CPSC 111
Introduction to Computation

Lecture 1

Thursday January 5, 2006

Who I Am

Tamara Munzner (call me Tamara!)
tmm@cs.ubc.ca
http://www.cs.ubc.ca/~tmm

ICICS X661
office hours soon to be determined

http://www.ugrad.cs.ubc.ca/~cs111/
http://www.webct.ubc.ca/

What This Course Is About

Calendar description: Basic programming constructs, data types, classes, interfaces, protocols and the design of programs as interacting software components.

Prerequisites: Mathematics 12.

- you must have taken Mathematics 12 before now or you will be dropped from the course.

- see CS advisors if you need prerequisite waived for equivalent work.

Who This Course Is For...

you've used a mouse and keyboard but
no prior programming experience assumed

if you have significant prior programming experience
consider the challenge exam

sign up and pay at dept office by Friday at noon
you'll be contacted with info

see challenge web page for practice questions
http://www.cs.ubc.ca/ugrad/info/planning/challenge111.shtml

...But Note Prerequisite Voodoo

If you have already received credit for CPSC124 and CPSC126 or for CPSC122 and CPSC128 then you cannot receive credit for CPSC111. If you have taken CPSC124 but not CPSC126 then you must take CPSC111 in order to advance through the computer science program. If you have taken CPSC122 but not CPSC128, please consult with a department advisor.

This probably isn't relevant to any of you!

Other Section

One other section of CPSC 111:

```
M W F  15:00 - 16:00  Steve Wolfman    DMP 310
```

- Styles may be different
  - lectures
  - slides
- Same
  - material, speed
  - teaching assistants, labs, tutorials, homework
  - exams

Slide Credits

Lectures based on previous CPSC 111 slides of Kurt Eiselt

Labs and Tutorials

No labs or tutorials this week.

Labs and tutorials begin next week.

Lab room: CICSR/ICICS 008 (in basement)

Reading

Your textbook is Big Java (second edition) by Cay Horstmann (Wiley and Sons).

You should get a copy. Seriously.

Read before class (except today).

Read chapter 1 for next time.

Weekly question: turn in Thursdays, start of class

Exam Dates

Midterm exam 1: Tuesday, February 7
6:30 to 8:00pm

Midterm exam 2: Thursday, March 16
6:30 to 8:00pm

Final exam: We don't know yet

Grading Scheme

Tentative grade calculation

10 labs	5%
4 assignments	15%
2 midterm exams	30%
Final exam	50%

Please note that in order to pass the course you must:

- obtain an overall grade of at least 50%
- obtain a grade of at least 50% on the final exam
- obtain an overall grade of at least 50% on the combined lab and assignment grades

If you fail to satisfy any of the above criteria, a grade no greater than 45% will be assigned in the course. The instructor reserves the right to modify this grading scheme as necessary throughout the term.

Assignment 0

Tell us about yourself

Fill out and return Tuesday Jan 10 in class

Have you hugged a computer today?

What computers have you interacted with recently?

What kinds of digital technology are part of the wallpaper of your life?

This is a first course in computer science...

...but what is computer science?

This is a first course in computer science...

...but what is computer science?

"Computer science is as much about computers as astronomy is about telescopes."

Edsger Dijkstra

This is a first course in computer science...

...but what is computer science?

"Computer science revolves around computational processes.... A process is a dynamic succession of events.... When your computer is busy doing something, a process is going on inside it."

Oliver Grillmeyer

This is a first course in computer science...

...but what is computer science?

"Computer science is the study of what computers do, not of what they are."

Kurt Eiselt, UBC

Processes, procedures, and programs

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

Processes, procedures, and programs

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

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

Processes, procedures, and programs

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

A procedure is a collection of instructions in some meaningful order that results in useful behavior 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.

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

Procedures and algorithms

Here's why we get frustrated when we start to learn to write programs 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!

Procedures and algorithms

Here's why we get frustrated when we start to learn to write programs 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 stupid!

How to avoid frustration

Practice, Practice, Practice

This material isn't conceptually incomprehensible, but...

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

Don’t wait until the last minute to get help

Bad things happen while learning a new skill. Start homework early; give yourself time for mistakes.

Don’t be too ambitious with your course load. You can’t slack off in this class, even for a few days (hours?).

Thinking in terms of process is crucial

Formulas aren't sufficient for describing how our world works. For example,

- Economic systems are processes
- Political systems are processes
- How HIV invades cells is a process
- How pharmaceuticals will interfere with HIV will also be a process

Being able to think about complex systems in terms of procedures and processes will be of value to you even if you never write another program after 111.

So what will you learn here?

How to get a computer to do your bidding:

- How to represent solutions to problems as procedures or algorithms
- How to represent those procedures as programs written in a programming language
- How to get the computer to turn your programs into processes that do useful stuff

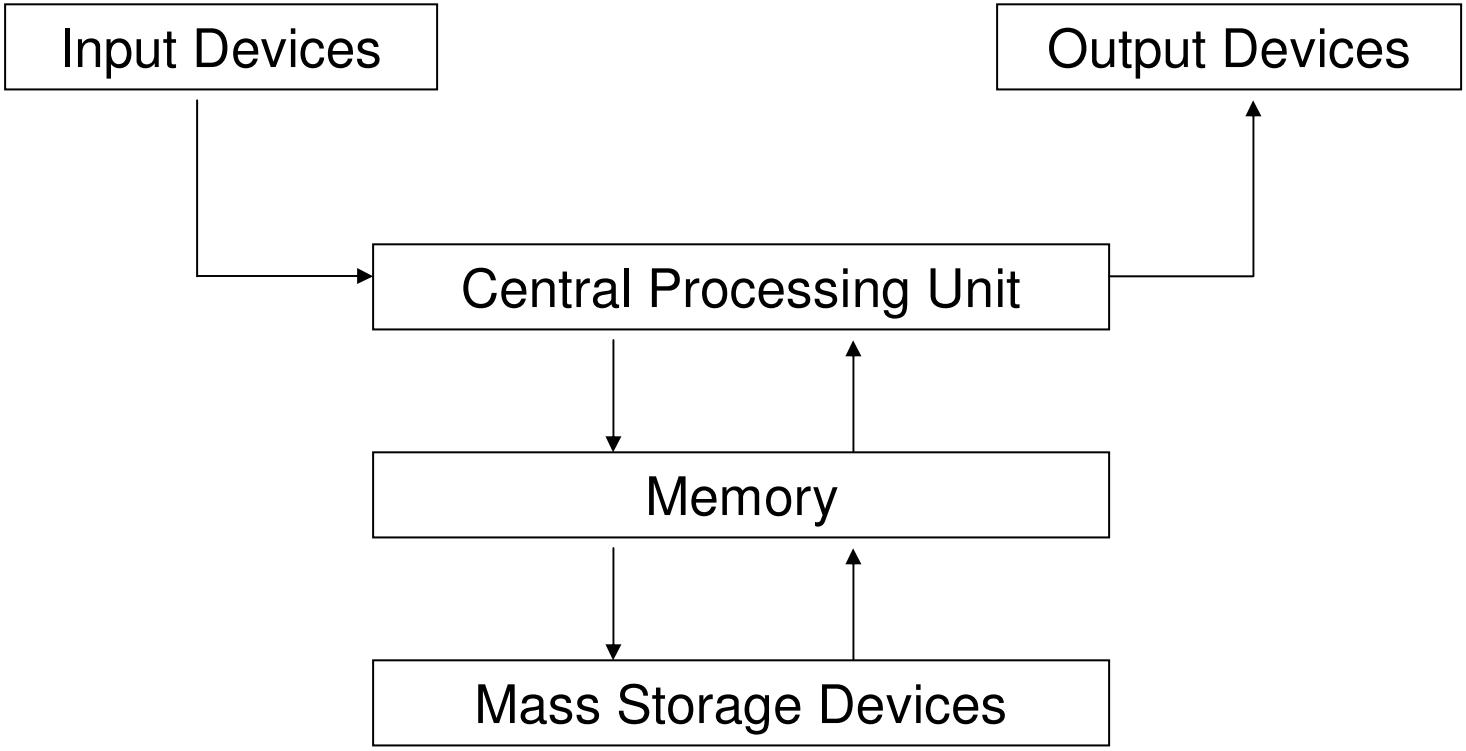
What about the computer itself?

Does this:

"Computer science is the study of what computers do, not of what they are."

mean we're not going to talk about the computer and how it works? No, we need to know something about this too, but it won't be the main emphasis of this course.

Computer hardware overview

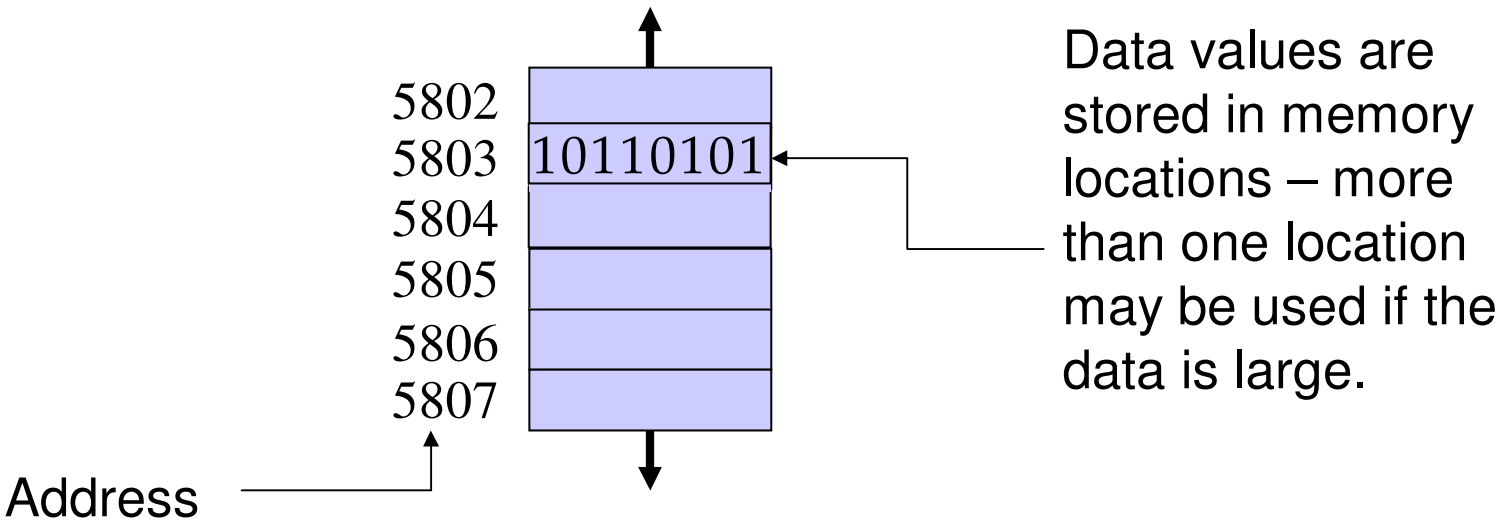


Memory

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).



Units of memory storage

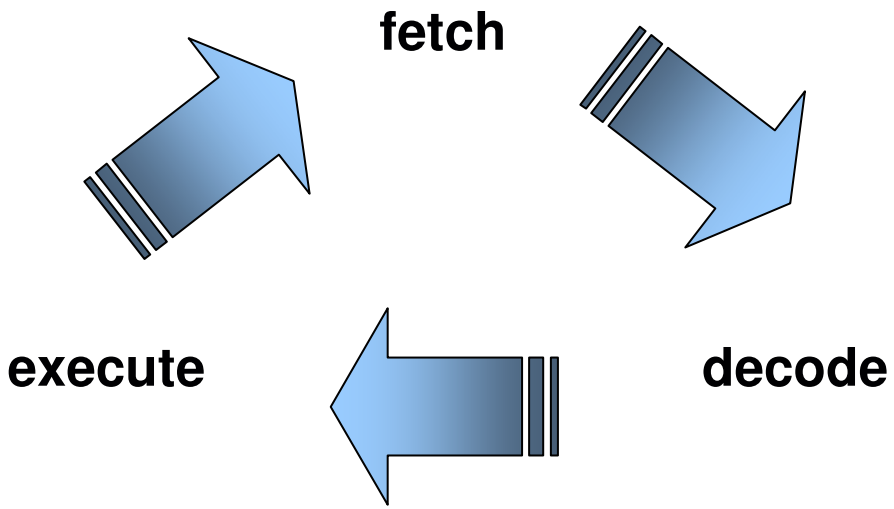
We measure units of memory in terms of bytes:

Unit	Symbol	# of bytes
byte (8 bits)		$2^0 = 1$
kilobyte	KB	$2^{10} = 1024$
megabyte	MB	$2^{20} = 1024^2$
gigabyte	GB	$2^{30} = 1024^3$
terabyte	TB	$2^{40} = 1024^4$

The central processing unit

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

The CPU has dedicated memory 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.



Binary representation

Programs and data are represented using a very simple binary coding scheme where the only symbols used are 0 and 1.

This simple binary system goes hand-in-hand with the digital hardware devices talked about earlier:

- Magnetic disks: magnetic material can be polarized to one of two extremes (north or south) to represent a 0 or a 1.
- Memory: each byte consists of 8 bits; each bit can be thought of as an electronic switch that is either off or on representing a 0 or a 1.
- CD-ROMS: the surface consists of smooth areas and pits representing 0's and 1's respectively.

Let’s see if you already know some geeky computer stuff...

What’s a bit?

Are bits really ones and zeros?

Why binary? Why not ternary? Why not decimal?

How many things can you encode with 1 bit? 2 bits? n bits?

What’s a byte?

What’s magic about 8 bits? Why not 9 or 10?

What can be represented by a byte?

256 different characters from your keyboard
(Java actually uses 2 bytes to represent a character... how many characters is that?)

256 different shades of gray in a black and white image

256 colors or shades of color in a color image

256 frequencies or tones to be played through a speaker

256 of anything...

Questions?