University of British Columbia
CPSC 111, Intro to Computation
2009W2: Jan-Apr 2010

Tamara Munzner

Introduction

Lecture 1, Mon Jan 4 2010

based on slides by Kurt Eiselt

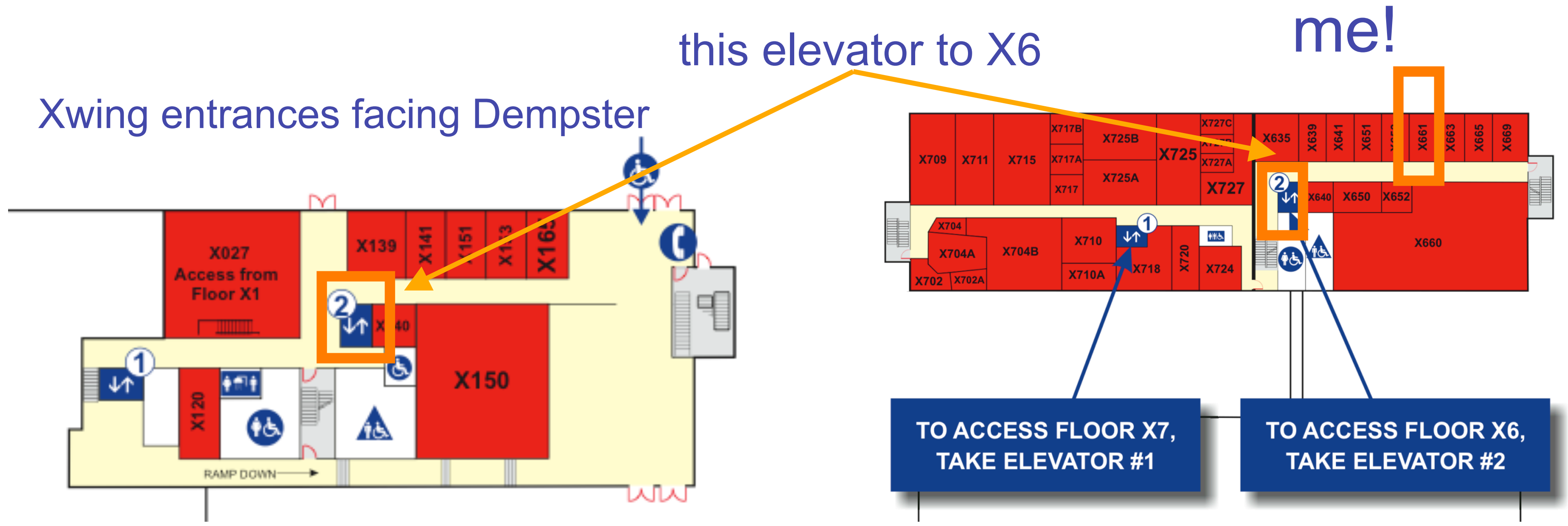
http://www.cs.ubc.ca/~tmm/courses/111-10

News

- no class this Friday Jan 8!

Who I Am

- Tamara Munzner
  - fine to call me either Tamara or Prof. Munzner
  - tmm@cs.ubc.ca, http://people.cs.ubc.ca/~tmm
- office location is X661 (tall wing of ICICS/CS bldg)
  - stay tuned for office hour time announcement!



What This Course Is About

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

Reality: Ignore the buzzwords for now. You're going to learn about computers and how to put together sequences of instructions to make them do useful stuff.

Prerequisites

- Mathematics 12 is the prerequisite
  - if you have not taken it you will be dropped from the course
  - see CS advisors if you need prerequisite waived because of equivalent work
- current stuff
  - you cannot get credit for both 111 and new 110 course
  - you cannot get credit for both 111 and 101
- old stuff
  - see course page for details if you took 122/124/126/128

Who This Course Is For...

- people who do not necessarily have any prior programming experience
  - you can succeed in this course if you have never ever written a computer program!
  - but we do assume you’ve probably used a mouse and keyboard...

Who This Course Is Not For...

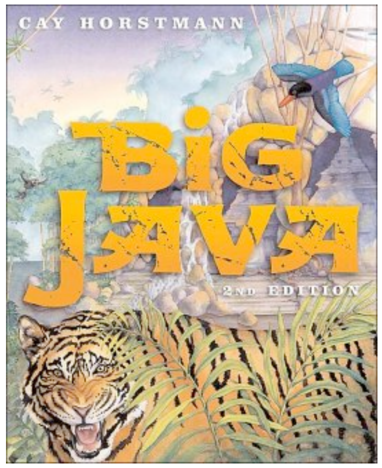
- people with significant prior programming experience
- if this is you, consider the challenge exam
  - sign up and pay at dept office (rm 201) by Friday at noon
  - you'll be contacted with further info
- see challenge page for practice questions
  - http://www.cs.ubc.ca/ugrad/info/planning/challenge111.shtml

Labs and Tutorials

- Labs
  - This week’s lab is take-home: do Lab 0 on your own
    - Link on WebCT Vista: http://www.vista.ubc.ca
  - In-person labs begin next week
    - In room 008 (ICICS/CS basement)
  - Labs are part of your grade. You must be enrolled in a lab. Don't skip labs. Each year, some students skip the labs and are surprised to find they have failed the course. Don’t let this be you!
- Tutorials
  - Start next week
  - Tutorials aren't part of your grade, but they're great educational opportunities. You should go.

Reading

- Textbook is Big Java by Cay Horstmann (Wiley and Sons)
  - either third edition or second edition is OK
- You should get a copy. Seriously.
- Read before class (except today).
- Weekly reading questions
  - turn in Fridays, start of class
  - starting next week
  - see weeklies web page
- This week's reading:
  - 1.1, 1.2



Exam Dates

- Midterm 1: Monday Feb 8, 6:30-8:00pm

- Midterm 2: Monday Mar 22, 6:30-8:00pm

- Final: we don’t know yet
  - don’t make travel plans until posted

Grading Scheme

Tentative scheme (I reserve the right to modify during the term):

10 labs	10%
4 assignments	20%
2 midterm exams	30%
Final exam	40%

All weekly reading assignments combined count as one assignment.
Lab 0 marks include several surveys to be taken over the term.

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.

Policies: Collaboration

- Exams must be done alone
- Labs and assignments may be done alone or in pairs. For pairs, turn in one assignment for the pair.
    - Collaboration is not just copying somebody else's work! Hints on how to succeed with pair programming:

      http://www.ugrad.cs.ubc.ca/~cs111/2009w1/pair.html

- More on plagiarism in the labs
    - Summary: don't do it. You'll probably get caught. It's not worth it. When in doubt, ask the instructor.

WebCT and Lab 0

- WebCT
  - On-line learning tool for labs, sample exams, discussions
  - http://www.vista.ubc.ca
  - Use your CWL id/password to log in
    - Same as you use for UBC wireless
- Use the WebCT discussion boards for questions on course material
  - Please read them regularly and use them instead of sending direct email to instructors and TAs
- To do this week on your own time:
  - Lab 0 (in the Labs folder)
  - Survey (also in Labs folder), counts as part of Lab 0 mark.
  - More surveys to come over the term, will also be part of Lab 0 mark.
- You'll find out more about WebCT in Lab 1

Adminstrative Stuff

- lecture slides will be posted before class (usually)
  - you can follow along and think and make personal notes (instead of scribbling everything frantically)
  - http://people.cs.ubc.ca/~tmm/courses/111-10
- you’ll also need a UBC CS undergrad account
  - very important to read or forward the email
  - more on this in Labs 0, 1
- UBC CS dept announcements

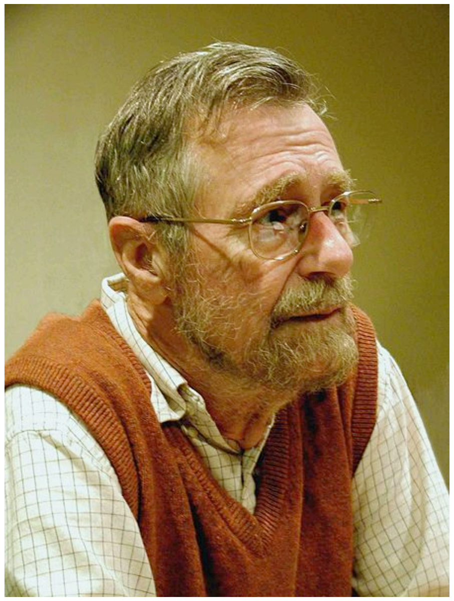
Course Admin Questions?

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



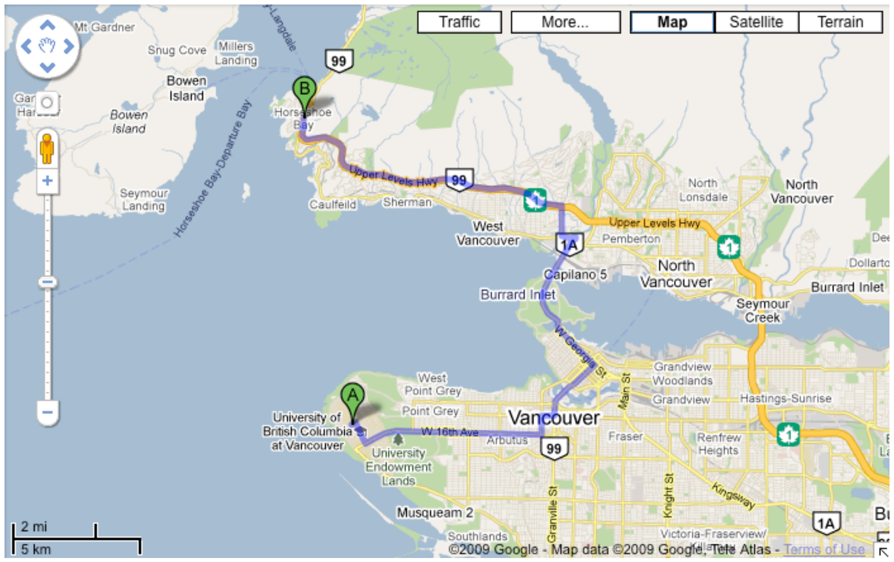
http://upload.wikimedia.org/wikipedia/commons/thumb/d/d9/Edsger_Wybe_Dijkstra.jpg/450px-Edsger_Wybe_Dijkstra.jpg

This is a first course in computer science...

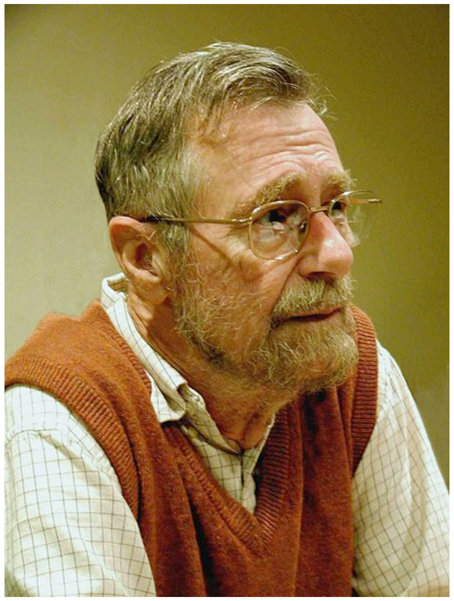
...but what is computer science?

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

Edsger Dijkstra



Dijkstra's shortest-path algorithm in operation



http://upload.wikimedia.org/wikipedia/commons/thumb/d/d9/Edsger_Wybe_Dijkstra.jpg/450px-Edsger_Wybe_Dijkstra.jpg

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

Tip #2

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

Tip #3

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

Questions?