Vim in your IDE

Our mouse is the essential tool for manipulating things on our computer, and our keyboard lets us input text. One advantage of the mouse is that it has a low learning curve: the screen shows us what we can do, and the mouse lets us do those things.

VIM provides a secondary tool for manipulating things on our computer. It's faster and more powerful than the mouse, and unlike an LLM its inputs are terse and latency free. If you'd like to learn the commands, you can more effectively manipulate text than you could with a mouse and keyboard.

In this session, we learn to:

- use the VIM extension in VS Code to move around code faster than we could with a mouse
  - toggle between VIM mode (VIM NORMAL) and regular IDE mode (VIM INSERT)
  - translate high level navigation (like "go to line 35 ½" into VIM command) to quickly drive to the place you're navigated
  -

Fill in one or more blanks

In Vim, I know how to __

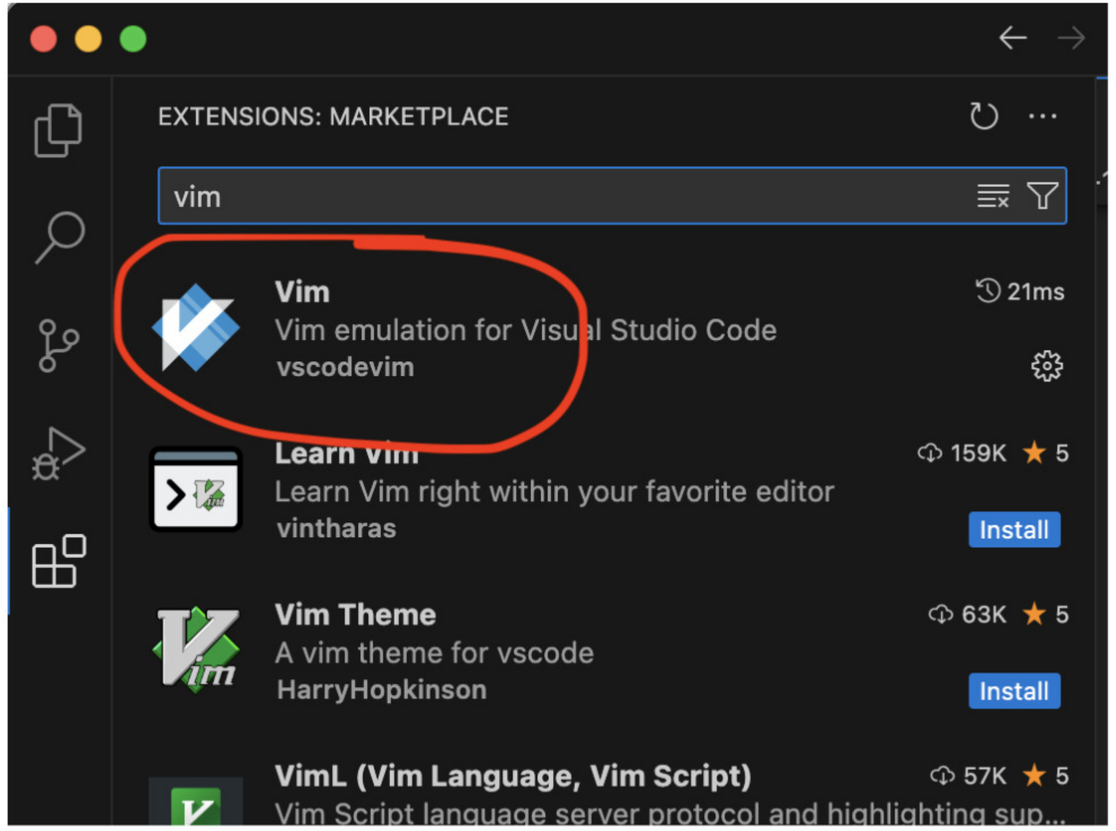
I have avoided Vim because __

Vim is interesting to me because __

Place a card with your name on it

Setup your laptop/workstation for the exercises

1. Install VS Code
2. Install Vim Extension in VS Code
    a. Ctrl+Shift+P
    b. Extensions: Install Extensions
    c. search: vim
3. Download and open desserts.ts



Vim vs Mouse

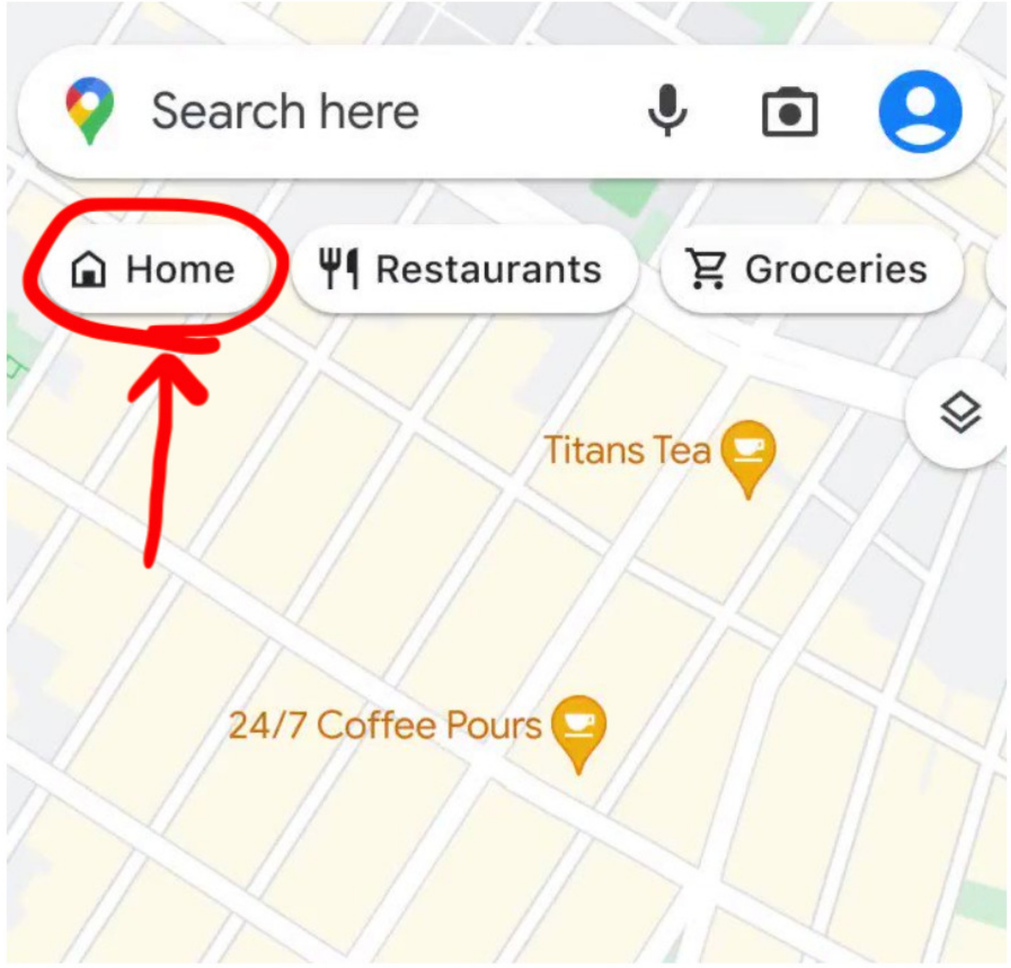
- intuitive
- good for discovering new actions
- more open ended

- within it's scope, it's faster
- gestures are more repeatable

```
VIM - Vi IMproved

version 9.0.1544
by Bram Moolenaar et al.
Vim is open source and freely distributable
```

How do we go "home"?



Before setting out on an adventure, we make sure we'll have a way to get back home

NORMAL = new to us; INSERT = familiar

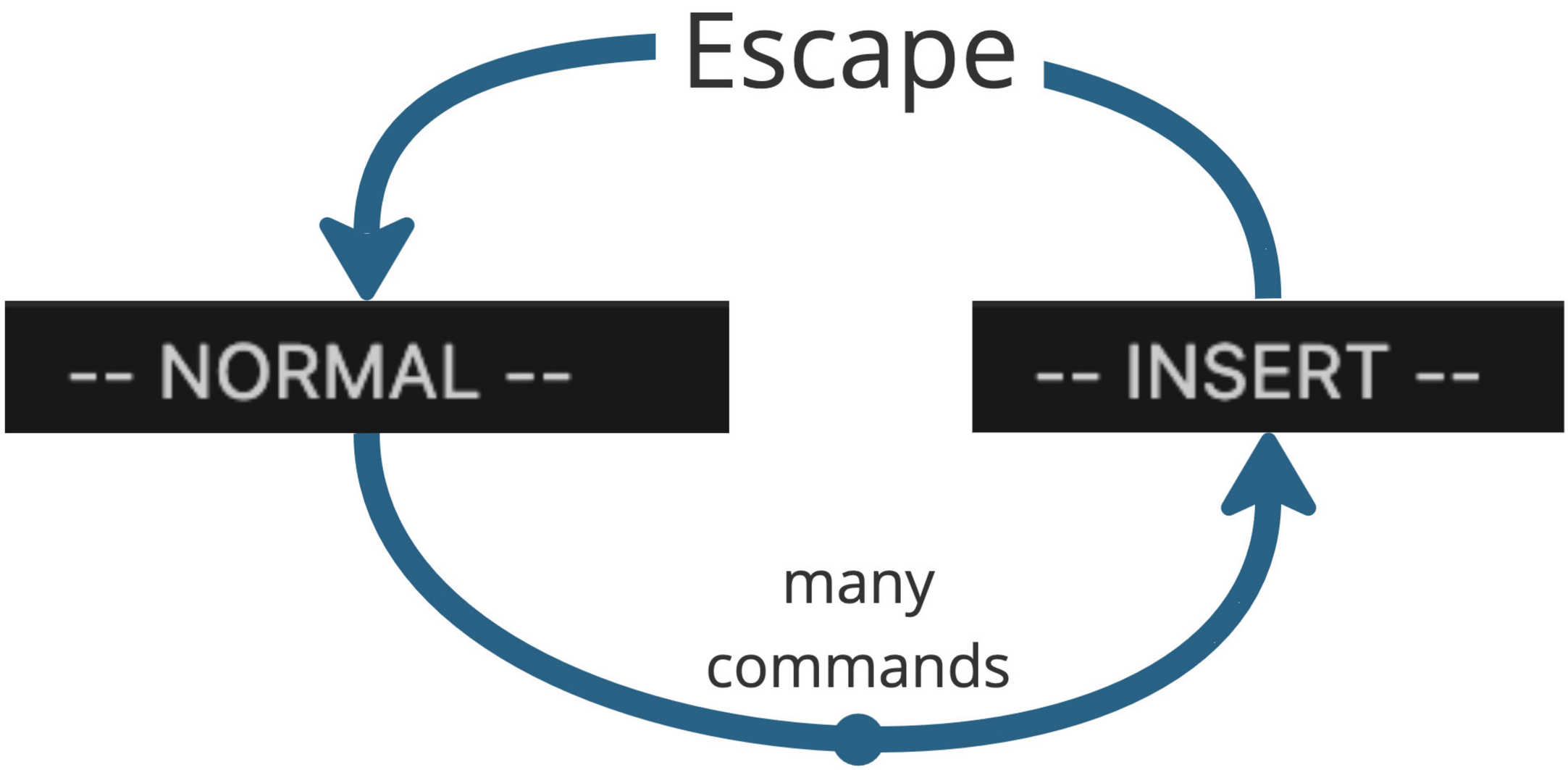
- this is the default mode in VIM
- our keyboard becomes a navigation device (a "second mouse")
- the new stuff we'll learn today takes place in Normal mode

-- NORMAL --

- in Insert mode, our IDE behaves the way we're used to
- in this mode, VS Code should behave in the way we're used to

-- INSERT --

Changing Modes



Changing Modes

Commands for entering Insert Mode

hidden

behind the cursor
at the start of the current line
after the cursor
at the end of the current line
on a new line after the cursor
on a new line before the cursor

In pairs, match the command with the behavior.

Use Escape to go back to Normal mode after each command.

Use your arrow keys or your mouse to move around the code

Changing Modes

Commands for entering Insert Mode

`o` `O` `i` `I` `a` `A`

- start writing behind the cursor
- start writing at the start of the current line
- start writing after the cursor
- start writing at the end of the current line
- start writing on a new line after the cursor
- start writing on a new line before the cursor

In pairs, match the command with the behavior.

Use Escape to go back to Normal mode after each command.

Use your arrow keys or your mouse to move around the code

Moving Around

Note: <cr> means Enter

VIM	Navigation
`G`	go to the bottom of the file
`gg`	go to the top of the file
`:33<cr>o`	on line 33½
`:12<cr>0`	just before line 12
`` `` ```	go back to where we were before
`:22<cr>A`	at the end of line 22
`:23<cr>I`	at the start of line 23
`/blue<cr>`	go to the next `blue`. Press `n` to skip to subsequent matches
`gg/fish =<cr>`	go to where we initialize `fish`
`gg/class <cr>`	go to the start of the class
`e`	after this word
`b`	before this word
`f)i`	just inside the close parenthesis (try this on the `Math.floor` line)

In pairs, take turns navigating each other through each of these commands.

After one person has tried each of these commands, switch roles and repeat.

Navigators: first, give the high level navigation. Then, if the driver needs it, give the low level VIM navigation.

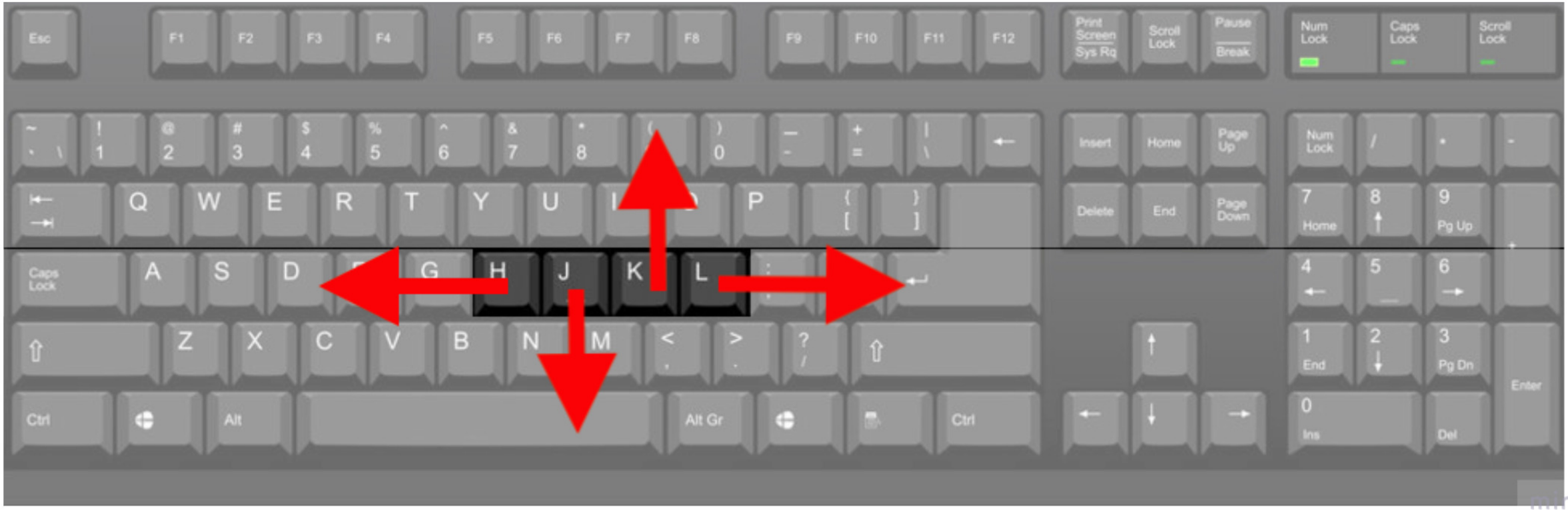
If you have extra time, repeat the exercise until the navigator can give only the high level navigation

Beyond Arrow Keys

Arrow keys are closer than the mouse, but still require moving away from our home row typing position.

Arrows from Home Row

Use h, j, k, and l to move the cursor
left, down, up, and right



Jump around

Commands to jump to a nearby word or letter

- **/word<cr>** to jump to the next occurrence of word
- **/ord<cr>** if you don't know if W is capitalized
- **fx** to jump to the next **x** character on the line
- **e** to jump to the end of the current word
- **b** to jump to the beginning of the current word
- **%** to jump to the matching brace, bracket, or paren

Repeat Commands

Put a number before a command to do it **n** times

- **5j** to go down 5 lines
- **3e** to go the the end of the 3rd word
- likewise for h, k, l, b, and others

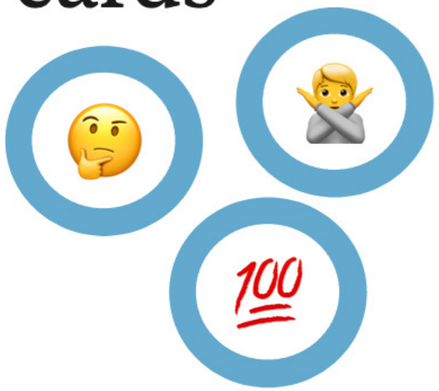
Vim Tag

In pairs, chase each other around dessert.js

1. Choose someone to be "it"
2. The person who is "it" describes a place in the file
3. The other person must get there as quickly as possible and insert *HERE* at the place that's described
4. If the *HERE* is in the right spot, then swap who is "it" and repeat

Reflection

1. Reflect on this training by adding stickies to the cards
2. Clone one of these emoji and place them on the statement below to show your reaction.



I could start using Vim in VS Code today if I wanted to

I'm keen to start using...

I don't want to forget...

I want to learn more about...