(Text?) Editors
Learning Basic Math

% echo "#include <iostream>
> using namespace std;
>
> int main() {
>     cout << "2 + 2 = " << 4 << endl;
>
>     return 0;
> } " > math.cpp

- That was cheap!
- Let's check it out...
Quick Game: Name an editor
Quick Game: Name an editor

THIS IS WHY YOU SHOULDN'T INTERRUPT A PROGRAMMER

if $c == ','$

...so if the current character is a comma, we set the backtracking flag...

commit #5765 to here
caused bug... How
or did it just expose it?

Hey, so I just sent you an email about that thing
Editors are the stuff between your head and the screen

- You can walk and talk
- You can walk and read
- Could you walk and code?
Editors are the stuff between your head and the screen

- You can walk and talk
- You can walk and read
- Could you walk and code?

Writing code is like writing text, until it isn't

- Organize an outline
- Choose a section, author a cohesive narrative
  - How long of a paragraph can you hold in your head?
  - How many lines of code can you hold in your head?
Editors are the stuff between your head and the screen

- You can walk and talk
- You can walk and read
- Could you walk and code?

Writing code is like writing text, until it isn't

- Organize an outline
- Choose a section, author a cohesive narrative
  - How long of a paragraph can you hold in your head?
  - How many lines of code can you hold in your head?

Programmers are translating all the time

- Between what you want the machine to do (concept/spec) and how you tell the machine to do it (code)
Editor Wars: An Educated Guess

- Programmers fight about silly things
  - Tabs vs Spaces, Where to put Braces
- That may not be so silly
Editor Wars: An Educated Guess

- Programmers fight about silly things
  - Tabs vs Spaces, Where to put Braces
- That may not be so silly

It is all about what best minimizes the friction between what's in your head and what's on the screen

- People tell me DVORAK is faster than QWERTY\(^1\), but probably not for me

---

\(^1\) Cool article questioning the "origin of QWERTY was to slow down typists" myth
Lecture Today

- **The Basics**
  1. Marcus struggles mightily with emacs
  2. And then does great in vim
  3. We cover where the mouse is a winner
     - *Goal:* Minimal competency in everything, so you can work with others

- **The Fancy Stuff**
  1. In "old" editors
  2. In "new" editors
  3. And what each other can't do
     - *Goal:* Exposure to the kind of things you can do. What is most useful is very personal.
Chart? Chart.

<table>
<thead>
<tr>
<th></th>
<th>emacs</th>
<th>vim</th>
<th>gedit</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Save file</strong></td>
<td>C-x C-s</td>
<td>:w</td>
<td>click save</td>
</tr>
<tr>
<td></td>
<td></td>
<td>:q!</td>
<td>click quit</td>
</tr>
<tr>
<td></td>
<td></td>
<td>:w then :q</td>
<td>click no</td>
</tr>
<tr>
<td></td>
<td></td>
<td>:wq</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>:x</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>ZZ</td>
<td></td>
</tr>
<tr>
<td><strong>Quit without saving</strong></td>
<td>C-x C-c n yes</td>
<td>:q!</td>
<td>click quit</td>
</tr>
<tr>
<td></td>
<td>M-x kill-emacs</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Save and quit</strong></td>
<td>C-x C-s then C-x C-c</td>
<td>:w then :q</td>
<td>click save</td>
</tr>
<tr>
<td></td>
<td>C-x C-c y</td>
<td>:wq</td>
<td>click quit</td>
</tr>
</tbody>
</table>
## Chart? Chart.

<table>
<thead>
<tr>
<th></th>
<th>emacs</th>
<th>vim</th>
<th>gedit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Save file</td>
<td><code>C-x C-s</code></td>
<td><code>:w</code></td>
<td>click save</td>
</tr>
<tr>
<td>Quit <strong>without</strong> saving</td>
<td><code>C-x C-c n yes</code></td>
<td><code>:q!</code></td>
<td>click quit</td>
</tr>
<tr>
<td></td>
<td><code>M-x kill-emacs</code></td>
<td></td>
<td>click no</td>
</tr>
<tr>
<td>Save and quit</td>
<td><code>C-x C-s</code> then <code>C-x C-c</code></td>
<td><code>:w</code> then <code>:q</code></td>
<td>click save</td>
</tr>
<tr>
<td></td>
<td><code>C-x C-c y</code></td>
<td><code>:wq</code></td>
<td>click quit</td>
</tr>
<tr>
<td></td>
<td></td>
<td><code>:x</code></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td><code>ZZ</code></td>
<td></td>
</tr>
</tbody>
</table>

### A digression: `w` vs `x` (vs `ZZ`)

- `:w` always writes, `:x` only if changed
  - Why do you care?
  - It's a bit annoying to need to care
<table>
<thead>
<tr>
<th>Menu</th>
<th><code>emacs</code></th>
<th><code>vim</code></th>
<th><code>gedit</code></th>
</tr>
</thead>
<tbody>
<tr>
<td>Save file</td>
<td><code>C-x C-s</code></td>
<td><code>:w</code></td>
<td>click save</td>
</tr>
<tr>
<td>Quit <strong>without</strong> saving</td>
<td><code>C-x C-c n yes</code></td>
<td><code>:q!</code></td>
<td>click quit</td>
</tr>
<tr>
<td></td>
<td><code>M-x kill-emacs</code></td>
<td></td>
<td>click no</td>
</tr>
<tr>
<td>Save and quit</td>
<td><code>C-x C-c y</code></td>
<td><code>:x</code> or <code>ZZ</code></td>
<td>click save</td>
</tr>
<tr>
<td></td>
<td></td>
<td><code>:wq</code></td>
<td>click quit</td>
</tr>
<tr>
<td>Enter some text</td>
<td><code>...just type</code></td>
<td><code>i</code> then type</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Search for a string</td>
<td><code>C-s</code></td>
<td><code>/</code></td>
<td>make search appear</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>click search</td>
</tr>
</tbody>
</table>
"Enter some text" in vim, oh the many ways

_i_ is the easiest, yet probably least used, way to enter text

- _i_ - Go into insert mode here
- _I_ - Go into insert mode at the beginning of the line
- _a_ - Go into insert mode right after here
- _A_ - Go into insert mode at the end of the line
- _o_ - Insert a new line below and go into insert mode
- _O_ - Insert a new line above and go into insert mode
- _s_ - Delete this character and go into insert mode
- _S_ - Delete this line and go into insert mode
- _C_ - Delete the line from here and go into insert mode
- _R_ - Enter _replace_ mode
Reviewing some old magic

Remember Homework 2?

$ wget 'https://drive.google.com/uc?id=0B4qlH840ZwikbkZLS3Z5YTVSeW8&export=dow
nload' -O eecs280-w15-p2.tgz
$ tar -xf eecs280-w15-p2.tgz
$ echo -e '#include <cassert>
#include "p2.h"
' > p2.cpp
$ grep ';' p2.h | grep -v ' */' >> p2.cpp

$ vim p2.cpp
# press 'jjj' so that your cursor is on the i of int sum(...
# press 'qq' to begin recording a macro into the vim register q
# press 'f;' jump to the ';' character
# press 's' remove character under the cursor and enter insert mode
# insert the needed text:
# ' <space>{<enter>assert(false);<enter>}<enter><escape>'
# press 'j' so that your cursor is on the i of int product(...
# press 'q' to finish recording the macro
# press '16@q' 16 times play the commands stored in register q
"Vim sucks because"...

it can't copy / paste

- Yes it can, you just \texttt{y}ank and \texttt{p}aste instead
- And to cut you off, you can use \texttt{"+y} and \texttt{"+p} to yank and paste from the system clipboard

you can't use the mouse

- Yes you can, you just have to \texttt{set mouse=a}

the defaults are terrible not great

- Now you're getting somewhere

the time and pain required to learn it aren't worth the payoff

- I am honestly unsure.
This class in a nutshell:

- You are willing to type the same command over and over until you aren't
  - So you learn about the up arrow
- You master the game of up-up-up-enter up-up-up-enter until you drive yourself nuts
  - So you learn how to put these commands in a script
- *You might be willing to master a new skill faster if you knew it was out there, and someone guided you towards how to do it*

The goal is to expose you to many things, hope you master half, and can come back and pick up the rest when you're ready

- We do try to emphasize the ones worth mastering first
A quick tour of some other editors and what they can do

- "General purpose" editors
  - Atom
  - Sublime
- Integrated development environments
  - XCode
  - Eclipse
  - Visual Studio