Connecting with Computer Science

CPSC 101/GRSJ 201
Dr. Rachel Pottinger
Course website:
http://www.ugrad.cs.ubc.ca/~cs101
Welcome to Connecting to Computer Science!

I’m Dr. Rachel Pottinger

- This is my 10th year teaching at UBC
- I have a PhD in computer science from the University of Washington
- My research area is databases
- My husband, Steve Wolfman, is another faculty member in the department
- I’m taking martial arts lessons, partially so I know what it’s like to be learning something.
Let’s find out a little something about you
Welcome to Connecting to Computer Science!

- What *is* a computer anyway?
- Get together with two or three neighbours, introduce yourselves, and discuss!
- Use all the cards we distributed
- For *all* exercises, write the names and student IDs of each student participating on one sheet of paper (Only groups of 2-4 students will receive credit!)
Exercise: What is a Computer?

- Divide your cards into two stacks: “computer” and “not a computer”.
- On the piece of paper given, write a clear 1-2 sentence definition of a computer (i.e., the definition that you used to create your stacks); this is your conceptual model.
- Pick an interesting card from each stack and write down your justification of its placement based on your definition.

If you only have one stack in the end, justify the lack of the other stack in place of your second card’s explanation.
How do you vote?

- Turn on the clicker by pressing the “On/Off” button (the blue power light will flash)
- Change the clicker frequency to AB
  - Press the On/Off button; the power light will flash
  - Enter the code “AB”; the green “vote status” light will flash.
- When I ask a question (and start the timer), select A, B, C, D, or E as your vote.
- Check your “Vote Status” Light:
  - Green light = your vote was sent AND received.
  - Red flashing light = you need to vote again.
**Not sure you saw the light? vote again.**
**Want to change your vote? You can vote again as long as the timer is still going.**
Clicker grades for today don’t count
Analytical Engine

Designed ca. 1832 by Charles Babbage

- A machine which would have received instructions from punch-cards—pieces of paper with holes punched in them in special patterns.
- The machine would perform calculations described by the holes using steam-powered gears and cogs. Results of previous calculations could control what calculations it performed next. Finally, it would print its results.

- A → Computer
- B → Not a computer
ENIAC

Developed in 1944 by Mauchly, Eckert et al.

- An electronic device which used vacuum tubes, electrical relays and other components to perform calculations.
- Operators wired together some of ENIAC’s electrical devices to represent the commands it should execute. It then executed those commands, performing literally thousands of additions in a single second. It could not decide what commands to execute next based on the results of previous calculations.
- A → Computer
- B → Not a computer
The Human Brain

Developer/development year disputed; ca. 100 000 BC?

• The human brain is a large organ situated in the skull of the modern human. Capable of a wide range of tasks from mathematical calculations to identifying Jelly Bean flavors, the human brain coordinates most functions of its associated human. It receives sensory information from many devices (eyes, ears, toes, etc.) and also controls devices that act in the world (hands, voice box, toes, etc.).

• The structure of the human brain consists of a massive network of interconnected neurons interacting through chemical and electric processes.

• A → Computer
• B → Not a computer
Course definition of a computer

A device that receives a list of instructions (drawn from a well-defined set of possible instructions) and interprets them to perform some process in the world, such as physical activity or transformation of information.

This definition isn’t the “right” one. It’s just a “useful” one.

We’ll discuss other definitions for what a computer is later in the course and revisit some these devices.
So that’s what a computer is. But why should we study them?

- They’re everywhere
  - Designing cars
  - Factory workers
  - DJs
And they’re also *fun*

It’s so much more fun when *you* can make the computer do what *you* want it to do than what other people want it to do!
And you can understand them (and what they do) better

Have you ever had the computer do something really crazy? It drives you nuts and you have no idea why it’s doing that? It’s much more fun when you can figure out what’s going on.
Plus, it’s even more fun when you combine computers with other things

Let’s face it. How many people out there are really excited about “the computer”? Most of us are more interested in what it can do for us! Like…
Data mining

- Ever wonder what those loyalty cards are for?
- Companies use them to see what people buy in order to increase sales
- Case study: Target decided to market to pregnant women

http://www.nytimes.com/2012/02/19/magazine/shopping-habits.html?pagewanted=1&_r=2&hp&
Want a more “positive” data mining example?

“Disease detectives in British Columbia were able to spot and halt an outbreak of hepatitis A infections in that province in early 2012 by comparing the foods the infected people had bought in previous months” using data mining and grocery store loyalty cards.

Frank Gehry, architect

“… if you figure out a way to make technology work for you, you can explore curved shapes and make them possible at competitive costs. You can do this because of the computer.”
How will we connect with computer science?
Image representation

Different methods for image representation

Should we store all of images using the same method?

A discussion of computer art & animation
Minds and Machines

• Can Machines Think?
• Artificial Intelligence
• Machine language and how computers work
• Logic Circuits
Digital DNA

- describe how the genetic code represents data digitally
- describe how computing enables advances in molecular biology and the health sciences
Diversity in computer science

- As we will cover, diversity is important for many reasons.
- Computer science has a generally non-diverse population.
- We’ll discuss some of the results and possible reasons for this.
So how are we going to do that?

• The best way to learn is by doing, so you’ll be doing a lot of “doing” in this course
• You’ll have a group project (of your own choosing)
• You’ll want to do some exercises
• Labs are time for you to
  • Get hands on practice with a computer
  • Work on your project
• “Lectures” will involve:
  • Doing exercises
  • Talking about questions you have about the reading
  • My explaining concepts
What does this mean for you?

• You need to do the reading ahead of time
  • You’ll be asked to do pre-lecture quizzes, including “reading questions” for new reading topics

• You need to come to class prepared to do things
  • Bring paper and a writing implement
  • Bring an iClicker
  • If you have a laptop computer, you’ll probably want to bring it — but no goofing off
No, really, I mean it: no goofing off on your computer in class.

- On the one hand, if you don’t want to learn and you only hurt yourself, that’s your problem.
- On the other hand, you’re not hurting only yourself.
- Researchers did a set of experiments where they asked students to multitask:
  - “Students … who were asked to multitask averaged 11 per cent lower on their quiz.”
  - “Students … who were surrounded by laptops scored 17 per cent lower on their quiz.”

Great! So what do you need to thrive in this course?

• We’ll be using something called Just in Time Teaching – *readings will be posted on the class website*

• There will be pre-lecture quizzes to make sure you’re up to speed. These will include a reading question. You may do the quiz as many times as you like – we will only grade the last reading question, though.

• I’ll use responses to the reading questions as a “starting point”.

• This means that what *you* are interested in drives the class

• It also means you have to do the reading or you won’t get much out of the class.

• The first optional quiz is due on Thursday at noon. (the rest are not optional)

• Readings will be posted at least a week ahead of time
But there will be other things for you to do, too

- active class participation (5%)
- Pre-lecture quizzes (5%)
- laboratory assignments (10%) – no lab this week
- the course project (20%)
- midterm exams (20%*) – in class Sep 26 & Nov 5.
- and a final exam (40%) – date TBD.

Those are the graded things. You’ll want to practice on your own, too; we’ll post practice questions.
Reading Question (part of the pre-lecture quiz)

- brief question generated while reading the assigned material in preparation for class.
- They are due by noon the day before lecture.
  - 0 marks: for not participating or submitting a frivolous question
  - 1 mark: for any question(s) that shows real thought and effort
  - 2 marks: for an outstanding question(s) (all questions selected for discussion in class earn 2 marks)
Class participation (5%)

• attendance and effort to participate in in-class activities.
• clicker responses
• in-class exercises
• discussion board postings
• TA and instructor evaluations of individual students' participation
Lab assignments (10%)

- Hands on experience on topics covered in class
- They are due at the end of your lab
- They will be marked at the end of the lab
- Released on Fridays for you to review during weekend
- May require reading prior to lab
Course Project (20%)

• Very exciting part of the course – you get to do something that reflects your interests!
• It’s a group project, so keep an eye out in lab
Exams

- In-class midterm exams (20%*)
  - Friday, September 26
  - Wednesday, November 5
- Final exam (40%)

- Passing Criteria:
  - obtain at least a 50% overall mark
  - 50% or greater on the course project
  - 50% or greater on the final examination
Policies

• University policy and departmental guidelines on academic conduct will be followed strictly.

• Lateness and Make-Up Policies: Assignment deadlines will be strictly and automatically observed.

• Scoring schemes for quizzes and participation have been designed so that it is possible to achieve a full aggregate score even if something unexpected happens.
Where to find information

- Start with the course website (http://www.ugrad.cs.ubc.ca/~cs101); it has links to everything else you need
- Discussions will be held on Piazza:
  - We’ll cover how to sign up in lab
  - You are responsible for checking in at least once every day or two.
  - Please direct all course-material questions to the discussion threads, not to email – you will get help much more quickly that way – both from course staff and from other students.
Email & Discussion Etiquette

- Only send email if you can't post the material on Piazza
  - personal issues, sample solution code

- Use meaningful subject headings
  - **OK**: “HTML question – my text vanished!”
  - **NOT OK**: “HELP IM STUCK :( :( “
Other Information

- make sure to follow up on To-Do’s listed on the web page and read announcements
- check the web page to learn about course staff, labs, exams and more

- **Acknowledgement**: Thanks to Holger Hoos, Meghan Allen, Anne Condon, Hassan Khosravi, Joseph Luk, Tristram Southey, Dave Tompkins, Kimberly Voll, Steve Wolfman and many others for valuable contributions to the slides and materials throughout.
Learning Goals for the Course

• **understand** how data structures, interfaces, and processes on computers are designed;

• **design** your own digital artifacts using computer applications and programs, by applying your understanding of data, interfaces, and processes using other resources available to you;

• **connect** your computing knowledge with your knowledge and interest in other disciplines
Learning Goals for Today

• understand the general outline of the course
• become comfortable with articulating and refining conceptual models in learning technical concepts; specifically

  categorize a well-described device as a computer or non-computer with clear, articulate reference to your own explicit definition and our course’s definition
Meta point: what is a learning goal?

- **Meta-meta point: what is a meta point?**
  - Meta: Pertaining to a level above or beyond. For example, *metadata* is data that describes data, *metalanguage* is language that describes language, etc. ([wiktionary](https://en.wiktionary.org/wiki/metadata))

- A learning goal is designed to tell you what information you should learn about a topic
- At the beginning of the topic, the learning goal may not even make any sense
- By the end of a topic, it should help you know what to study