

How do I edit a file in a Terminal?



NCML lab meeting (open)

2024-08-21 Seung-Goo KIM

"Terminal" of what?



IBM 9020 equipment at the Jacksonville ARTCC at 1970s

Source: https://www.faa.gov/about/history/historical_perspective (Ch. 4, pp. 19)





The first Unix in 1969, Bell Lab

The mighty PDP-7

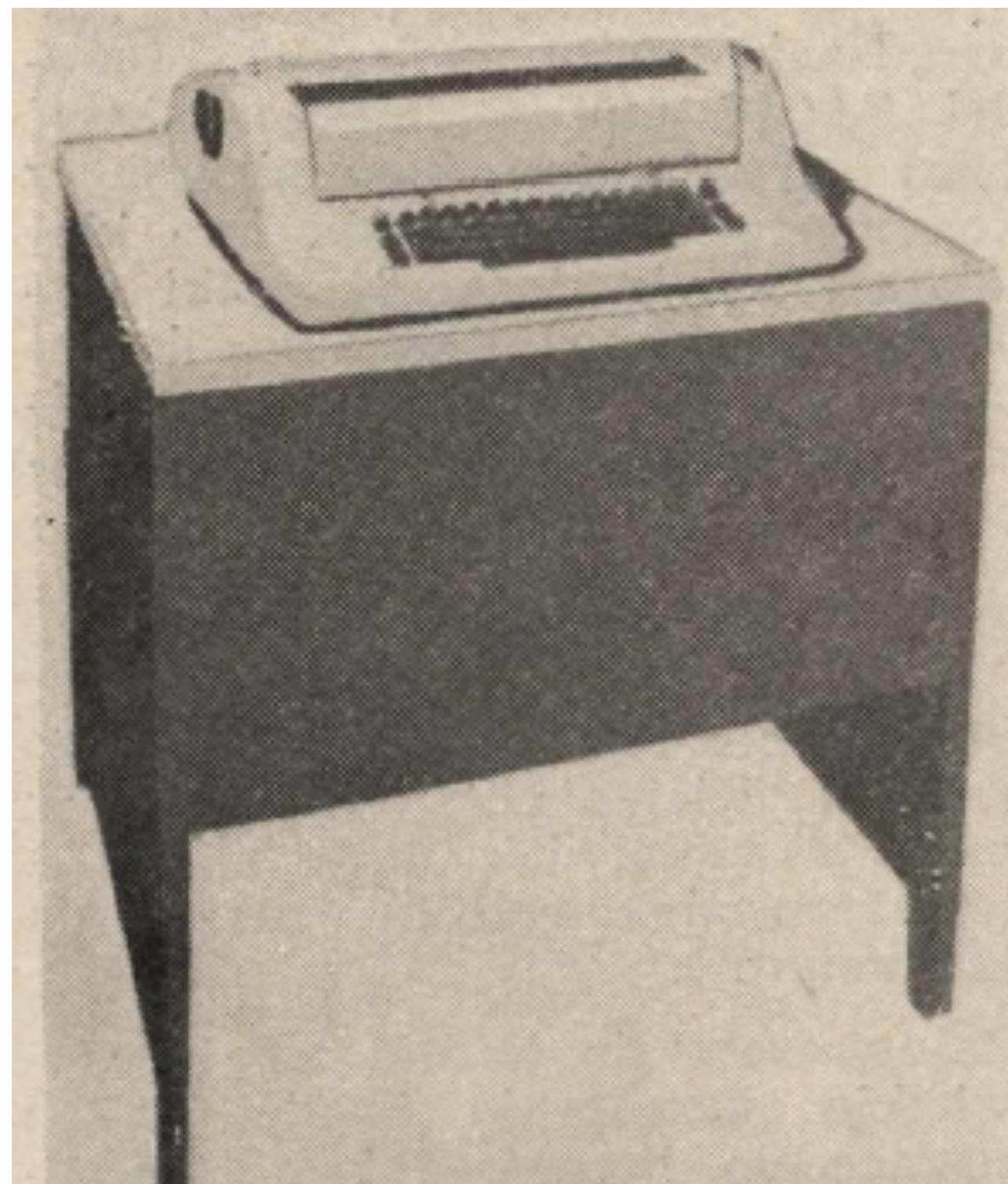


Ken (seated) and Dennis (standing) at a PDP-11 in 1972.



Early "Terminals" and modern emulations

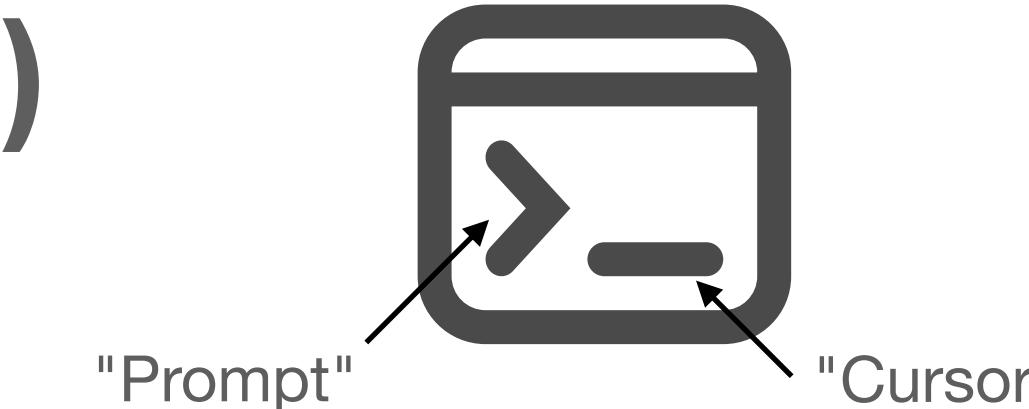
"Command line interface" (CLI)



IBM 2741 (1960s-1970s), "Teleprinters"



DEC VT100, 1978, the first to support cursor control on display



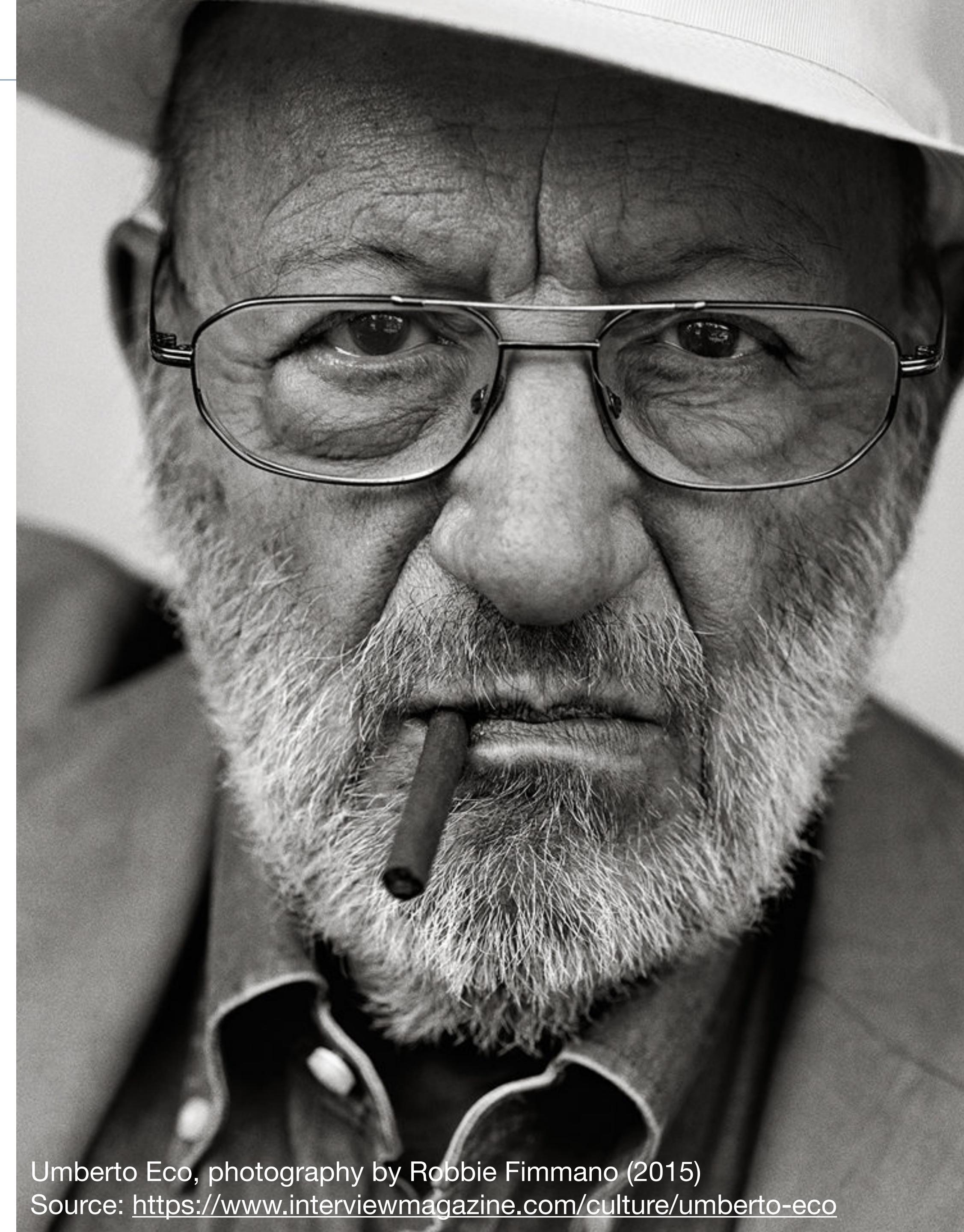
MacBook Air, 2022



The whole "computer" (CPU+GPU+RAM+SDD) is under the keyboards and it still looks like a "terminal" (keyboard+display)

Why do you want to emulate an ancient relic?

- "The book is like the spoon, scissors, the hammer, the wheel. Once invented, it cannot be improved. You cannot make a spoon that is better than a spoon." (Umberto Eco, Jean-Calude Carriere, 2009, "This is Not the End of the Book", Chapter 1, Northwestern University Press)
- So is the terminal! (Meself)



Umberto Eco, photography by Robbie Fimmano (2015)

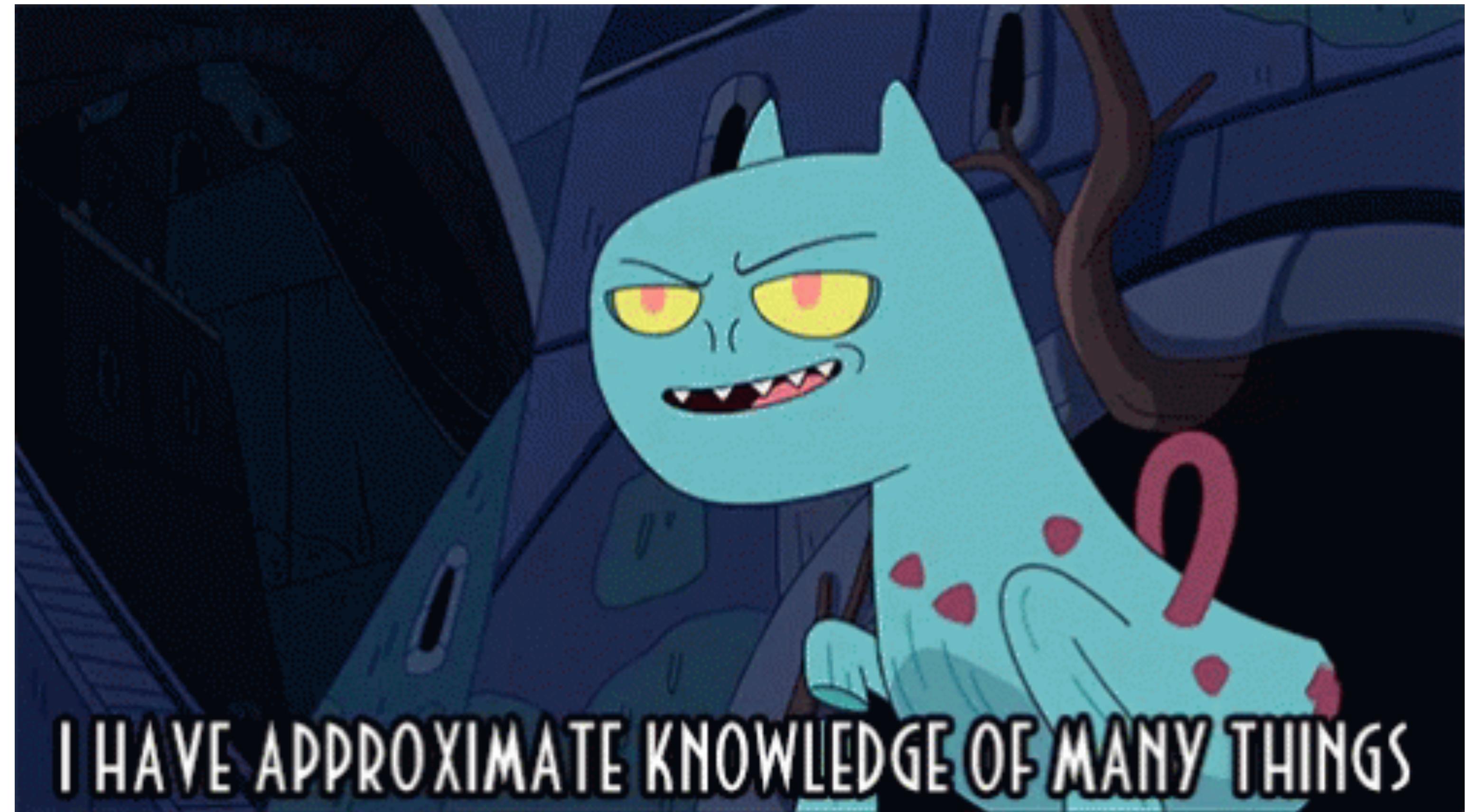
Source: <https://www.interviewmagazine.com/culture/umberto-eco>

Agenda



- **DEFINITION:** What is a "Terminal" (Command Line Interface; CLI)?
- **MOTIVATION:** Why should we use a Terminal and edit a file there?
- **METHODS:**
 - bash
 - Emacs
 - Vi, Vim, Neovim
 - nano & others

Disclaimer: what I say can be slightly inaccurate.



Demon Cat from the episode "Dungeon" of a series *Adventure Time*, (C) Cartoon Network, 2010-2018.

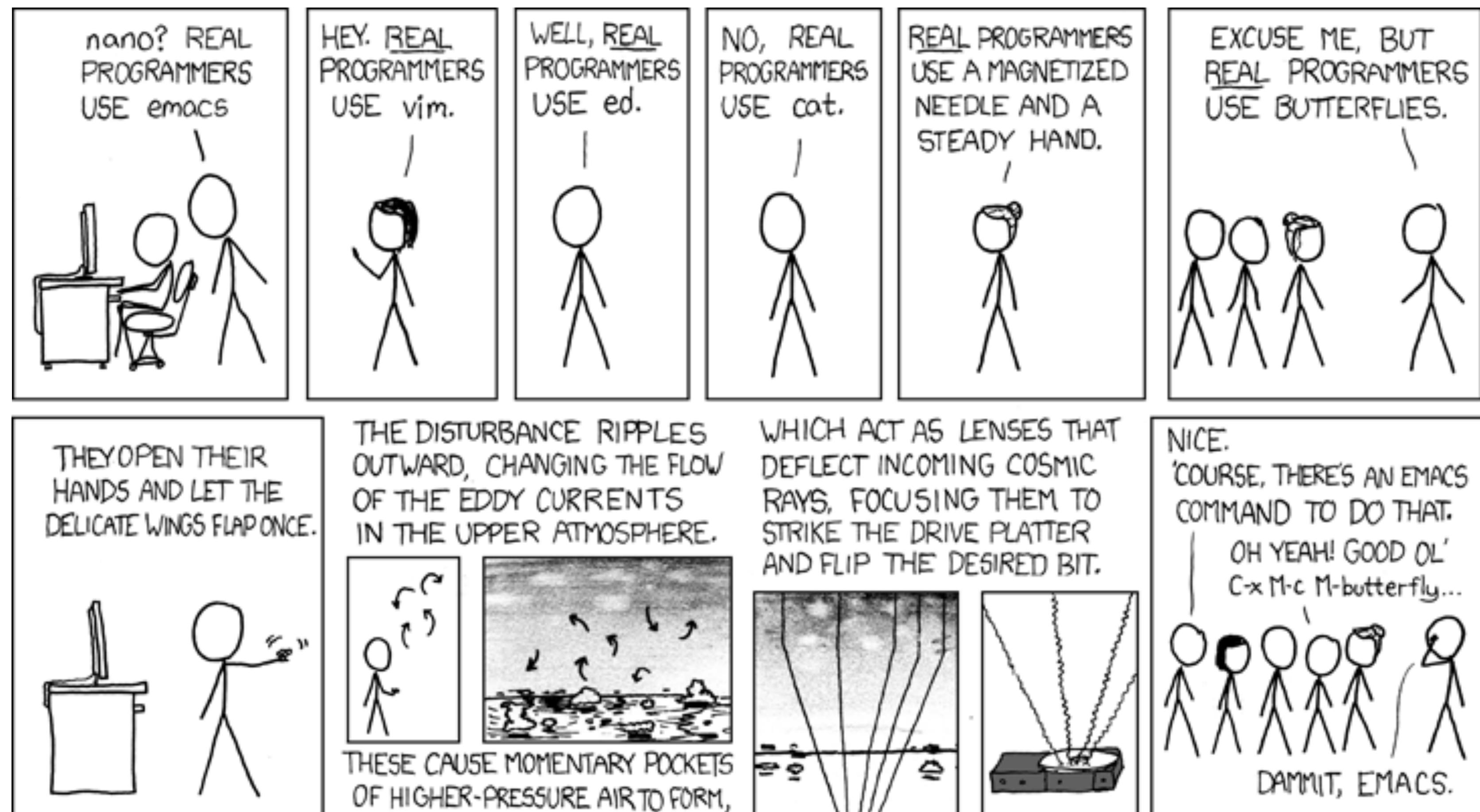
Agenda



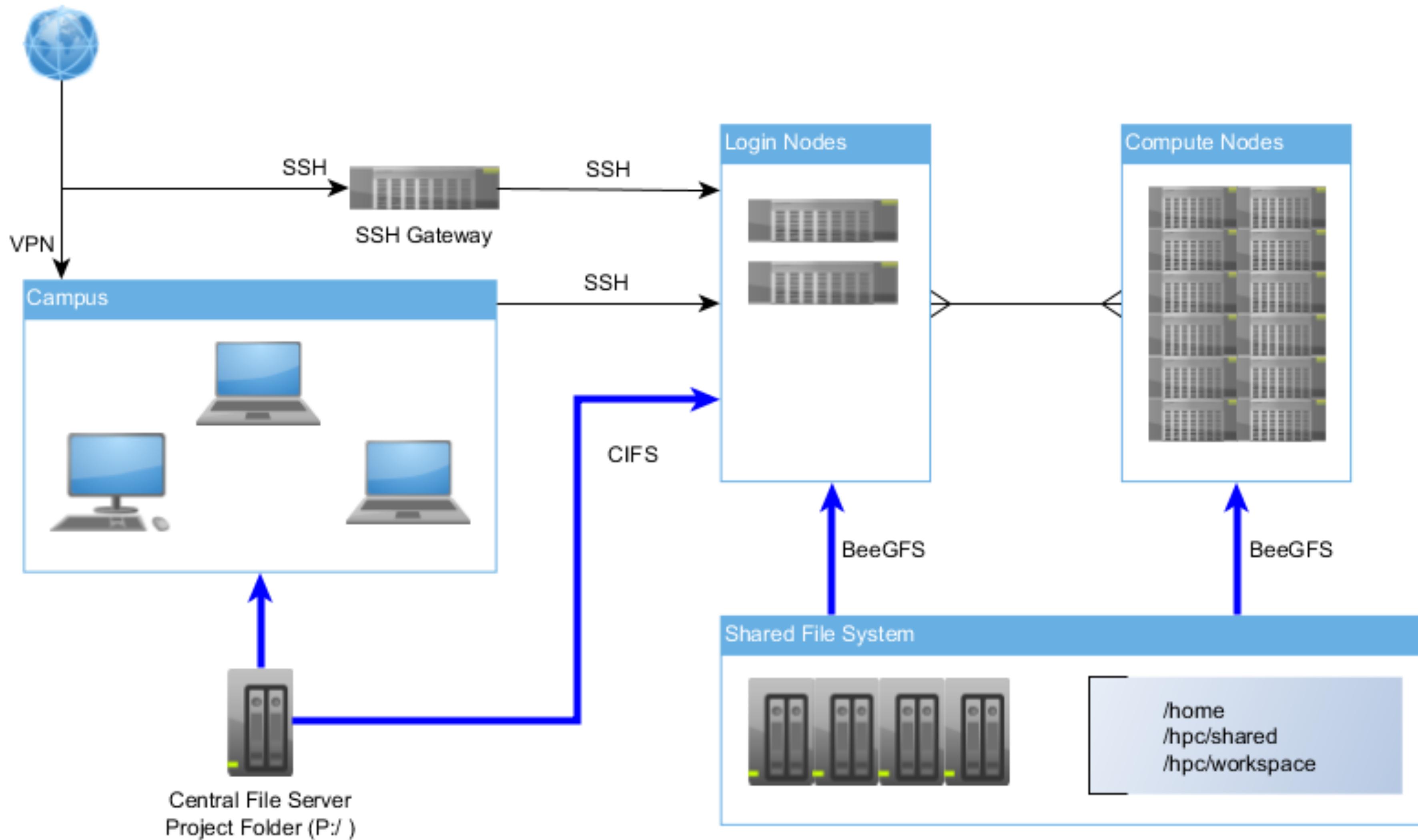
- **DEFINITION:** What is a "Terminal" (Command Line Interface; CLI)?
- **MOTIVATION:** Why should we use a Terminal and edit a file there?
- **METHODS:**
 - bash
 - Emacs
 - Vi, Vim, Neovim
 - nano & others

Because that's the way it's meant to be!

(or to show everyone that you're a REAL programmer)



Or, because we are still using large computers!



How do we use those large computers?

We need to connect to the large computer via network

	Virtual Network Computing (VNC) – GUI	Secure Shell (SSH) – CLI
Pros	<ul style="list-style-type: none">• Easy to learn/explore	<ul style="list-style-type: none">• Scriptable (faster to repeat)• Quick to launch (near-native)
Cons	<ul style="list-style-type: none">• Slower to repeat (need to click a lot; difficult to macro)• Slow to launch a virtual environment	<ul style="list-style-type: none">• Difficult to learn/explore



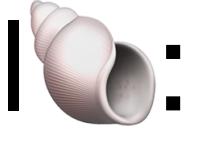
Agenda



- **DEFINITION:** What is a "Terminal" (Command Line Interface; CLI)?
- **MOTIVATION:** Why should we use a Terminal and edit a file there?
- **METHODS:**
 - Bash shell commands
 - Emacs
 - Vi, Vim, Neovim
 - nano & others

A shell? and kernel?

Popular shells

- Shell : An interpreter for humans  to communicate with the core of an OS .
- In fact, shells are **Scripting Languages** like R or Python. You can define variables, call functions, use conditionals, enumerate iterations, ...
- Most widely installed shells:
 - "sh" (Bourne SHell), 1979 [default in Version 7 Unix]
 - "bash" (Bourne-Again SHell), 1989 [default in many Linux]
 - "zsh" (Z shell), 1990 [default in macOS]
 - And "pwsh" (PowerShell), 2006, [default in Windows] but open-sourced and available across OSs

Bash

Bourne-Again SHell



- First release: Brian Fox (1989-06-08, Free Software Foundation)
 - As a free software alternative (GNU project w/ Richard Stallman) for the Bourne SHell (SH)
- Ported to Linux by Linus Torvalds, and widely used as a default shell in various Linux distributions



Brian Jhan Fox (b. 1959)



Bash demo

REDIRECTIONS, ECHO, CAT, SED



Bash demo

conCATenate and print files, Stream EDitor

```
$ echo "X" > file.txt  
$ echo "Y" >> file.txt  
$ cat << EOF > file2.txt  
$ sed 's/oldString/NewString/g' test.txt  
$ sed 's/t[^ ]*xt/TEXT/g' test.txt
```

Agenda



- **DEFINITION:** What is a "Terminal" (Command Line Interface; CLI)?
- **MOTIVATION:** Why should we use a Terminal and edit a file there?
- **METHODS:**
 - bash
 - Emacs
 - Vi, Vim, Neovim
 - nano & others



What is Emacs?

"Emacs is more powerful than any OSs"



GNU Emacs

Editor Macros

- Original EMACS: David A. Moon & Guy L Steele Jr. (1984) MIT AI Lab
- Gosling Emacs: James Gosling (1981) UniPress (sold at 395 \$/copy in 1983)
- GNU Emacs: Richard Stallman (1984) Free Software Foundation



Richard Stallman (b. 1953)
a.k.a. St. IGNUcius, the Church of Emacs

Emacs demo

Open & close, copy & paste, find & replace

Emacs demo

Open & close, copy & paste, find & replace

- ▶ C-x b bufname RET
- ▶ C-x C-c
- ▶ C-a, C-e
- ▶ C-@, M-w, C-y
- ▶ C-x U
- ▶ M-%

Agenda



- **DEFINITION:** What is a "Terminal" (Command Line Interface; CLI)?
- **MOTIVATION:** Why should we use a Terminal and edit a file there?
- **METHODS:**
 - bash
 - Emacs
 - Vi, Vim, Neovim
 - nano & others



What is Vim?

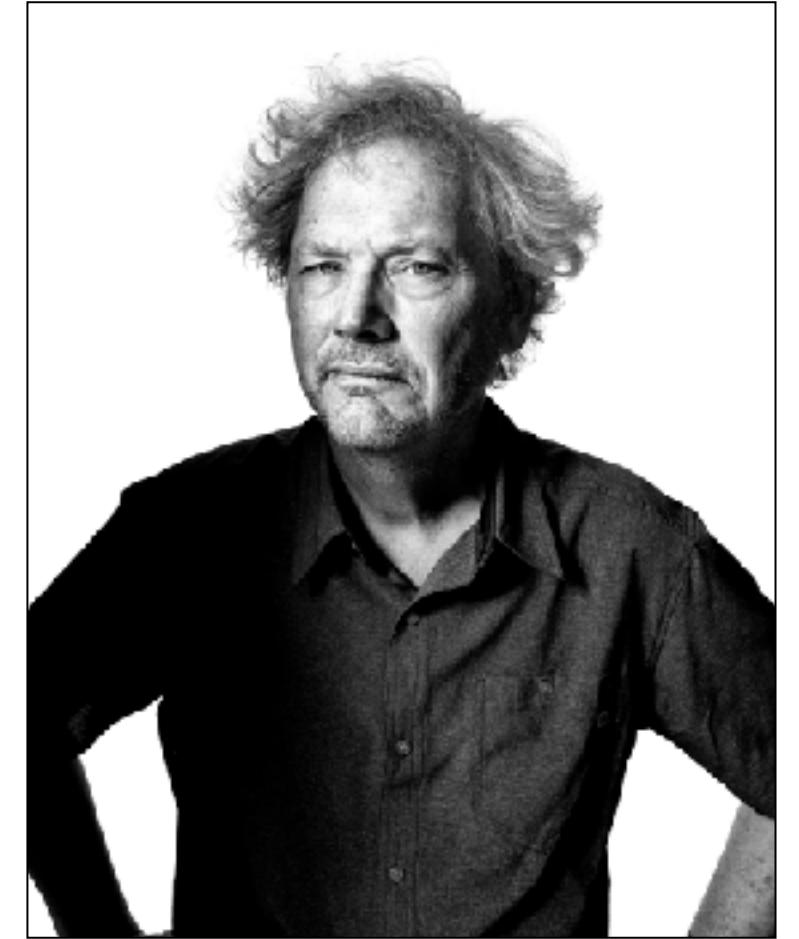
"It will be painful at first and painful at last. Good."



Vi/Vim/neovim

"vi" for visual mode of a line editor called "ex"

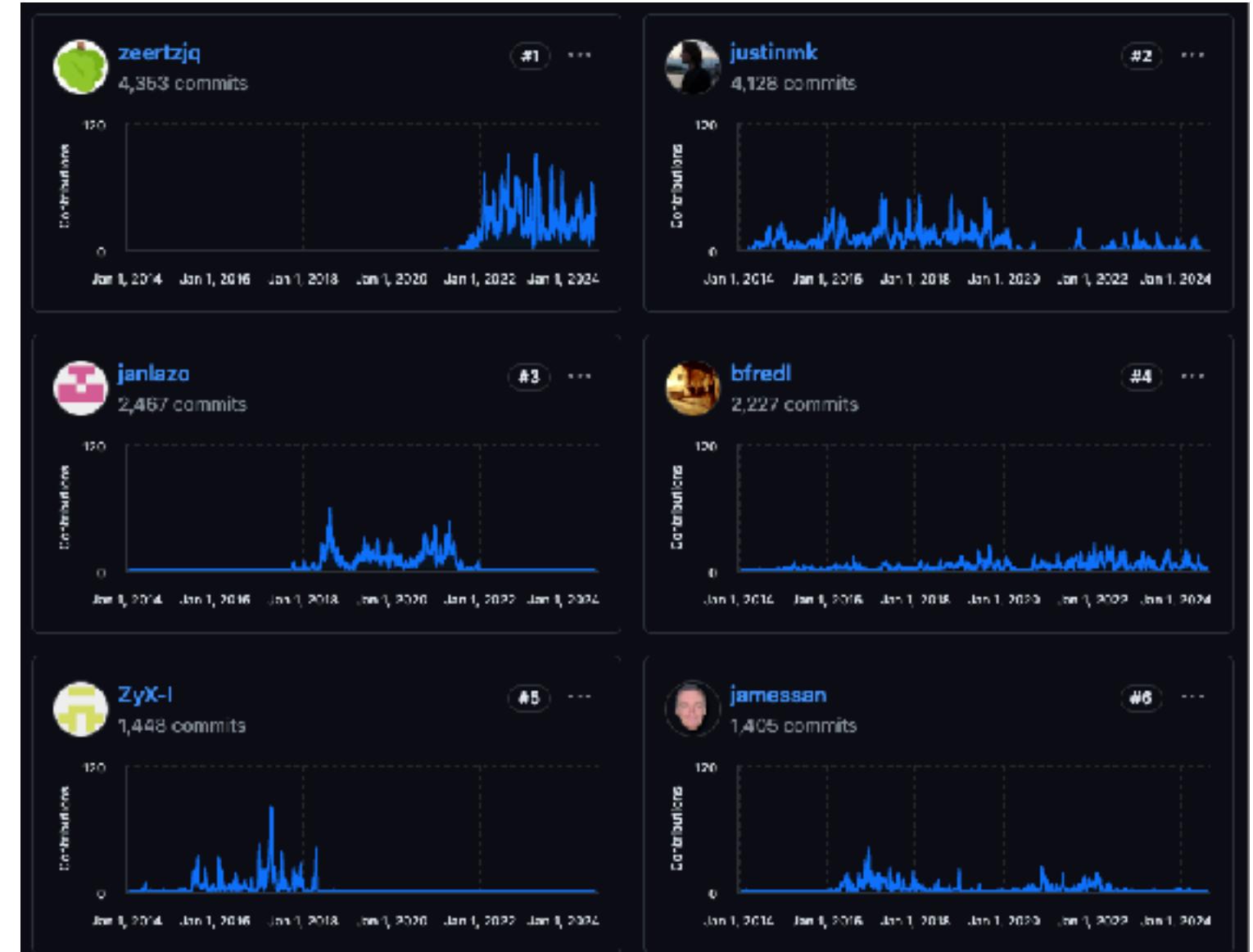
-  /vi:/ : William N. Joy (1976), BSD-licensed
-  /vɪm/ : Bram Moolenaar (1991)
-  /'nɛ.oʊɪm/: 30 core-devs & 1300+ contributors (2014)
- Most popular CLI-editor (Vim: 22.29%, Neovim: 11.88%, Nano: 8.98%, Emacs: 4.69%) in [Stack Overflow survey 2023](#) (for GUI-editor, VS Code: 73.71%, VS: 28.43%; multiple answers)



Bill Joy (b. 1954)



Bram Moolenaar
(1961-2023)

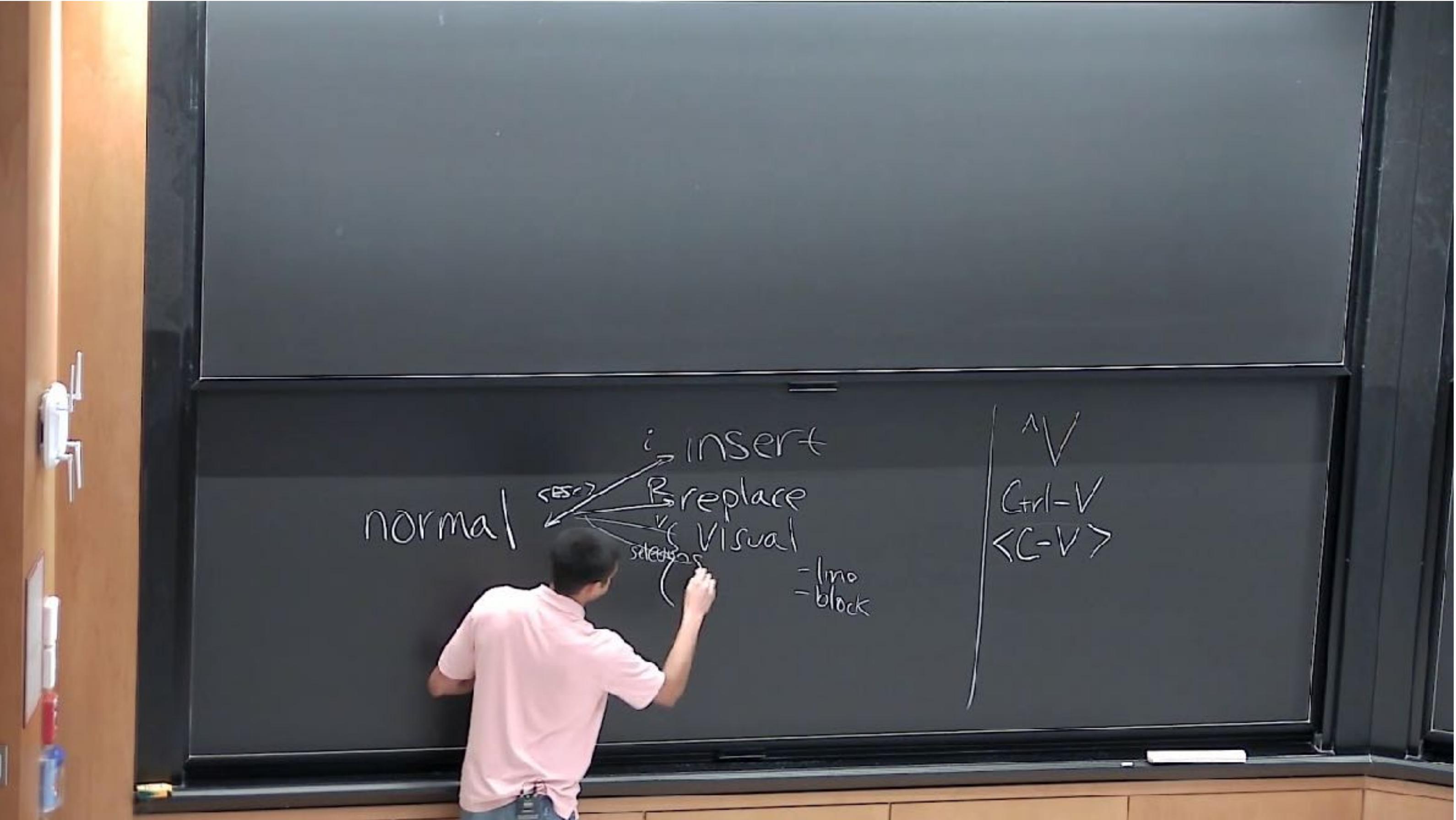


<https://github.com/neovim/neovim/graphs/contributors>



MIT lecture on how to use Vim! 😱

"Vim's interface is a programming language!"



“The Missing Semester of Your CS Education”

<https://missing.csail.mit.edu/>

[./missing-semester](#) | [lectures](#) | [about](#)

[./missing-semester](#) | [lectures](#) | [about](#)

2020 Lectures

- **1/13:** [Course overview + the shell](#)
- **1/14:** [Shell Tools and Scripting](#)
- **1/15:** [Editors \(Vim\)](#)
- **1/16:** [Data Wrangling](#)
- **1/21:** [Command-line Environment](#)
- **1/22:** [Version Control \(Git\)](#)
- **1/23:** [Debugging and Profiling](#)
- **1/27:** [Metaprogramming](#)
(build systems, dependency management, testing, CI)
- **1/28:** [Security and Cryptography](#)
- **1/29:** [Potpourri](#)
- **1/30:** [Q&A](#)

Video recordings of the lectures are available [on YouTube](#).

Why we are teaching this class

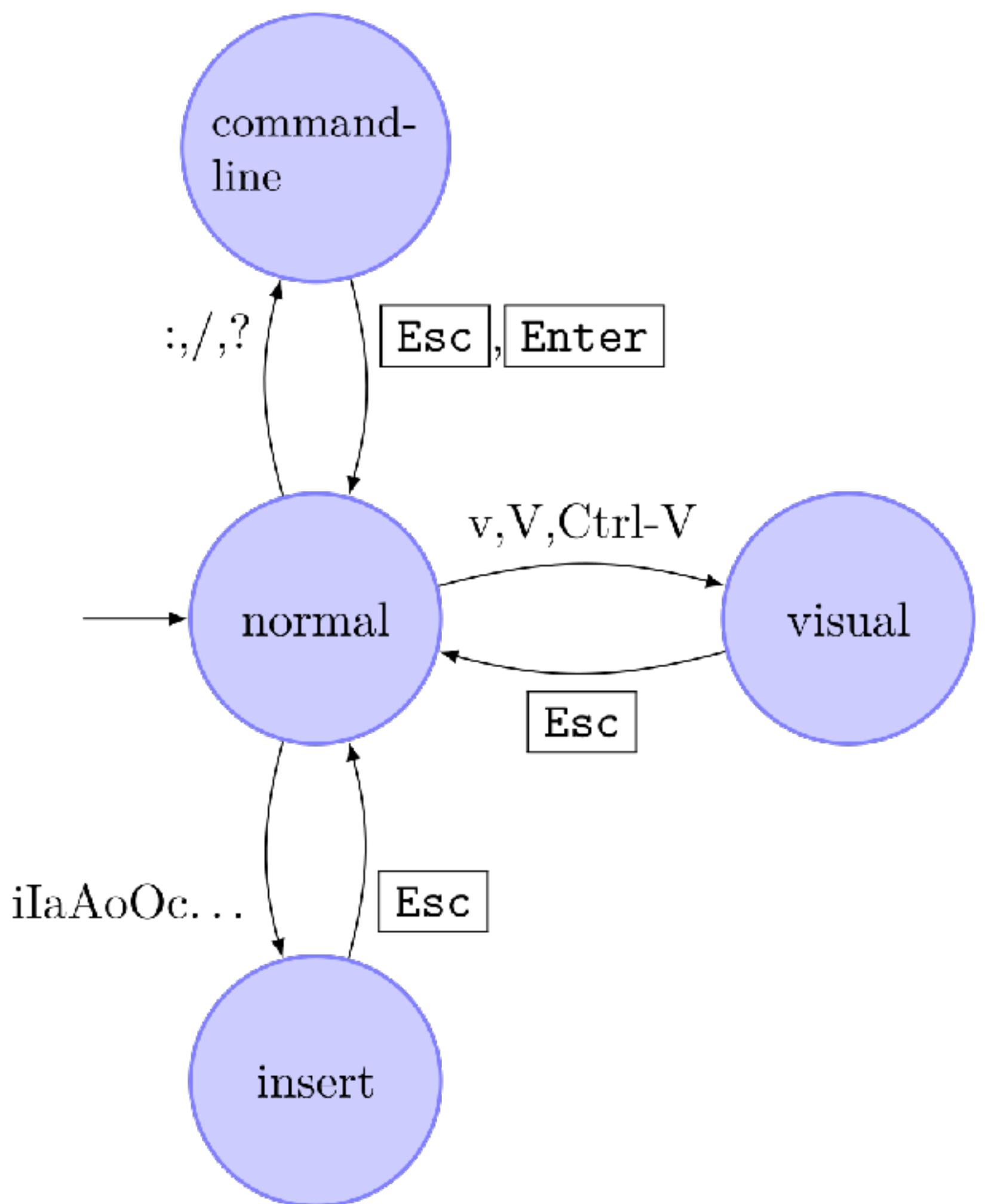
During a traditional Computer Science education, chances are you will take plenty of classes that teach you advanced topics within CS, everything from Operating Systems to Programming Languages to Machine Learning. But at many institutions there is one essential topic that is rarely covered and is instead left for students to pick up on their own: computing ecosystem literacy.

Over the years, we have helped teach several classes at MIT, and over and over we have seen that many students have limited knowledge of the tools available to them. Computers were built to automate manual tasks, yet students often perform repetitive tasks by hand or fail to take full advantage of powerful tools such as version control and text editors. In the best case, this results in inefficiencies and wasted time; in the worst case, it results in issues like data loss or inability to complete certain tasks.

These topics are not taught as part of the university curriculum: students are never shown how to use these tools, or at least not how to use them efficiently, and thus waste time and effort on tasks that *should* be simple. The standard CS curriculum is missing critical topics about the computing ecosystem that could make students' lives significantly easier.

Vim demo

open & close, copy & paste, find & replace



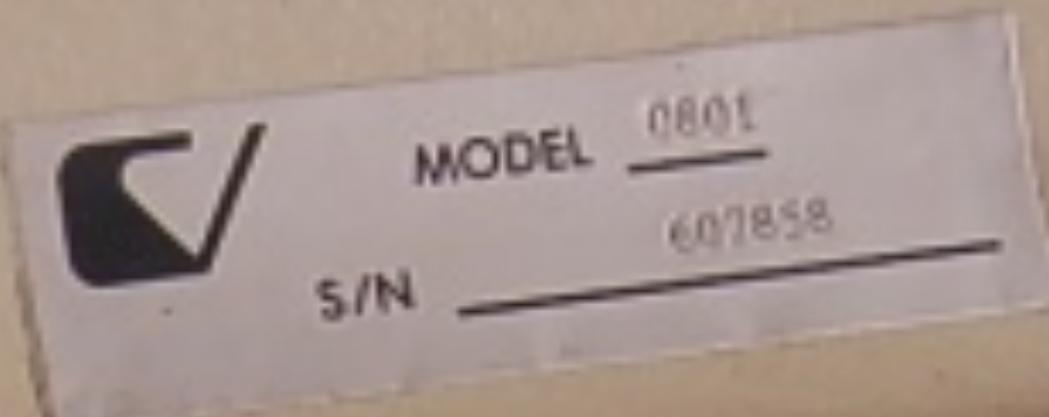
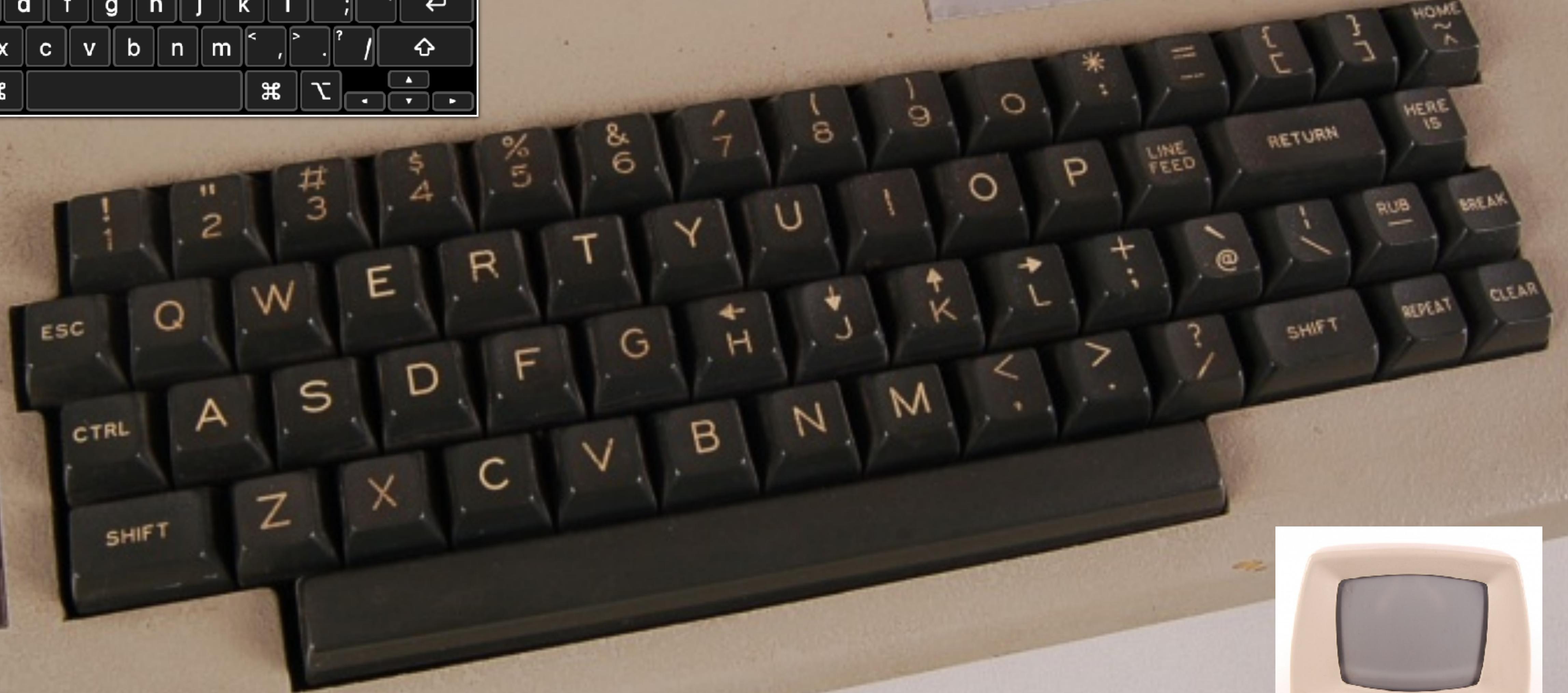
Vim demo

Normal mode commands

General: u, C-r, .

Motions: h j k l, w b e, } {, 0 ^ \$, H M L, C-u C-d, C-f C-b, G g g, [number]G, f [char], F [char], [count]l, [count]b, %, /[pattern], ?[pattern]

Edits: d[motion], c[motion], i I, o O, a A, yy, y [count]y, y[motion], p, [count]p



Vim demo

How you talk to Vim: Composability of Vim's syntax

[edit]..[motion]: dw

[edit]..[[count]×[motion]]: d3w

[edit]..[object]: das

[count]×[motion|edit|general]: 3w, 2p, 100.

[count]×[[edit]..[motion]]: 3dw

Vim demo

Command mode

/[pattern] AND n OR N

:noh

:1,10s/this/That/gc

:%s/this/That/g

:sp OR :vsp AND C+w w

:tabedit OR tabnew
AND gt OR gT

:edit [filename]

:ls

:[number]RET

:qa!

:wq

Agenda



- **DEFINITION:** What is a "Terminal" (Command Line Interface; CLI)?
- **MOTIVATION:** Why should we use a Terminal and edit a file there?
- **METHODS:**
 - bash
 - Emacs
 - Vi, Vim, Neovim
 - nano & others



cli text editors



All Images Videos News Web Books Finance

Tools

Default

Windows

Best

Text editor Software / Command-line interface

From sources across the web



Vim

GNU General Public License



Emacs

GNU General Public License



GNU nano

GNU General Public License



Neovim

Apache License 2.0



vi

BSD licenses



gedit

GNU General Public License



ne

GNU General Public License



ed

MIT License



Kakoune

Unlicense



NEdit

GNU General Public License



JED

GNU General Public License



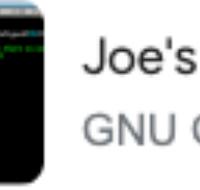
Leafpad

GNU General Public License



MS-DOS Editor

Proprietary software



Joe's Own Editor

GNU General Public License



Edinburgh Compatible C...

BSD licenses



XEDIT



ex



Pico

Apache License



BBEdit Lite

Freeware



Nano demo

Cursor movement, copy & paste, find & replace

How to set up nano as a default editor for git?

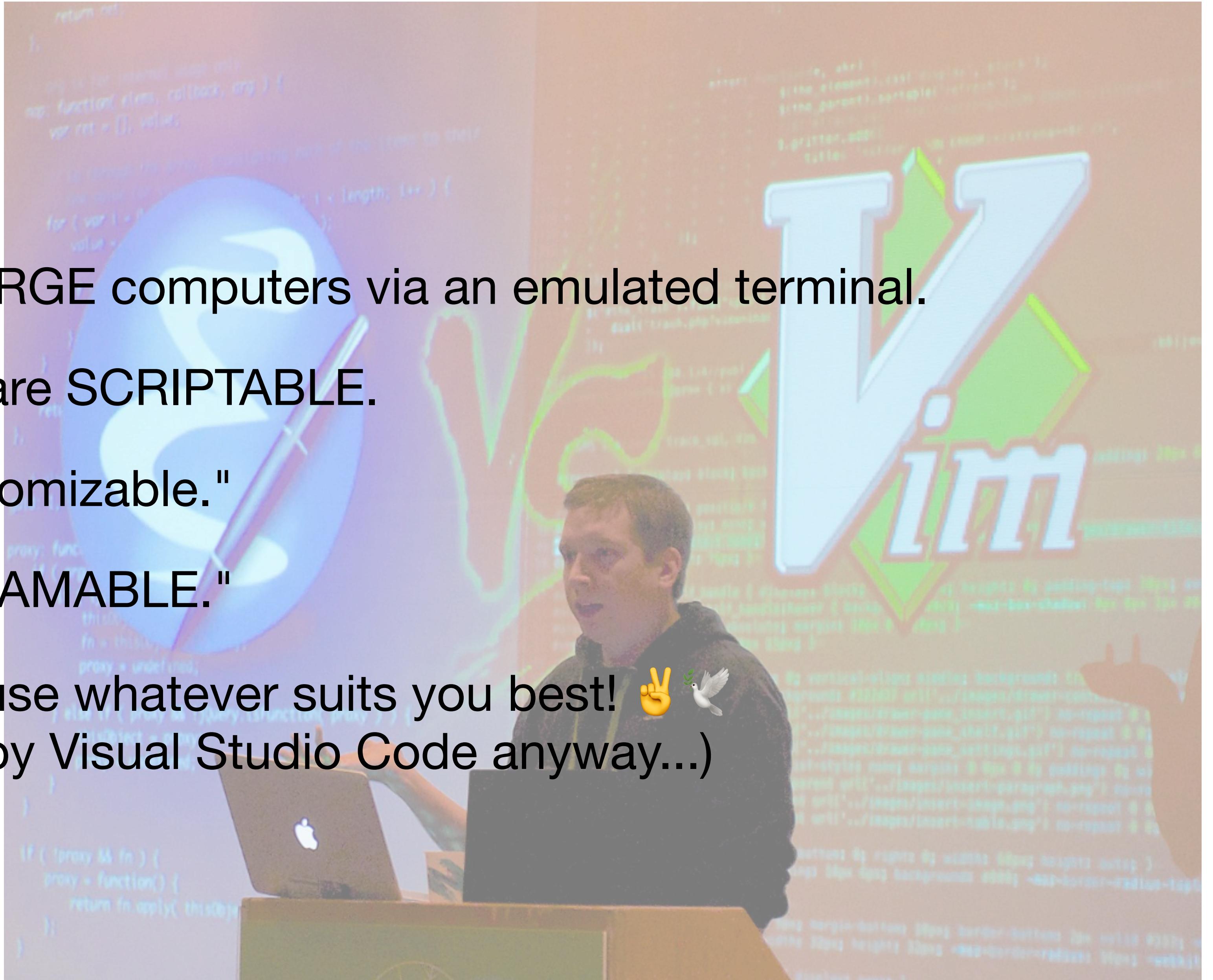
If you don't like Vim... 😢

```
$ git config --global core.editor nano
```

Summary



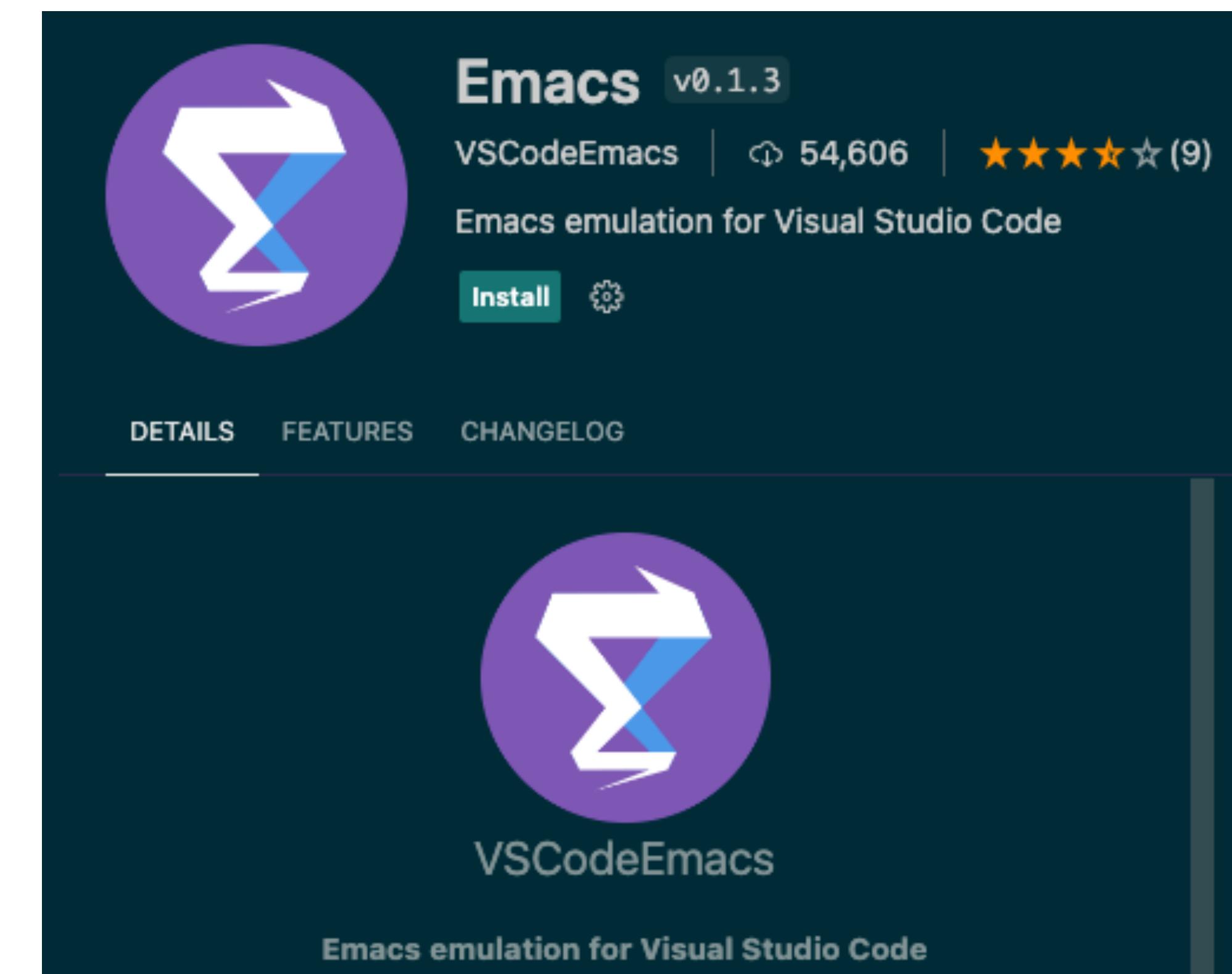
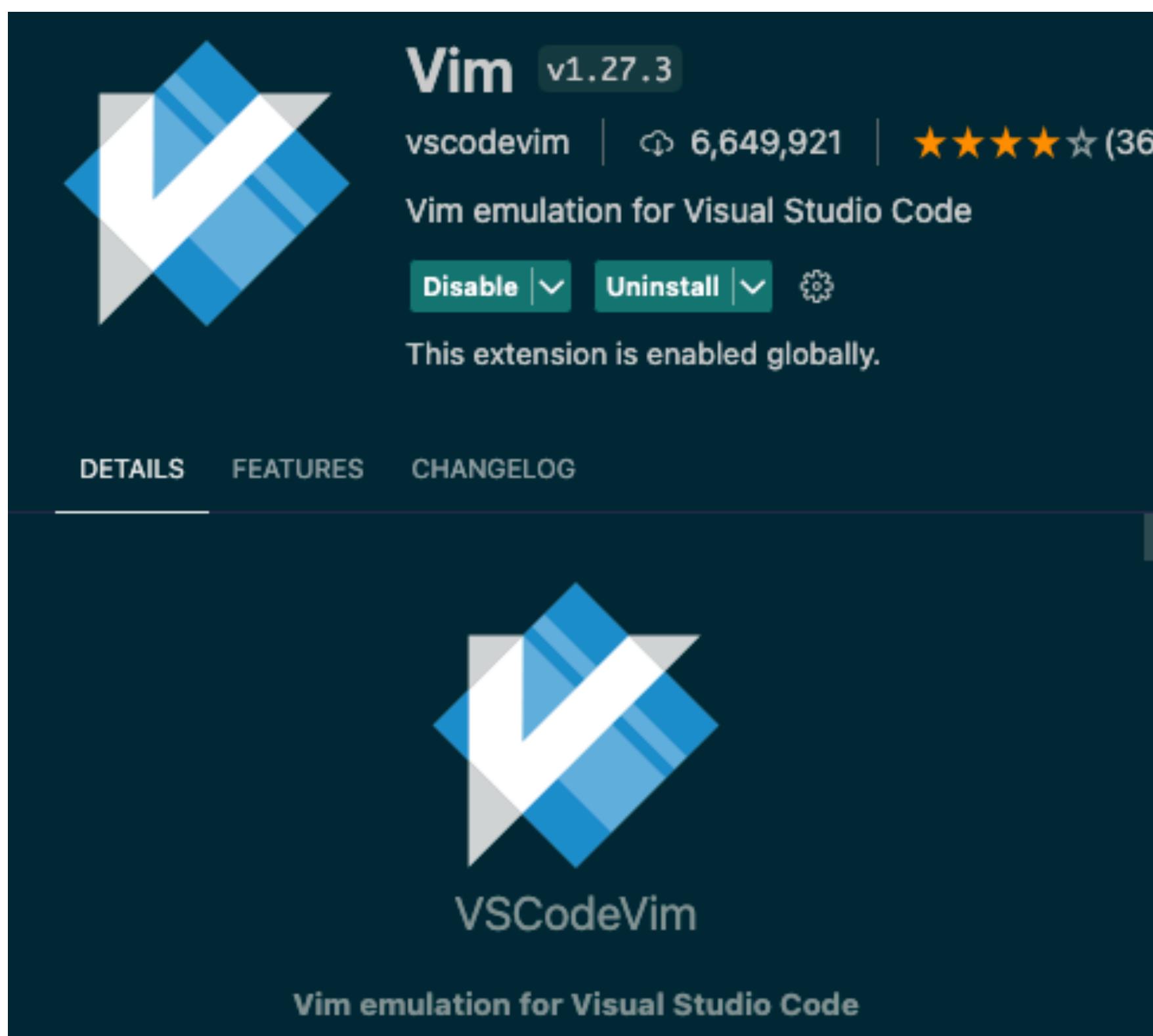
- We are still using those **LARGE** computers via an emulated terminal.
- Command-line interfaces are **SCRIPTABLE**.
- "Emacs is completely customizable."
- "Vim's interface is **PROGRAMABLE**."
- But of course you should use whatever suits you best! 🤝🕊️
(The Holy War has ended by Visual Studio Code anyway...)



Summary



- (Or is the War back?)



Discussion

Personal thoughts 🤔

- Do CLI-text editors replace IDE?
- Do CLI-text editors really enhance productivity?



(n)vi(m) and/or Emacs as an IDE?

The screenshot shows a terminal window with several tabs open, each displaying code snippets. The tabs include `utils.js`, `smooth.css`, `todoBtns.jsx`, `api.js`, and `animations.css`. The left pane shows a file tree with files like `assets`, `config`, `src` (containing `components`, `css`, `api.js`, `App.jsx`, `index.jsx`, `land.vue`, `store.jsx`, `utils.js`, `utils.jsx`), `static`, `babel.config.js`, `index.html`, `LICENSE`, `package.json`, `README.md`, `vercel.json`, and `vite.config.js`.

The right pane displays the code for `utils.js` and `api.js`. The bottom right pane shows the code for `animations.css`, featuring keyframes for `smoothPopup` and `slideIn`.

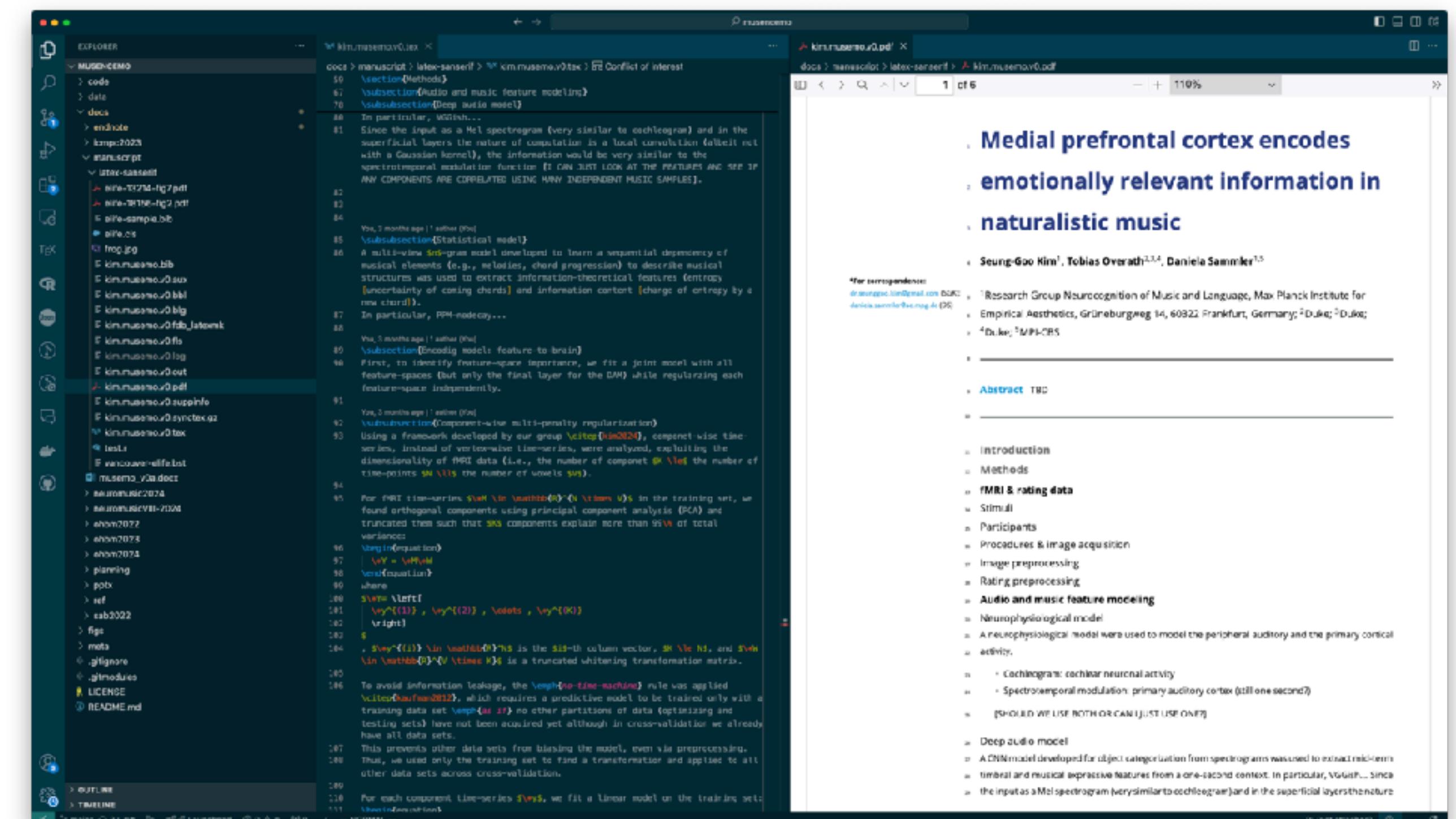
A floating terminal window in the bottom right corner shows the output of the command `astro dev`, indicating that `astro` v1.0.0-beta.53 has started in 27ms, and provides links to the local host at `http://localhost:3000/` and the feedback page at `https://astro.build/issues`.

The status bar at the bottom shows the current mode as `INSERT`, the file name as `utils.js`, and other status indicators.

IDE (Integrated development environment)?

Interpreter/compiler + text editor (+ shell + file browser + ... + AI) 

- Visual Studio for C++ and .NET
- Visual Studio Code for JavaScript/HTML, Python, ..., and LaTex!
- IntelliJ IDEA for Java
- PyCharm, Spyder, IDLE for Python
- MATLAB for MATLAB
- RStudio for R





But, is it just a nerdy joke or else?

e.g., "Chuck Norris used negative one keystroke to write the entire OS"

- Obviously it's very fun.
- Or does saving a few milliseconds make really a difference?
- So far, no controlled user-test data available... 😢

 **VimGolf**

Real Vim ninjas count **every** keystroke - do you?
Pick a challenge, fire up Vim, and show us what you got.

[Changelog](#), [Rules & FAQ](#), updates: [@vimgolf](#), [RSS](#).

Rearrange array to single level

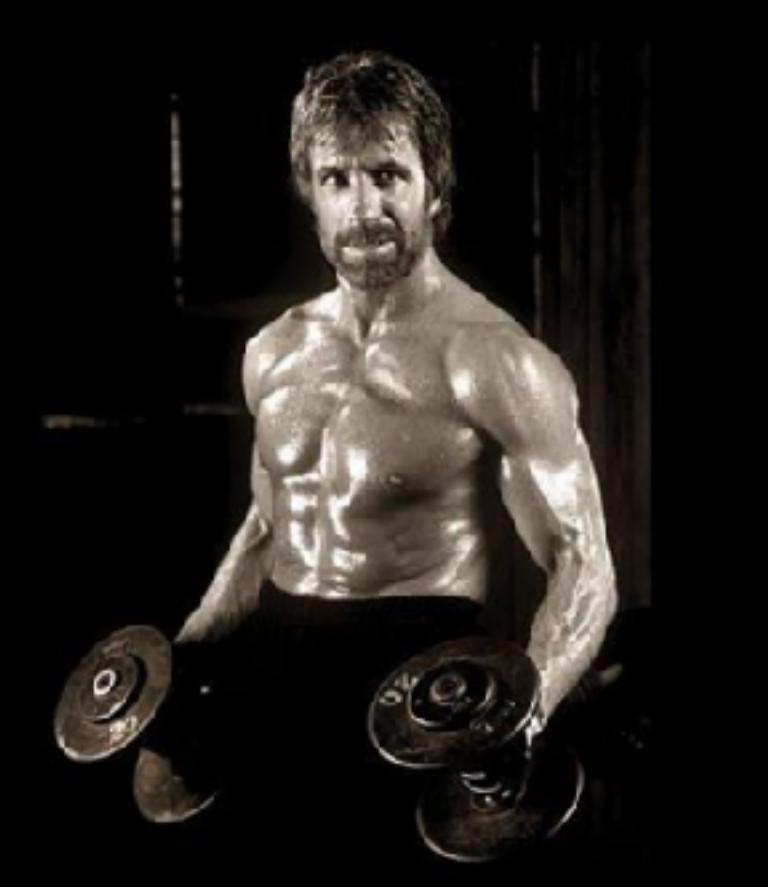
The goal is to flatten the array into a single list and remove any empty elements.

Start file

```
[  
  [],  
  [1],  
  [0, 1, 2],  
  [3, 4, 5],  
  [6, 7],  
  [8, 9],  
  []]
```

End file

```
[1, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
```



Chuck Norris, from his Instagram (@chucknorris)

Leaderboard (*lowest score wins*):

 	#1 - Peppa Pig / @PeppaPi95550250 07/22/2024 at 12:45AM	16
 	#2 - John Braxler / @braxler 07/22/2024 at 01:07PM	16
 	#3 - Alex Lewin / @_AlexLewin 07/23/2024 at 09:49PM	16
 	#4 - Danilo J. S. Bellini 🇧🇷 / @danilobellini 07/31/2024 at 12:58AM	16

I found one from Google Scholar!

de Oliveira, B. C., & Zuchi, J. D. (2020). Efficiency in Writing Software With Vim. *Revista Interface Tecnológica*, 17(2), 386-397. <https://doi.org/10.31510/infa.v17i2.1066>

- Only one study from Brazil... violating all Fisher's principles – randomization, replication, orthogonality with only 5 subjects... 😞 😞

Before

```
int page_number = 44;
int query = 9;
int id = 55;
int last_page_number = 12;
int foo = 1;
int bar = 89;
int fizz = 11;
int buzz = 392;
int foo_ext_number = 31;
int title_id = 31;
```



After

```
int page_number = 1;
int query = 2;
int id = 3;
int last_page_number = 4;
int foo = 5;
int bar = 6;
int fizz = 7;
int buzz = 8;
int foo_ext_number = 9;
int title_id = 10;
```

Source: Author (2020).

After the subjects performed the scenario above, the results were as follows:

Figure 6 - Test case results.

Subject	Text Editor	Elapsed Time (seconds)
Subject 1	VS Code	10s
Subject 2	VS Code	9s
Subject 3	Vim	8s
Subject 4	VS Code	32s
Subject 5	VS Code	12s

Source: Author (2020).



Perhaps? LaTex vs. MS Word

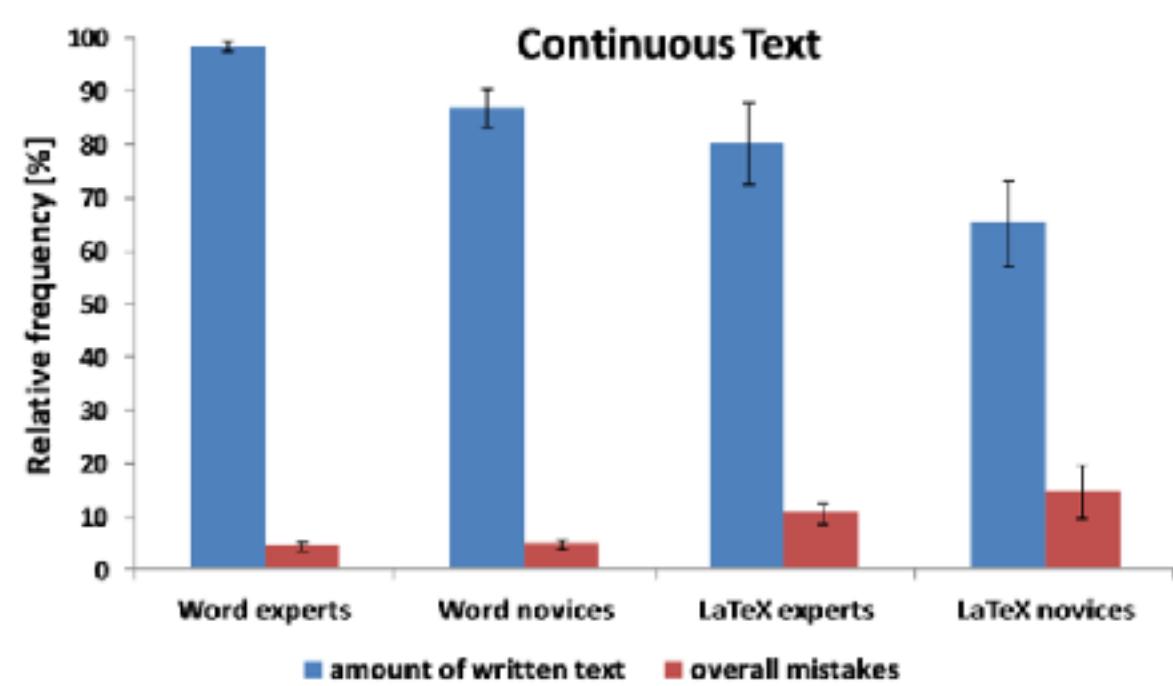


Fig 4. Mean amount of text written within 30 minutes and the overall number of mistakes for the continuous text for the four groups of participants (Word experts, Word novices, LaTeX experts, and LaTeX novices). Error bars represent the standard error.

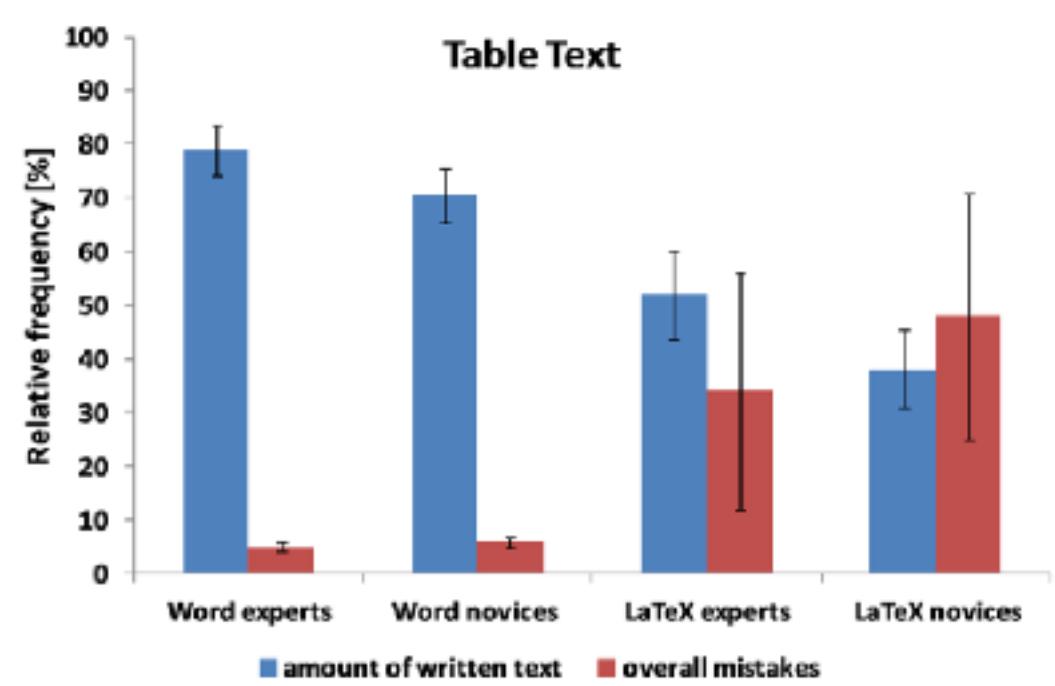


Fig 5. Mean amount of text written within 30 minutes and the overall number of mistakes for the table text for the four groups of participants (Word experts, Word novices, LaTeX experts, and LaTeX novices). Error bars represent the standard error.

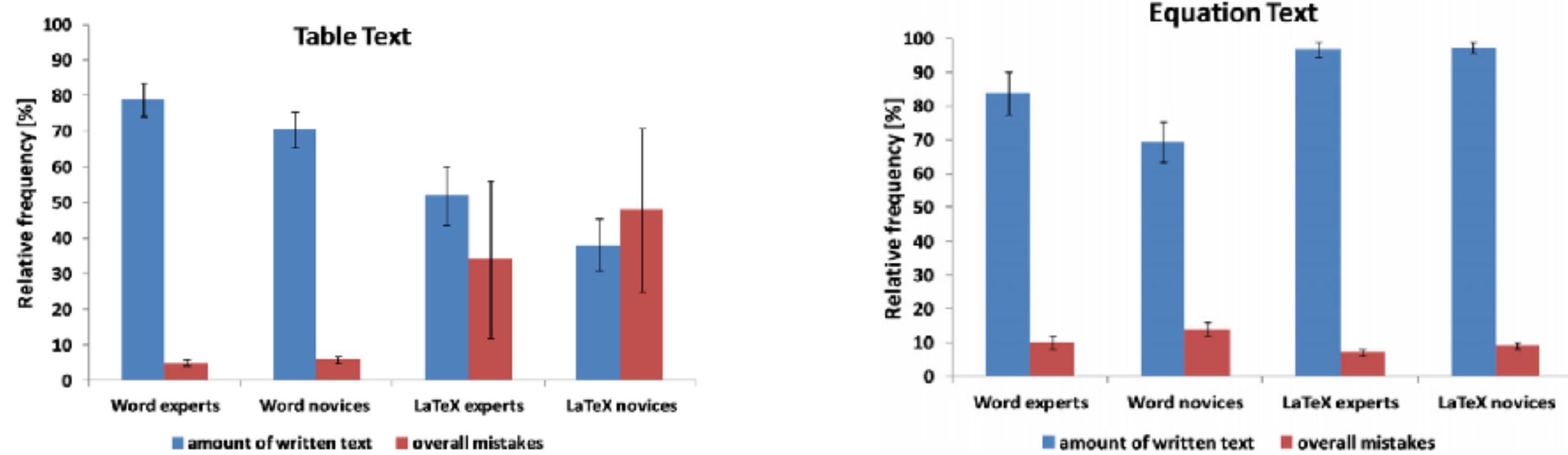


Fig 6. Mean amount of text written within 30 minutes and the overall number of mistakes for the equation text for the four groups of participants (Word experts, Word novices, LaTeX experts, and LaTeX novices). Error bars represent the standard error.

Table 3. Results from the usability questionnaire ISO 9241–10.

	Software	
	Word	LaTeX
Usability questionnaire		
Tiredness	3.4	1.9
Frustration	3.3	2.0
Enjoyment	3.6	1.7
Suitability for the task	0.6	1.1
Self-descriptiveness	-0.2	0.9
Controllability	1.6	1.0
Conformity with user expectations	1.3	0.7
Error tolerance	0.3	1.1
Suitability for individualization	0.2	1.1
Suitability for learning	0.4	1.1

doi:10.1371/journal.pone.0115069.t003

$$= \frac{-1}{a(\phi_c)} p_c(x(j)) \quad (64)$$

N = 10 / group
"Novices" <500 hr
"Experts" >1000 hrs

14 females, 26 males
Physics: 12, Psychology: 5,
Computer Science: 4, ...

<https://github.com/seunggookim/clied>



Slides PDF & text samples

Time for (more) questions
& discussion! 😎