Some slides borrowed from Kurt Eiselt and Steve Wolfman. Some learning goals from Beth Simon.

Readings
Your textbook is Big Java (3rd Ed).
This week’s reading: Chapter 1.

Administrative Stuff
You will need: a Campus-Wide Login (CWL), and an account with the CS department.

You can find instructions for getting all of these on WebCT. This course is on the new Vista server:
http://www.vista.ubc.ca

A good starting point is the course website, which will also get you to WebCT:
http://www.ugrad.cs.ubc.ca/~cs111/

Labs and Tutorials
Labs and tutorials start next Monday.

However, do Lab #0 (on WebCT) on your own, this week.

You must be enrolled in a lab section. If the lab sections you need are full, go to ICICS Room 201 and see a Computer Science undergrad advisor!

Learning Goals
By the end of class today you will be able to...

- Define what “binary representation” means and give some intuition why computers use binary.
- Explain how a computer memory is organized and what a memory address is.
- Describe how a computer program causes the computer to perform a process
- Explain why it’s useful to have both machine/assembly language and high-level computer languages (and how they are different)
- Describe what compilers and interpreters do

Last Time...
- Computer science gets physical artifacts to aid or exhibit thinking!
- Current preferred basic component is the MOS transistor...
Computer Organization

- What do you do with a few billion transistors?
- Some ideas have withstood the test of time:
  - Binary representation of data
  - General-purpose, programmable machine:
    - Memory, accessed by numerical addresses
    - Processor, which executes a sequence of instructions
  - ...

Binary

- “Binary” just means that you have only two symbols, normally written 0 and 1.

Binary Representation of Data

- “Binary” just means that you have only two symbols, normally written 0 and 1.
- You can use binary to represent data in any way that you want:
  - 0 might mean “sun”, 1 might mean “rain”
  - 0 might mean the number 0; 1 might mean 1
  - 0 might mean the number 1; 1 might mean 0!

We can choose what things mean, but we must be consistent.

Binary Representation of Data

- What if you want to represent more than two things? E.g. {sun, rain, snow}
- Must use more than one “bit” to represent your data!
  - 00 = sun
  - 01 = rain
  - 10 = snow
  - (11 = undefined)

Why Binary?

- It’s easier to build a device that has to distinguish between only two values (instead of more values).
- E.g., with the MOS transistor, if the voltage at the gate is high enough, it conducts. If it’s not high enough, it doesn’t.

Binary Representation of Data

- There are standard, widely used encodings for many things (positive integers, signed integers, real numbers, different alphabets, Chinese characters, etc.)
- But in the computer, it’s all just a bunch of bits!
- Does 00 represent “sun”? Or the number 0?
Computer Organization

- What do you do with a few billion transistors?
- Some ideas have withstood the test of time:
  - Binary representation of data
  - General-purpose, programmable machine:
    - Memory, accessed by numerical addresses
    - Processor, which executes a sequence of instructions
  - …

Memory

Memory consists of a series of locations, each having a unique address, that are used to store programs and data.

When data is stored in a memory location, the data that was previously stored there is overwritten and destroyed.

Each memory location stores one byte (or 8 bits) of data. Each bit is a 0 or a 1 (more later).

What does a memory do?

- It sits around and remembers stuff…
- It responds to two kinds of requests:
  - Read: “Please tell me what you have stored in the memory location at address x.”
  - Write: “Please write this data … to the memory location at address x.”
- Who makes those requests?

Core Hardware Components (Highly Simplified)

- Input Devices
- Output Devices
- Central Processing Unit
- Memory
- Mass Storage Devices

The Central Processing Unit

The CPU executes instructions in a continuous cycle known as the “fetch-decode-execute” cycle.

The CPU has dedicated storage locations known as registers. One such register is known as the program counter which stores the address in memory of the next instruction to be executed.
Controlling the Computational Behavior

Because of the fetch-decode-execute cycle, we control the computer to make it do what we want by giving it a sequence of little steps (the instructions) for it to do.

Computer Programming

- Input Devices
- Output Devices
- Central Processing Unit
- Memory
- Mass Storage Devices
- Computer Program

(Old Slide) What's a Computer?

How is a computer different from a video game console? Or a DVD player? Or a telephone? Or a bank machine?

The computer is general. It can be all of the other devices.

Making the computer do what we want is called programming the computer.

Computer Programming

- You can make the computer do anything that it's capable of. The only limits are space, time, I/O devices, and your skill and creativity
- It takes work.
  - The biggest program you'll write in 111 will be a few hundred lines long.
  - Windows Vista is 50 million lines long.
  - You have to write in a language the computer understands.

George and Stephen go to France

- George is American. He knows only English.
- Stephen is Canadian. He is bilingual in English and French.
- How can George communicate in France?

George and Stephen go to France

1. If he wants to communicate quickly, then Stephen can interpret – translating French to English and English to French on-the-fly.
2. If there's a lot of stuff to translate (e.g., a speech, or a long document), then Stephen can translate the whole thing at once. Now, George can read it whenever he wants.
George and Stephen go to France

Translations can be combined:
In the Louvre, they see inscriptions in Egyptian hieroglyphics.

A museum sign gives a French translation.
Stephen interprets the sign for George.
George can understand the hieroglyphics.

Health Education in Remote Areas

In remote areas of the world, there are languages spoken by small groups of people, and also a national language spoken by the mainstream, e.g.:
- Many native languages vs. Spanish in Latin America
- Minority languages vs. Mandarin in China
- Regional languages vs. Hindi or English in India

How do you provide health info to the isolated?

Health Education in Remote Areas

Original Info (English)

Localized Field Manual (Spanish)

English-Spanish Translation

Localized
Interpreter

Local
Interpreter

Nahuatl Speaker

Maya Speaker

Zapoteco Speaker

Machine Language

This is the “native language” of a computer.
Remember: Everything is in binary!
Each instruction does very little.
The computer does them very fast.
Each kind of processor has its own machine language, e.g.:
- x86 (Intel, AMD), Windows and new Apples
- PowerPC (Freescale, IBM), older Apples, Sony PS3
- SPARC (Sun), used in Sun servers
- Many more...

Examples: An instruction to read the content of a memory location:
- x86: movl %eax,(%ebx)
  10001011 11000000 00011011
- SPARC: ld [%r8+0],%r9
  11 01001 00000 01000 1 0000000000000

(Examples not guaranteed to be perfect. It’s very error-prone to do these by hand!)

Assembly Language

Assembly language is just an easier-to-read version of machine language:
- x86: movl %eax,(%ebx)
  10001011 11000000 00011011
- SPARC: ld [%r8+0],%r9
  11 01001 00000 01000 1 0000000000000

(Almost never do humans write machine language. It’s just a nit-picky translation.)
A High-Level Language is a computer language designed to be easier for humans:
- \( a=b+c; \)
- Must be translated into machine language so the computer can understand it.
- Who does the translation? The computer!

Java Does Both!

Interpreters and Compilers

- An interpreter translates the high-level language into machine language on-the-fly, executing the instructions as it goes.
- A compiler translates the high-level language program all at once in advance.
- Both compilers and interpreters are themselves computer programs.
- Which is better? (Remember George and Stephen in France?)

A Simple Java Program

// Our first Java program.
/* Traditionally, one’s first program in a new language prints out “Hello, World!”
*/
class HelloTester {
    public static void main(String[] args) {
        System.out.println("Hello, World!");
    }
}
Controlling the Computational Behavior

A procedure or algorithm is a collection of instructions in some meaningful order that results in useful behaviour on behalf of the device that executes the instructions.

When the instructions are written in a symbolic language that can be executed by a computer, the procedure is called a computer program.

A process is what happens when a computer follows a program - it's a procedure in execution.

Procedures and algorithms

A procedure or algorithm is a collection of instructions in some meaningful order that results in useful behaviour on behalf of the device that executes the instructions.

When the instructions are written in a symbolic language that can be executed by a computer, the procedure is called a computer program.

A process is what happens when a computer follows a program - it’s a procedure in execution.

Procedures and algorithms

Computer people often use the words "procedure" and "algorithm" interchangeably...we will too.

An algorithm is:
- a finite procedure
- written in a fixed symbolic vocabulary
- governed by precise instructions
- moving in discrete steps, 1, 2, 3, ...
- whose execution requires no insight, cleverness, intuition, intelligence, or perspicuity
- and that sooner or later comes to an end

David Berlinski in The Advent of the Algorithm

Here's why we get frustrated when we start to learn to make computers do stuff:

An algorithm is:
- a finite procedure
- written in a fixed symbolic vocabulary
- governed by precise instructions
- moving in discrete steps, 1, 2, 3, ...
- whose execution requires no insight, cleverness, intuition, intelligence, or perspicuity
- and that sooner or later comes to an end

We don't have a lot of practice at being precise!

How to avoid frustration

Practice, Practice, Practice

It takes a lot of practice to learn to be precise enough to make a computer do what you want

It takes a lot of practice to keep from assuming that the computer is smarter than it really is

It takes a lot of practice to get good at this stuff

Questions?