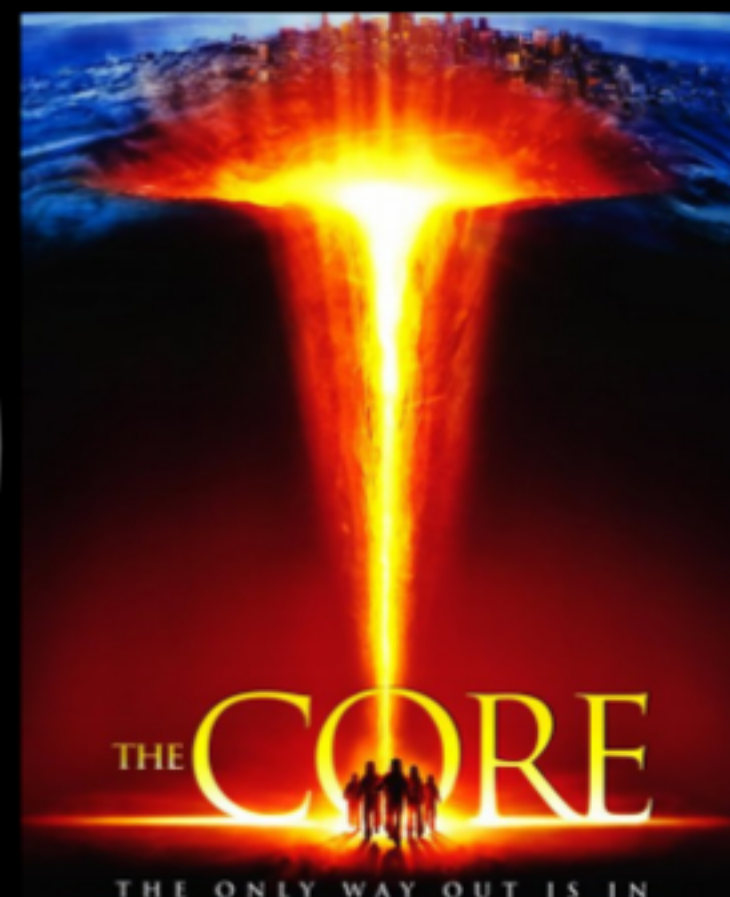


ABOUT ME

- * I am a Masters' student of Computer Science at USU
- * I work as a software engineer at Spillman Technologies
- * I have been a full-time Vim user since 2003



Twelve Angry Men

People, I just want to say, you know, can we all get along?
Can we get along?
-- Rodney King'



People from user culture (**Muggles**)

People from programmer culture (**Wizards**)





What issues divide these cultures?



My life is like a cat
I'm not good at thinking like a robot
I don't want a lifestyle, I just want to see cats do cute stuff!
I literally don't care how my computer works, so long as it can Tumblr
I do all of my work on an iPad, and so should you!

W() _ _ _ _ _ | | _ _ | | _ _ | () _ _ _ _ _
 \ \ \ / | | _ _ _ | | _ _ _ (- < | | _ \ - _) | | | / / - _) _
 \ ^ \ / | / _ _ \ , _ | | \ _ , _ / _ / | _ . _ ^ _ | | | | \ \ _ _ ()

- * A computer is a tool that can do **anything** I can imagine
- * I'm **good at thinking** logically and literally
- * The way I compute **is** my **lifestyle**
- * I am **comfortable** with the idea of working on a computer
- * Computers **do my bidding**

[illegible]

- * Help you to recognize when you work like a **Muggle**
- * Teach you to think like a **Wizard**
- * Start you off with **Vim's 10** best features



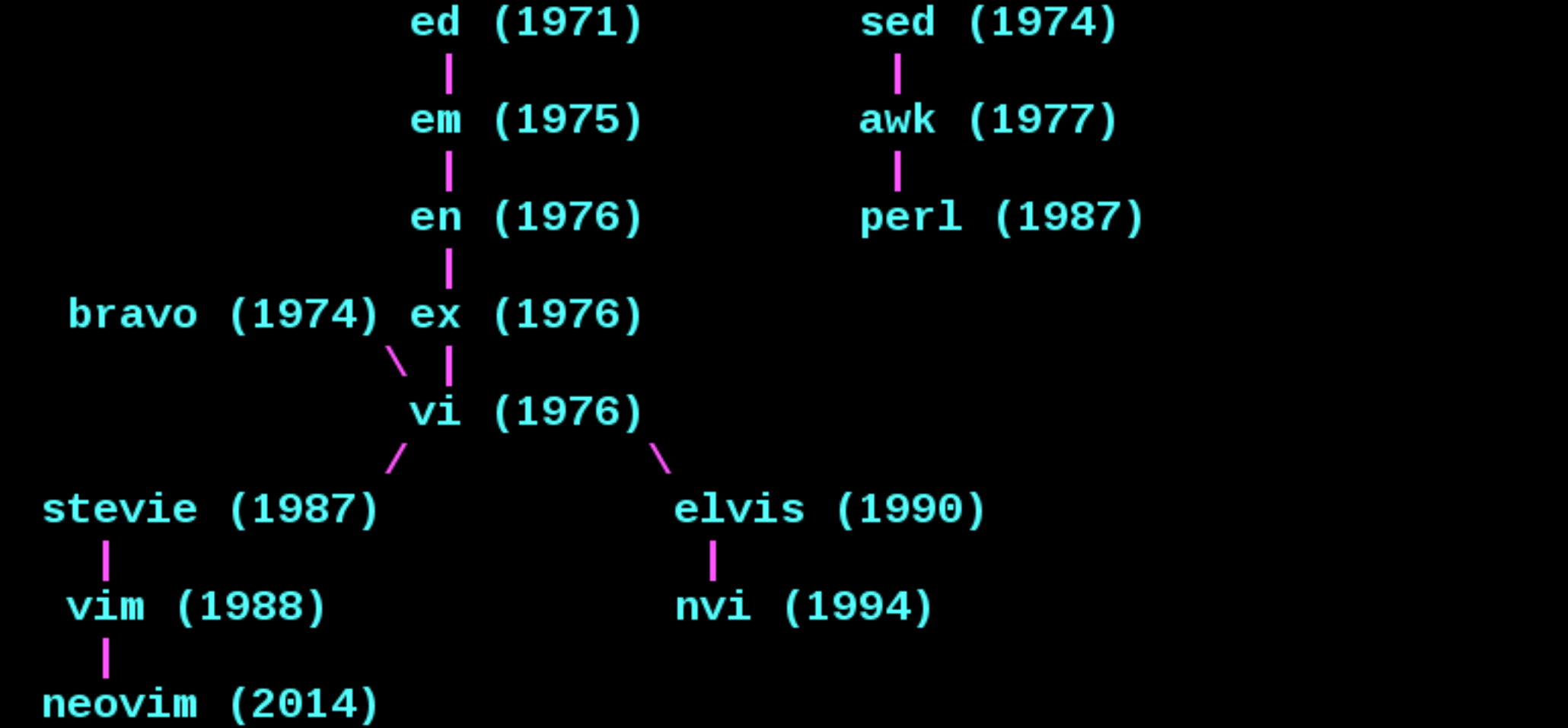
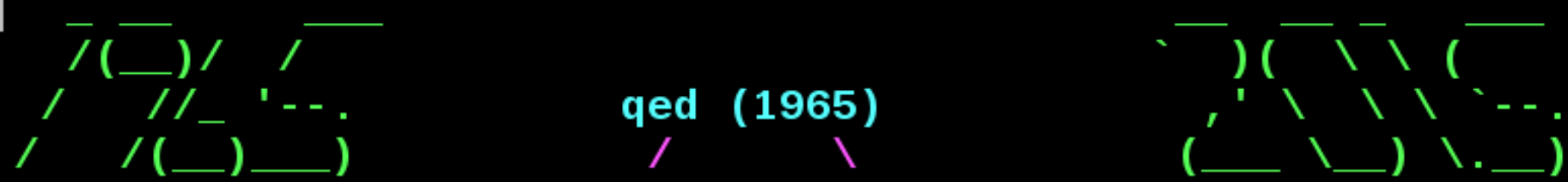
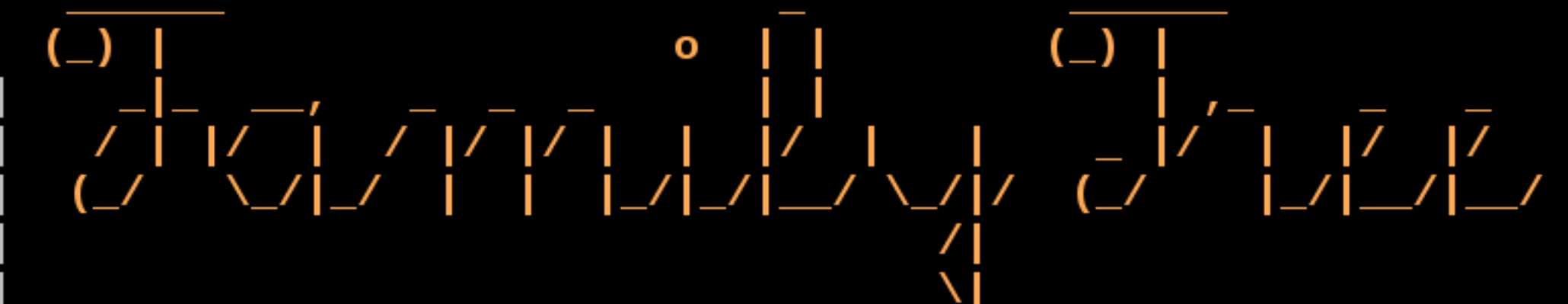
Install Vim (if you don't already have it)

```
$ wget http://unnovative.net/OpenWest_vim.tgz
```

This presentation is interactive

At the beginning of the 21st century, the world was a very different place than it is today. The world was a very different place than it is today.

Those who cannot remember the past
are condemned to re-implement it poorly...
...in JavaScript



The Lear Siegler ADM 3A



<http://www.vintage-computer.com/otheritems.shtml>

vi's design decisions were accidentally awesome

CUU = CXXU



```
( _ _ ) ( _ ) _ ( _ ) ( _ _ ) ( \ ( _ ) / _ _ ) / _ _ ) ( \ / ) ( _ _ ) ( _ _ \ ( _ _ ) ( _ _ ) / _ _ )
 ) ( _ ) _ ( _ ) ( _ ) ( ( ( _ - . \ _ _ \ \ / _ _ ) ( _ ) ( _ ) ) ( _ ) ( _ ) _ ) \ _ _ \
 ( _ _ ) ( _ ) ( _ ) ( _ _ ) ( _ ) \ _ _ \ _ _ / ( _ _ / \ / ( _ _ ) ( _ _ / ( _ _ ) ( _ _ ) ( _ _ /
```

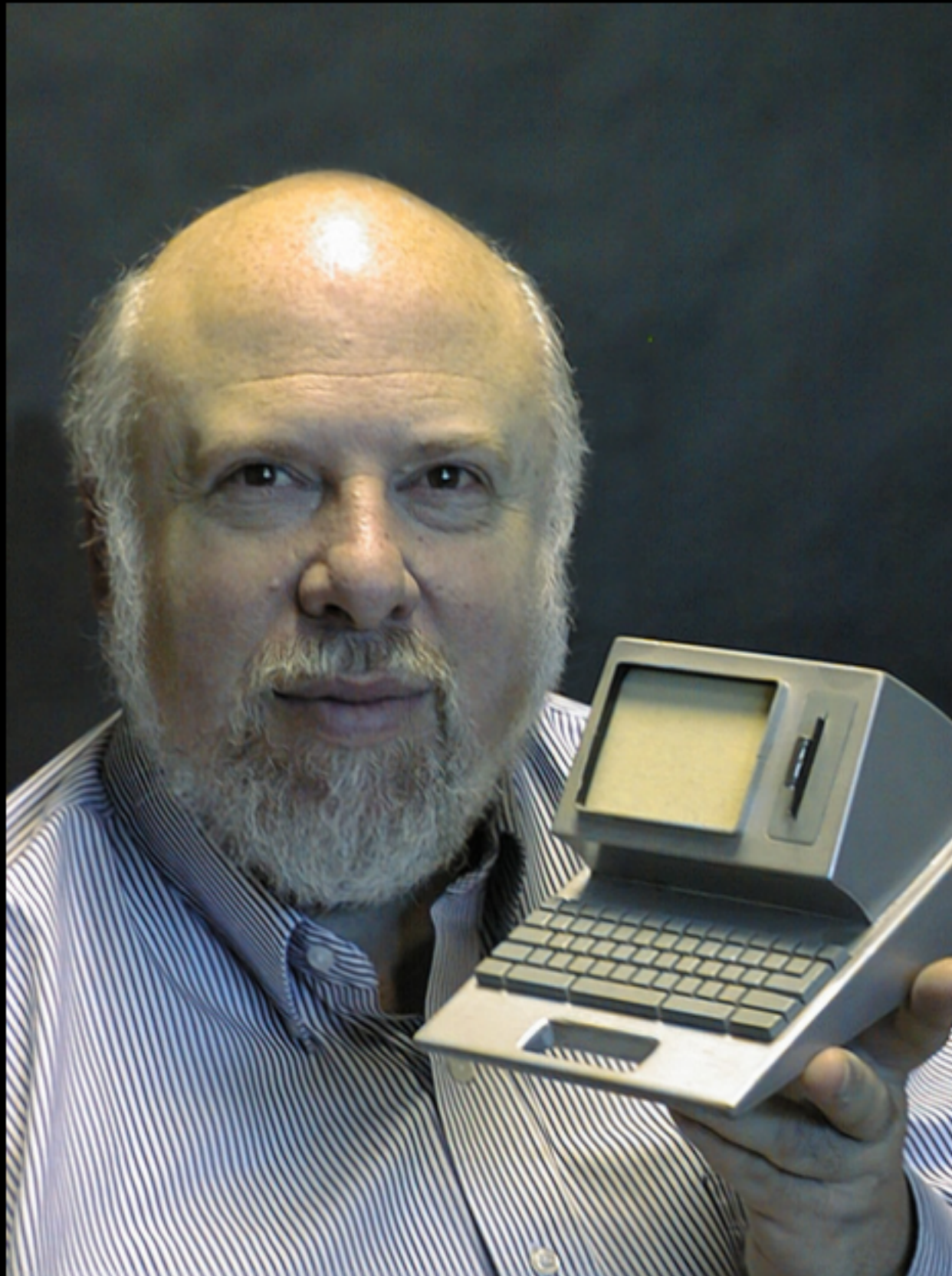
```
( _ \ / \ / _ ) ( _ _ \ ( _ _ ) ( \ ( _ ) / _ _ ) ( )
 ) ( _ ) / ) ( _ ) ( _ ) ( ( ( _ - .
 ( _ / \ _ _ ) ( _ ) \ _ _ ) ( _ _ ) ( _ ) \ _ _ \ _ _ / ( )
```

- * Is not 'discoverable'
No affordances, tooltips, guides, etc.
- * No GUI (i.e. no pointer support)
- * Too many 'cryptic' commands
Ctrl-C != copy, Ctrl-X != cut, Ctrl-Z != undo, etc.



- * Is modal (**BOO! HISS!**)...
...and doesn't tell you which mode you're in

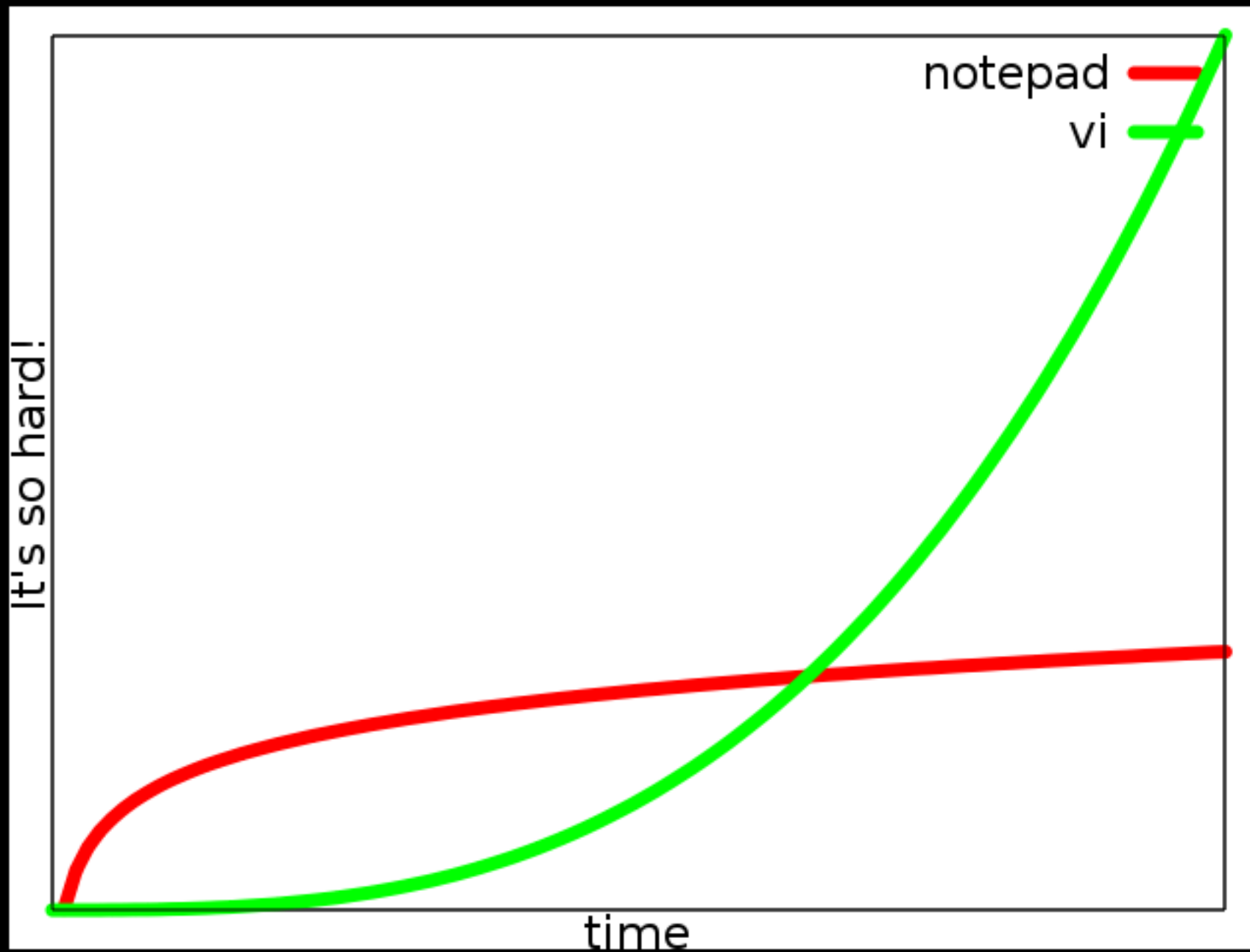
"Modes are a significant source of errors, confusion, unnecessary restrictions, and complexity in interfaces."



-- Jef Raskin
picture by Aza Raskin

And yet, despite all this, vi(1) is used today by more people than ever!

Why do you think this is?



UI experts don't count on your ability or desire to **learn**

Who are the "experts" designing interfaces for?

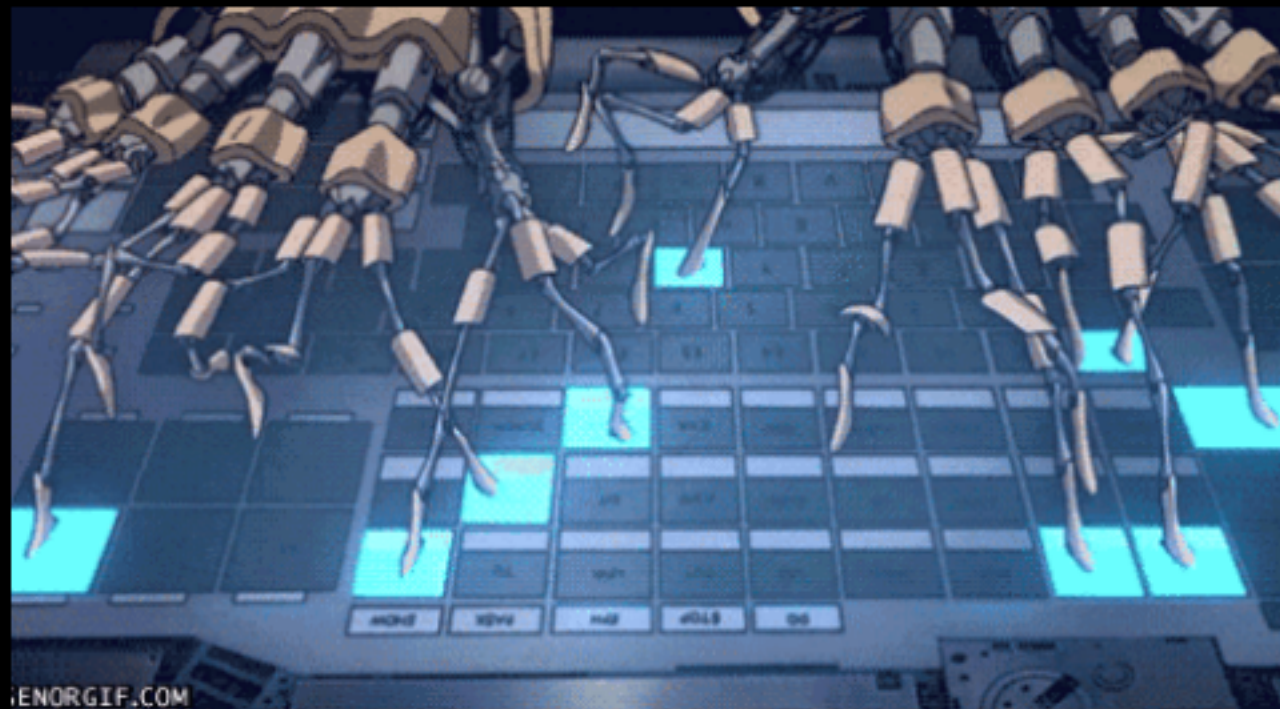


Their prototypical user combines this much intellect...



... with this many hands

`vi`, on the other hand, unleashes your inner cyborg



Old school is the best school

old school is the best school



Regular Expressions

You don't have to experience a file linearly

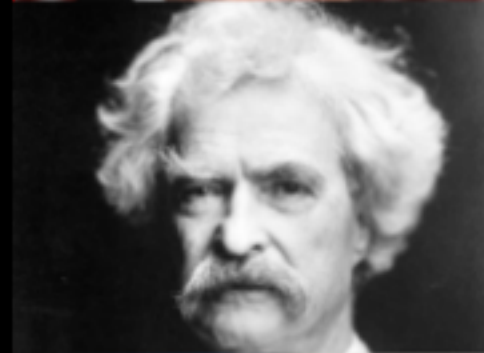
With regexes you can always

0. find what you are looking for
1. count how many times it occurs
2. make many tedious changes in one shot

Regular Expressions do the stuff that computers are good at

< /
/
/
/
/_ /
/_ / () Marks

No, not those Marks!



< /
 / /
 / /
 / _
 / _ / (_) Marks

Use marks to keep track of interesting places in a file

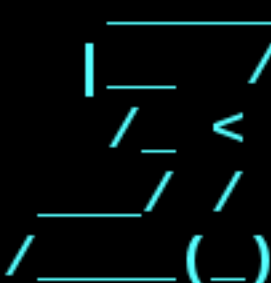
`vim` has already been setting marks for you;
you ought to use them!



Commands prefixed with a ':' are holdovers from ex(1)

You can apply them with sniper-precision with marks and regexes

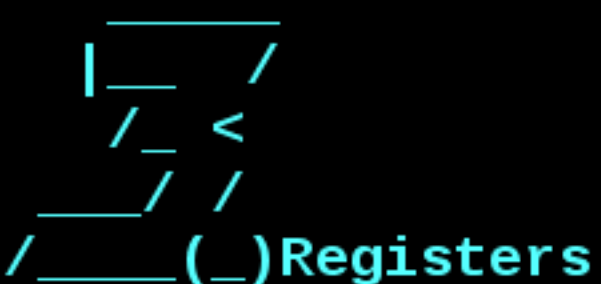
Type Ctrl-F to edit the command-line in a Vim buffer



Registers

a.k.a. "copy & paste buffer", a.k.a. "the clipboard"

vi fills some registers for you automagically as you work



vi can execute the contents of a register as a macro

Vim may record your keystrokes into a register for convenient playback

data = code

Young marklar, your marklars are wise and true.
-- Marklar

People are getting very excited about a new generation of pretty editors

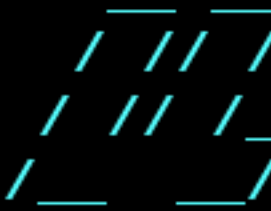


But they still use a mouse to convey your intentions to the computer

/ / / /
/ / / /
/ _ _ /
/ _ / (_) Vi is a language



If we're being honest, how well does this really work?



Vi is a language

This is the reason no other editor will ever replace vi

Vi's language composed of three simple concepts:

Count Operator Motion

Count is optional and often overlooked:

A computer doesn't mind mindless repetition, but you should!

Why hit a button **thirty** times when you can just say "**30**"?

Operator is one of a few familiar commands:

| | c | |
| | _ | |
| / _ \ |

change

| | d | |
| | _ | |
| / _ \ |

delete

| | y | |
| | _ | |
| / _ \ |

yank

Motion is the really clever bit

Bill Joy realized that text is intrinsically structured into

words,
lines,
sentences,
paragraphs,
blocks,
etc.

If this is how people percieve text, then so should their editor!

Just about anything that moves the cursor counts as a **motion**:

marks,
regular expressions,
h j k l of course,
and so much more!

A computer shall not waste your time or require
you to do more work than is strictly necessary.
-- Jef Raskin

c o m

Lesson 1



Over the years (and **Vim** has been around for 23 of them)
Vim has grown under the charge of Bram Moolenaar, its BDFL



One developer, Thiago de Arruda,
decided to take matters into his own hands and created a fork




NeoVim is very much in alpha, but has some promising features already:

- * Code cleanup and refactoring
- * Asynchronous processing
- * Meta and Ctrl+Shift key chords recognized in TTY
- * MessagePack remote API
- * A very trendy-looking webpage

```
git clone https://github.com/neovim/neovim.git
```


EXERCISE 1





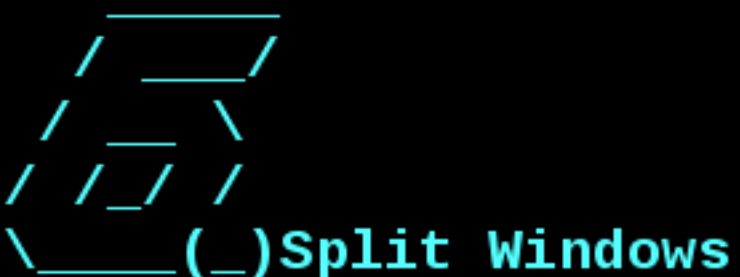
/(_)_I work with a bunch of tools...

"Vim is a component among other components"
:help design-speed-size

```
:noremap Q !!$SHELL<CR>
```

DO THE FOLLOWS

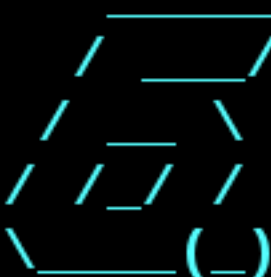


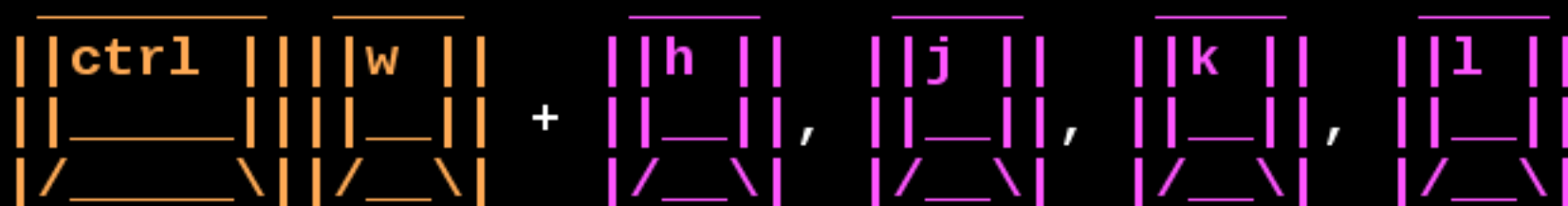


Vim is very window-flexible

Open five files at once...

...or open one file in five places; it's all good!

 Split Windows



It's not unlike a tiling window manager (XMonad, Awesome, dwm, i3)



()Split Windows

This naturally proceeds to "Can I compare two files in **Vim**?"

ctrl				w			
/				\			

+

h		j		k		l	
/		\		/		\	

,

h		j		k		l	
/		\		/		\	

,

h		j		k		l	
/		\		/		\	

,

h		j		k		l	
/		\		/		\	

\\ / () _ _ _ _ | () _ _ _ _ / _ _ _ _ _ _ _ _
\\ / | _ _ _ _ / | | / _ _ _ _ | | | / _ _ _ _ _ _ _ _
\\ / | _ _ _ _ / | | / _ _ _ _ | | | / _ _ _ _ _ _ _ _






/_/_/(_)**Visual Mode**

A flexible and easy way to select a piece of text for an **operator**

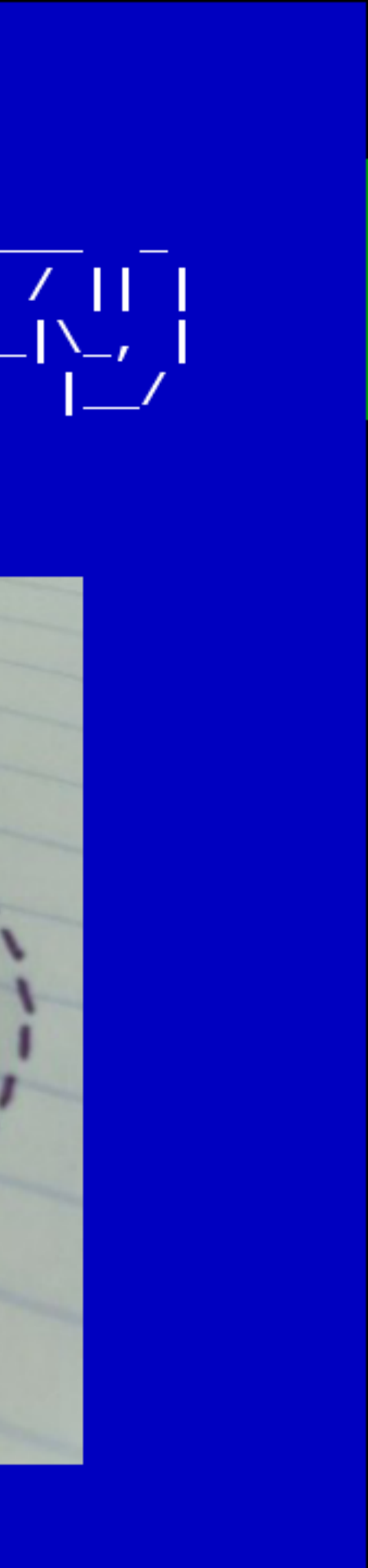
It is also the only way to select a block of text

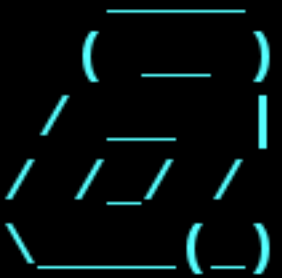
Visual Mode

Visual Block mode is a good substitute for trendy "multiple cursors"
(:substitute// covers the rest of the use cases)

`<C-v>I` and `<C-v>A`

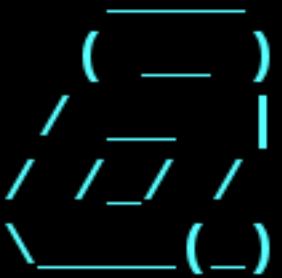
wherever you would use multiple cursors





Insert mode completion

<C-n> Pops up a small window containing keywords from your files



Insert mode completion

<C-x><C-f> Pops a list of filenames

tl;dr

<C-n> and <C-p> from insert mode

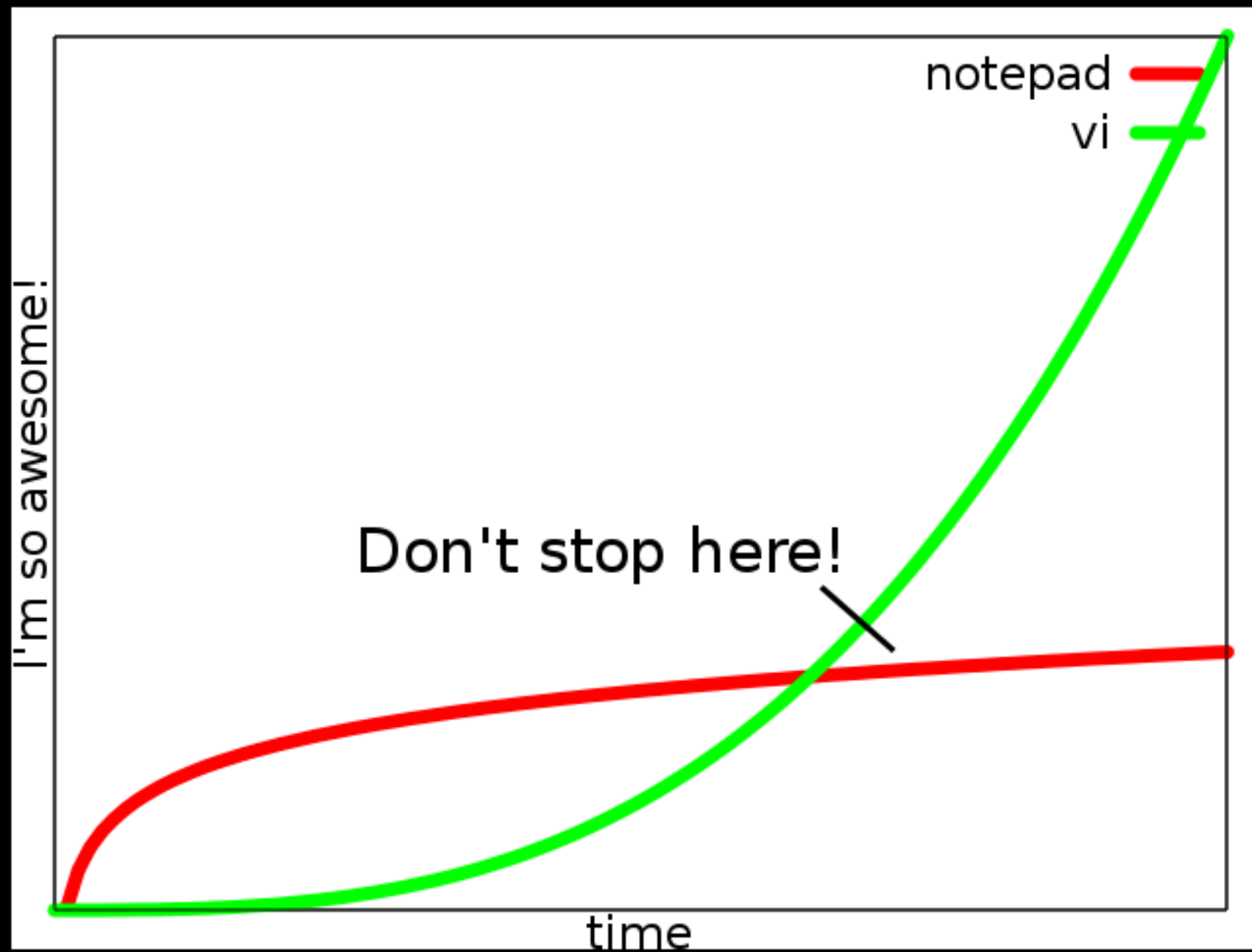
LOST IN TRANSLATION



The trick to learning **Vim**, or any other useful thing
is to do it a little bit at a time



The great thing about **Vim** is that there is always more to learn



///

THE EXOTIC

If you don't remember anything else...



/_____(_)Text Objects

Text objects allow **Vim** to recognize textual structures such as:

"quoted" 'string' `literals`,

[brackets],

(parens),

<Tags>

<markup type="XML">

<nested/>

</markup>

</Tags>

{

blocks.of(\$source, 'code');

}

And so much more!

Text Objects

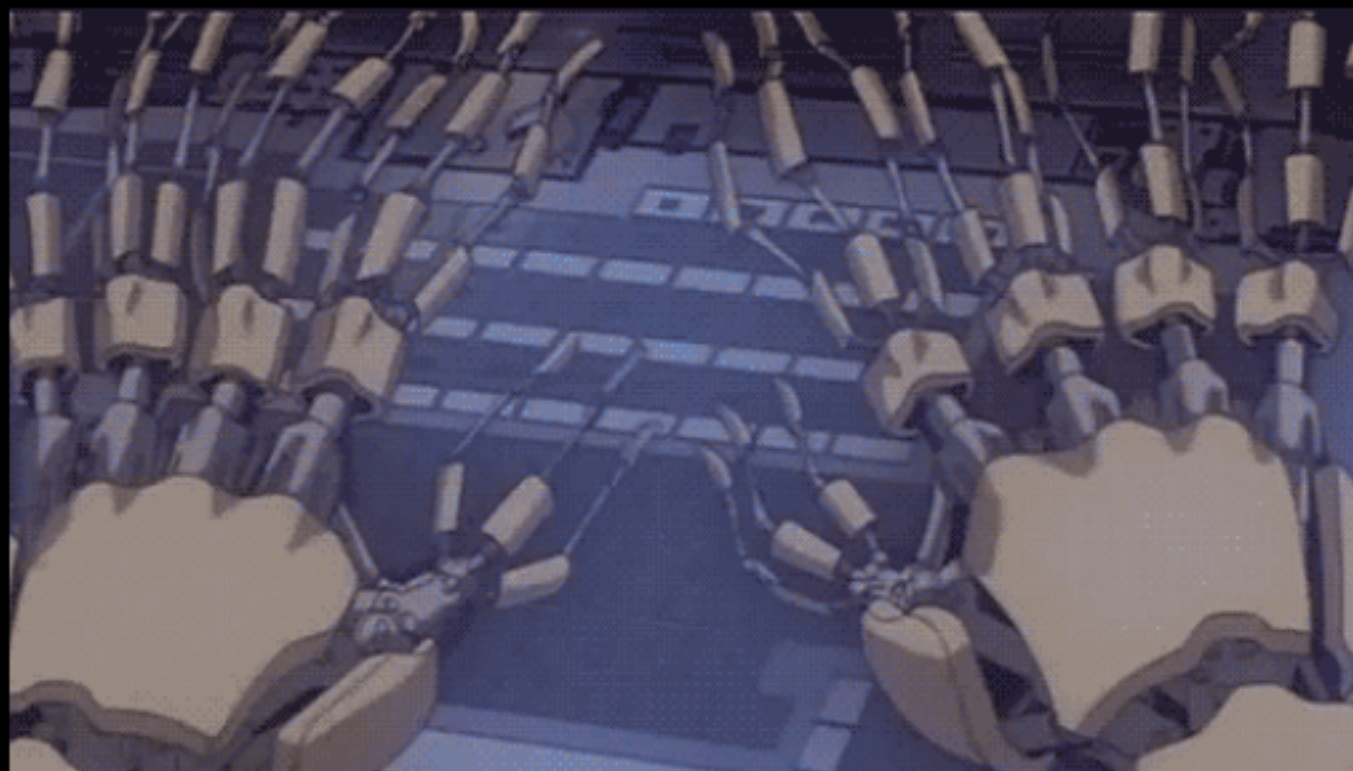
In the  scheme of things,

These fit into the **motion** category.

However, they don't move the cursor like the other motions.

tl;dr

<u> d </u>	<u> a </u>	<u> t </u>	<u>/ ^ \ /</u>	<u> \ </u>	<u> </u>	<u> </u>	<u>\ \</u>
<u> _ </u>	<u> _ </u>	<u> _ </u>	<u>(() (</u>	<u> M </u>	<u> </u>	<u> _</u>	<u>))</u>
<u>/ _ \</u>	<u>/ _ \</u>	<u>/ _ \</u>	<u>\ \ _ ^ \</u>	<u>_ </u>	<u> _</u>	<u> _</u>	<u>/ _ /</u>



THE EDIT

Like a lot of things about UNIX,
[vi] only **seems** difficult

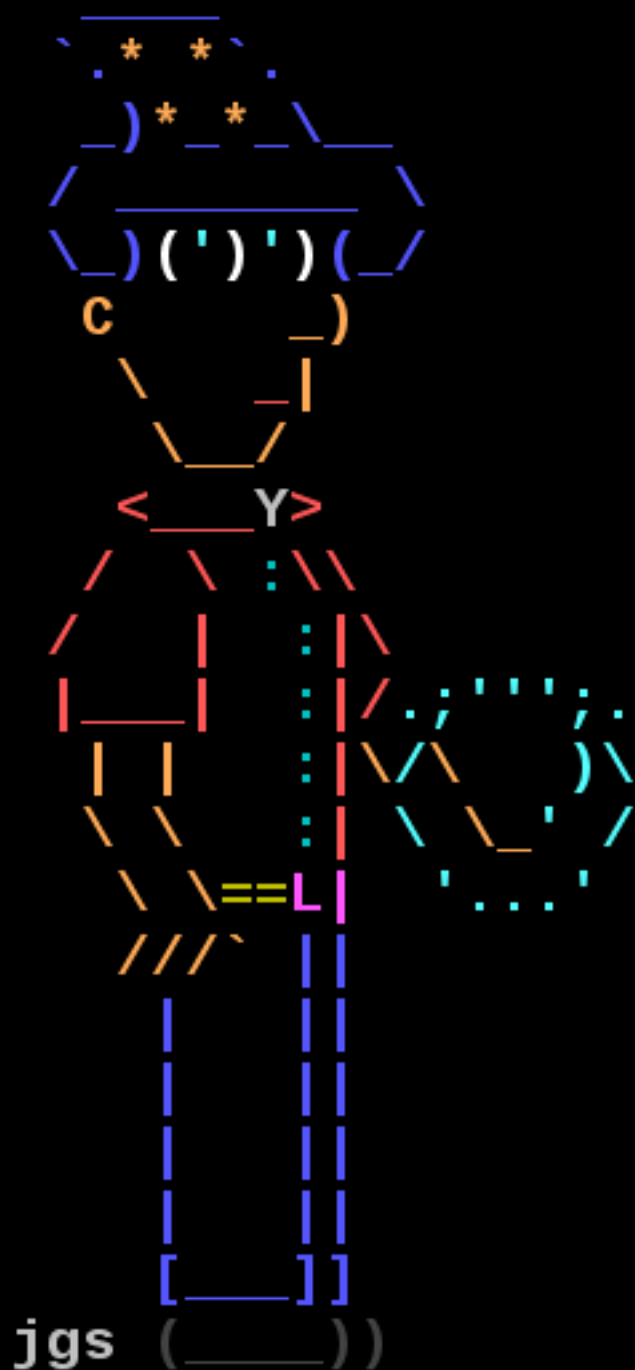
-- Tim O'Reilly

Each of these skills let you do more work with less effort:

0. Regular Expressions
1. Marks
2. Ex commands
3. Registers
4. vi is a language
5. External programs
6. Splitting windows
7. Visual mode
8. Insert-mode completion
9. Text objects

The key is to work on one thing at a time for a few days

- * It will either become muscle-memory
- * Or it won't fit your style



Before we part, I'd like to leave you
with one final piece of advice:

```

/      88888888b 8888ba.88ba      .d888888      a88888b. .d88888b      \
|      88          88  `8b  `8b d8'          88  d8'      `88 88.      "'
|      a88aaaa      88      88      88 88aaaaa88a 88          `Y88888b.
|      88          88      88      88 88          88 88          `8b
|      88          88      88      88 88          88 Y8.      .88 d8'      .8P
|      88888888P dP      dP      dP 88          88      Y88888P'  Y88888P
|
|      .d88888b  dP          dP  a88888b. dP          dP .d88888b  dP dP dP
|      88.      "' 88          88 d8'      `88 88      .d8' 88.      "' 88 88 88
|      `Y88888b. 88          88 88          88aaa8P'  `Y88888b. 88 88 88
|          `8b 88          88 88          88      `8b.          `8b dP dP dP
|      d8'      .8P Y8.      .8P Y8.      .88 88          88 d8'      .8P
|      Y88888P  `Y88888P'  Y88888P' dP          dP  Y88888P  oo oo oo
|
\

```

```

      \      ^__^
      \      (oo)\_______
          (_____)       )\  /
              ||----w |
              ||     ||

```