2^32 = 2^2 \* 2^30 = 4G

2^15 = 32k

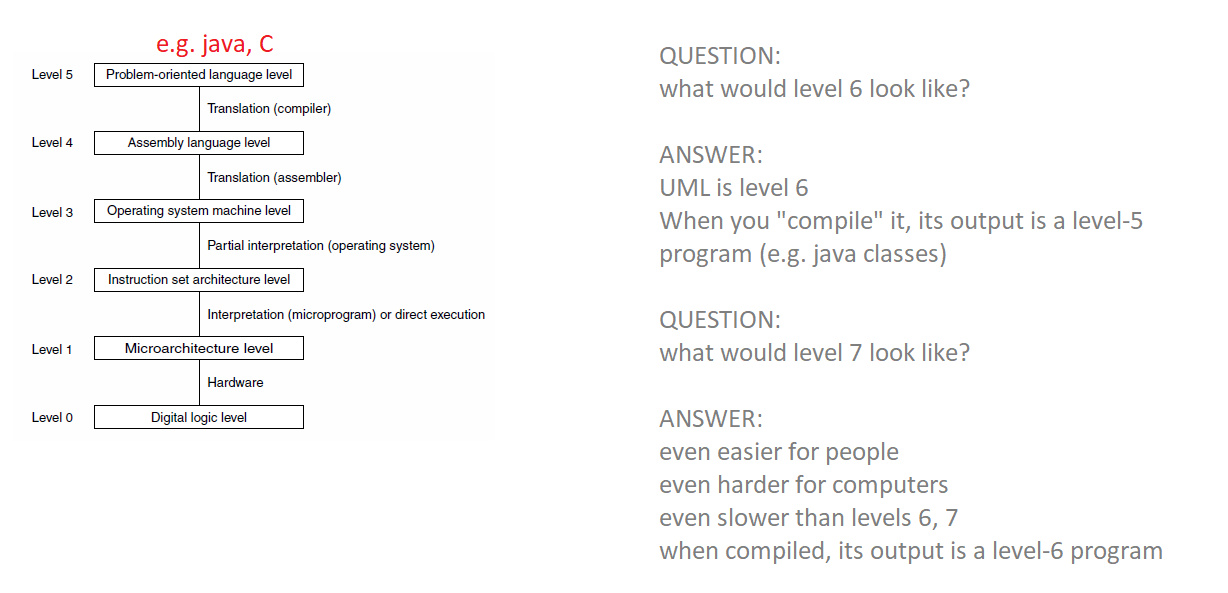
2^29 = 512M

Answer like this on our quizzes and exams



e.g. software vs. hardware:

Chess DVD vs. chess card



**Definitions:**

Translation: aka compilation Converts an entire higher-level-language program into the   
 equivalent lower-level-language program all at once. Eg Write a   
 video game in C, compile it, can throw away the source code  
 and run / keep the executable. Compiling is slow; running the executable is fast.

Interpretation: Converts one instruction from a higher-level-language program into the equivalent lower-level-language instruction(s) and executes it immediately; then repeat with the next instruction E.g. JavaScript program.

Page 52: question 4

