Administrative notes
January 9, 2018

• Survey: https://survey.ubc.ca/s/cpsc-100-student-experience-pre-2017w2/
  Worth bonus 1% on final course mark

• We’ll be using iClickers today
  • If you want to try REEF/iClicker mobile, use it today

• My office hours have been set (for both CPSC 100 and CPSC 103):
  • Wednesdays 1PM - 3PM or by appointment
  • Thursdays 1:30PM - 3PM
  • Friday 3PM - 4PM (starting Jan 12)

• For more TA office hours, check the Contacts page

Compositional Thinking
http://www.ugrad.cs.ubc.ca/~cs100
Labs start this week

Check the Labs page on the webpage if you are interested in seeing what you are going to be doing
Computer Basics and Algorithms
Learning Goals

- CT Building Block: Students will be able to define the difference between hardware, applications, and the operating system.
- CT Building Block: Students will be able to give examples of a problem for which there are different algorithms, give examples of cases where one algorithm works better than the other, and reason about which algorithm is likely to work better overall.
- CT Building Block: Students will be able to evaluate and compare algorithms in terms of its efficiency (time and space requirements).
Hardware vs. Software

- **Hardware** refers to the physical devices
- **Software** refers to programs that run on the hardware to implement applications
Hardware

Computational Thinking
http://www.ugrad.cs.ubc.ca/~cs100
Computers have gotten much better at storing information in smaller spaces.

Today: 10nm

http://www.anandtech.com/show/2928
How has this changed computing?

Eniac: ~18,000 pieces of data

My laptop: ~8,000,000,000,000 pieces of data

http://www.computerhistory.org

Computational Thinking

http://www.ugrad.cs.ubc.ca/~cs100
How do computers organize all that data anyway? Let’s look at a related question.

Discuss in a group of 3-4.
Where do you store your clothes?
- in the closet
- drawer
- suitcases
- pile on the chair
- laundry hamper
Computer memory vs. laundry: registers

- The CPU/“chip” actually does the work
- It includes registers, which hold the data that the computer is actively working with
- This is a bit like the clothes that you are actually wearing
- It’s very small, very fast to access
Computer memory vs. laundry: cache

- The CPU also has several layers of *caches* – fast memory that is actually on the chip.
- This is a bit like keeping clean laundry in a small pile near where you put it on.
Computer memory vs. laundry: RAM

- The computer also has RAM. It’s not on the chip, but is on the motherboard.
- This is slower to access than the cache and a bit bigger, but fairly fast and fairly small.
- This is a bit like keeping clothes in a small closet.
The computer also has a hard drive/solid state drive (ssd).

This is not on the motherboard. It’s very slow to access, but generally much, much bigger.

This is a bit like keeping clean laundry in a storage locker or at your parents’
Even though these distances seem small, they make a real difference

Admiral Grace Hopper explains the nanosecond: https://www.youtube.com/watch?v=9eyFDBPkJyw
Making chips smaller also makes them faster, but that’s coming to an end: Moore’s Law no longer holds

"Intel cofounder Gordon Moore [...] observed in 1965 that every year twice as many [transistors] could fit onto a chip, and in 1975 adjusted the pace to a doubling every two years. [...] But Intel pushed back its next transistor technology, with features as small as 10 nanometers from 2016 to late 2017”

Making chips smaller also makes them faster, but that’s coming to an end: Moore’s Law no longer holds.

Computer speed and memory on a chip will double every 18 months to 2 years.

http://en.wikipedia.org/wiki/Moore's_law
Fortunately, not all computing speed up is due to hardware.

White line is hardware speedup
Green line includes software speedup (note log plot)
Software
Special note: Operating System

- An operating system is a special kind of software that allows the other software to run.
- Most of you probably use Windows or Mac OS.
So how does software work?

- Software works by programmers writing down algorithms in a special languages that the computer can understand.
- Reminder: an algorithm is a precise, systematic method for producing a specified result.
- Example: an algorithm for using the clicker
  1. Turn on the clicker by pressing the “On/Off” button.
  2. Select the clicker channel (AB)
  3. When I ask a question in class (and start the timer), select A, B, C, D, or E as your vote.
  4. If the “Vote Status” Light is green, the vote was received.
     Do nothing
     Else
     Vote again
What’s missing from that definition?

- A computer!
- We looked at one computerless algorithm already (sorting cards)
- Let’s look at another: https://www.youtube.com/watch?v=pWso-qRalIM#t=26s
- We’ll call the algorithm shown here the Breaking Bad algorithm.
That’s one way of doing it. Here’s another.

“It is a simple enough system," Bauby explains. "You read off the alphabet . . . until, with a blink of my eye, I stop you at the letter to be noted. The maneuver is repeated for the letters that follow, so that fairly soon you have a whole word.”

In other words, you go through each letter, row by row, and read off all the letters.

We’ll call this the Sequential algorithm
Which way is best? vs. **Sequential**

- Let’s assume that our goal is to minimize the number of letters that have to be looked at to spell a word. So each letter you look at has a cost of “1”
- We also have to decide if there’s any extra “cost” to having the “signals”. Let’s say each signal also costs “1”
Example of cost counting: letter F

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>E</td>
<td>F</td>
<td>G</td>
<td>H</td>
<td>3</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>I</td>
<td>J</td>
<td>K</td>
<td>L</td>
<td>M</td>
<td>N</td>
<td></td>
</tr>
<tr>
<td>O</td>
<td>P</td>
<td>Q</td>
<td>R</td>
<td>S</td>
<td>T</td>
<td></td>
</tr>
<tr>
<td>U</td>
<td>V</td>
<td>W</td>
<td>X</td>
<td>Y</td>
<td>Z</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

**Breaking Bad**
- 2 to get to the “E” row
- 1 to signal the “E” row
- 2 to get to “F” in the row
- +1 to signal “F”

**Sequential**
- 8 to get to the letter F
- +1 to signal “F”

---

6 total cost

9 total cost
Group Exercise (3-4 people): which way works best?

• Find a word that works better the Breaking Bad/video way.
• Find a word that works better the second/sequential way

Each word has to be at least 4 letters long.
Example of cost counting: letter F

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>B</td>
<td>C</td>
<td>D</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>E</td>
<td>F</td>
<td>G</td>
<td>H</td>
<td>3</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>I</td>
<td>J</td>
<td>K</td>
<td>L</td>
<td>M</td>
<td>N</td>
<td></td>
</tr>
<tr>
<td>O</td>
<td>P</td>
<td>Q</td>
<td>R</td>
<td>S</td>
<td>T</td>
<td></td>
</tr>
<tr>
<td>U</td>
<td>V</td>
<td>W</td>
<td>X</td>
<td>Y</td>
<td>Z</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

**Breaking Bad**
- 2 to get to the “E” row
- 1 to signal the “E” row
- 2 to get to “F” in the row
- +1 to signal “F”

6 total cost

**Sequential**
- 8 to get to the letter F
- +1 to signal “F”

9 total cost

Find a word that works better the **Breaking Bad/video** way.

Find a word that works better the second/sequential way.

Both words must be at least 4 letters!

Use the same chart

Computational Thinking
http://www.ugrad.cs.ubc.ca/~cs100
Bringing it back together

What are some words that work better the Breaking Bad way?
- Vest
- Seat

What are some words that work better sequentially?
- Taro
- Neat
- Teal
- Eats
Clicker question

So which is best in general?

A. The first method (Breaking Bad)
B. The second method (sequentially)
That problem was human, but not very common. Let’s look at sorting again

Your “computer” sorts things for you every day

- E-mails
- Contacts
- Music

There’s so much sorting that a LOT of time has been devoted to coming up with ways to sort.
Before we can sort, we need to swap

An oven does not always heat evenly. So when making cookies to cook them evenly, you have to change positions of the cookies: the ones on the top go to the bottom, and the ones on the bottom go to the top.

In groups, create an algorithm to swap the location of the cookie sheets in the picture. Assume you only have one oven mitt… and don’t have burn-proof hands.
Swapping algorithms

1. Take out the top row sheet
2. Put it on the cabinet
3. Pull out one on the bottom
4. Put it on the top rack
5. Put whatever is on the cabinet back in
Okay, great! Let’s get ready to sort.

- Get into your groups
- Each group will be handed a deck of cards
- Take out 7 cards from the same suit and shuffle them; you will use these to create hands
- Take out 7 more cards and place them face down. You will use these to represent “maximum” values.
- It does not matter if the Ace is high or low, as long as you are consistent.
Let’s sort! Sorting algorithm #1: Simple sort

1. Place the unsorted cards in the top row
2. Repeat steps 3 through 6 until no unsorted card remains
   3. Initially mark the first (leftmost) card
   4. Working right from the second card, compare the marked card to the current card. If the current card is smaller than the marked card, move the marker to the current card
   5. Move the marked card to the sorted hand
   6. Put a “Max*” (upside down) card in the empty unsorted slot
3. Stop

* The algorithm has to compare the slot, so there has to be something in it. “Max” is just a number higher than the others so it doesn’t get sorted in by accident.
Sorting algorithm #2: Selection Sort

1. Deal the unsorted cards
2. Put a divider at the left of the unsorted cards
3. Repeat steps 4 through 7 until one unsorted card remains
   4. Initially mark the first (leftmost) card
   5. For each card to the right of the second card:
      If the card is smaller than the marked card,
      move the marker to the current card
   6. Swap the marked card with the first unsorted card
      (just to the right of the divider)
   7. Advance the divider to the right one card

8. Stop
Sorting algorithm #3: Insertion Sort

1. Deal the unsorted cards
2. Put a divider after the first card
3. Repeat steps 4 through 9 until no card is to the right of the divider
   4. Select the first card to the right of the divider. This is the “new” card.
   5. Select the rightmost of the cards before the divider. This is the “old” card
   6. Repeat steps 8 and 9 until there is no “old” card, or the value of the “new” card is less than the value of the “old card”
      7. If the “new” card is less than the “old” card, swap the two
      8. Let the “old” card be the card to the left of the cards just compared
   9. Advance the divider to the right one card
10. Stop
Our first clicker question is up next. First, how to vote

- Turn on the clicker by pressing the “On/Off” button.
- Our clicker channel is channel AB – change each session
  - To change your frequency, hold your On/Off button for 2 seconds (the blue Power light will flash)
  - Press “AB”: Your “vote status” will turn green
- 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? Just vote again.
    **You can change your vote as long as the timer is going.

Normally these would be graded for class participation, but today’s just for fun.

Computational Thinking
http://www.ugrad.cs.ubc.ca/~cs100
Which sort is best?

A. Simple Sort
B. Selection Sort
C. Insertion Sort

- Insertion sort is more efficient in sorting (more linearly)
- Insertion sort felt like the fastest
There are many ways to define “best”

Common ones are time and space

We’ll start with space

Things that take space include:
- memory slots for cards
- markers, dividers

We’ll concentrate on memory slots
How many memory slots are needed for cards for Simple Sort for 7 cards? (don’t forget swapping!)

A. 7  
B. 8  
C. 14  
D. 15  
E. Other

When counting, ignore things like markers and dividers

Simple Sort Reminder:
1. Place the unsorted cards in the top row
2. Repeat steps 3 through 6 until no unsorted card remains
   3. Initially mark the first (leftmost) card
   4. Working right from the second card, if the current card is smaller than the marked card, move the marker to the current card
   5. Move the marked card to the sorted hand
   6. Put a “Max” (upside down) card in the empty unsorted slot
   7. Stop
How many memory slots are needed for cards for **Selection Sort** for 7 cards (don’t forget swapping!)

A. 7

B. 8

C. 14

D. 15

E. Other

When counting, ignore things like markers and dividers

B. 8

Selection sort reminder:

1. Deal the unsorted cards
2. Put a divider at the left of the unsorted cards
3. Repeat steps 4 through 7 until one unsorted card remains
4. Initially mark the first (leftmost) card
5. For each card to the right of the second card:
   - If the card is smaller than the marked card, move the marker to the current card
6. Swap the marked card with the first unsorted card (just to the right of the divider)
7. Advance the divider to the right one card
8. Stop
How many memory slots are needed for cards for Insertion Sort for 7 cards (don’t forget swapping!)

A. 7
B. 8
C. 14
D. 15
E. Other

When counting, ignore things like markers and dividers

B. 8

Insertion sort reminder

1. Deal the unsorted cards
2. Put a divider after the first card
3. Repeat steps 4 through 6 until no card is to the right of the divider
   4. Select the first card to the right of the divider
   5. Swap this card to the left until it arrives at the correct sorted position
   6. Advance the divider to the right one card
7. Stop