Introduction to Git and Debugger

FoA 2025 Week 7

Git

- Created in 2005 by Linus Torvalds — the creator of Linux
- Used to manage the Linux kernel’s source code
- Before Git:
    - Bitkeeper, CVS, Subversion
    - Centralized
    - Slow
    - Limited branching
    - Merging ofter result in conflicts
    - Not Open sourced
    - ……

Git

- Distributed
- Fast
- Branching
- Open source

Git Basics: Key Features

- Initializing repository: `git init`
- Commits : `git commit -m “some message”`
- Branching: `git branch feature-x >>> git checkout feature-x`
- Merging : `git checkout main >>> git merge feature-x`
- History : `git log`
- Diffs : `git diff`
- Undo : `git checkout file.c`
- Revert : `git revert <commit-id>`

DEMO: Feel free to do this with me

.gitignore

- A special file that tells Git which files or folders to ignore (i.e. not track)
- You dont need to write it yourself
  - https://www.toptal.com/developers/gitignore

```
*.o
a.out
*.log
.vscode/
build/
```

Working with Remote Repos

```
• git clone <url>    # Download a copy of a remote repo

• git remote -v      # Show linked remotes (e.g. origin)

• git push           # Upload commits to remote

• git pull           # Download + merge latest commits

• git fetch          # Download latest commits only

                       (no merge)
```

Working with Remote Repos

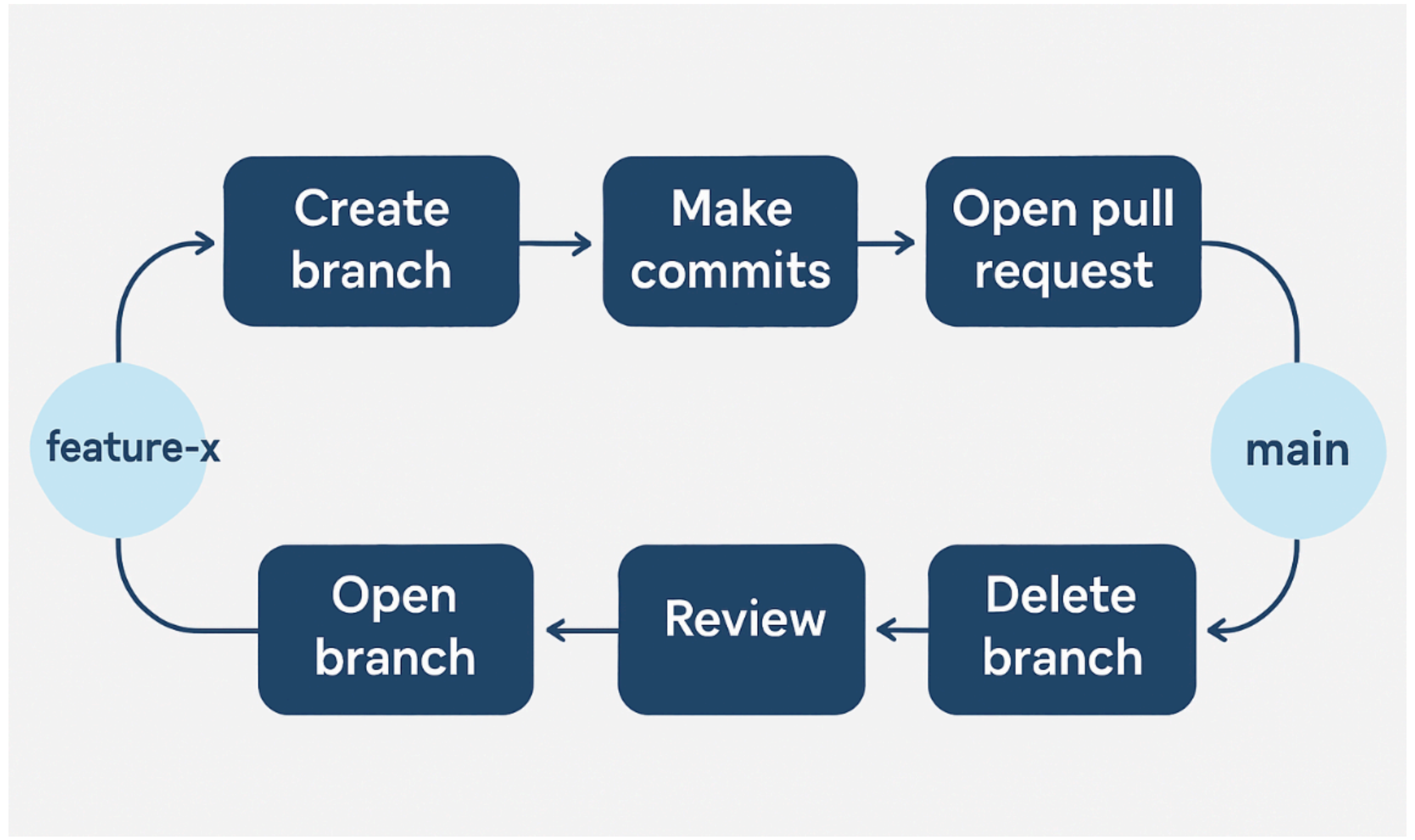
Clone the repo

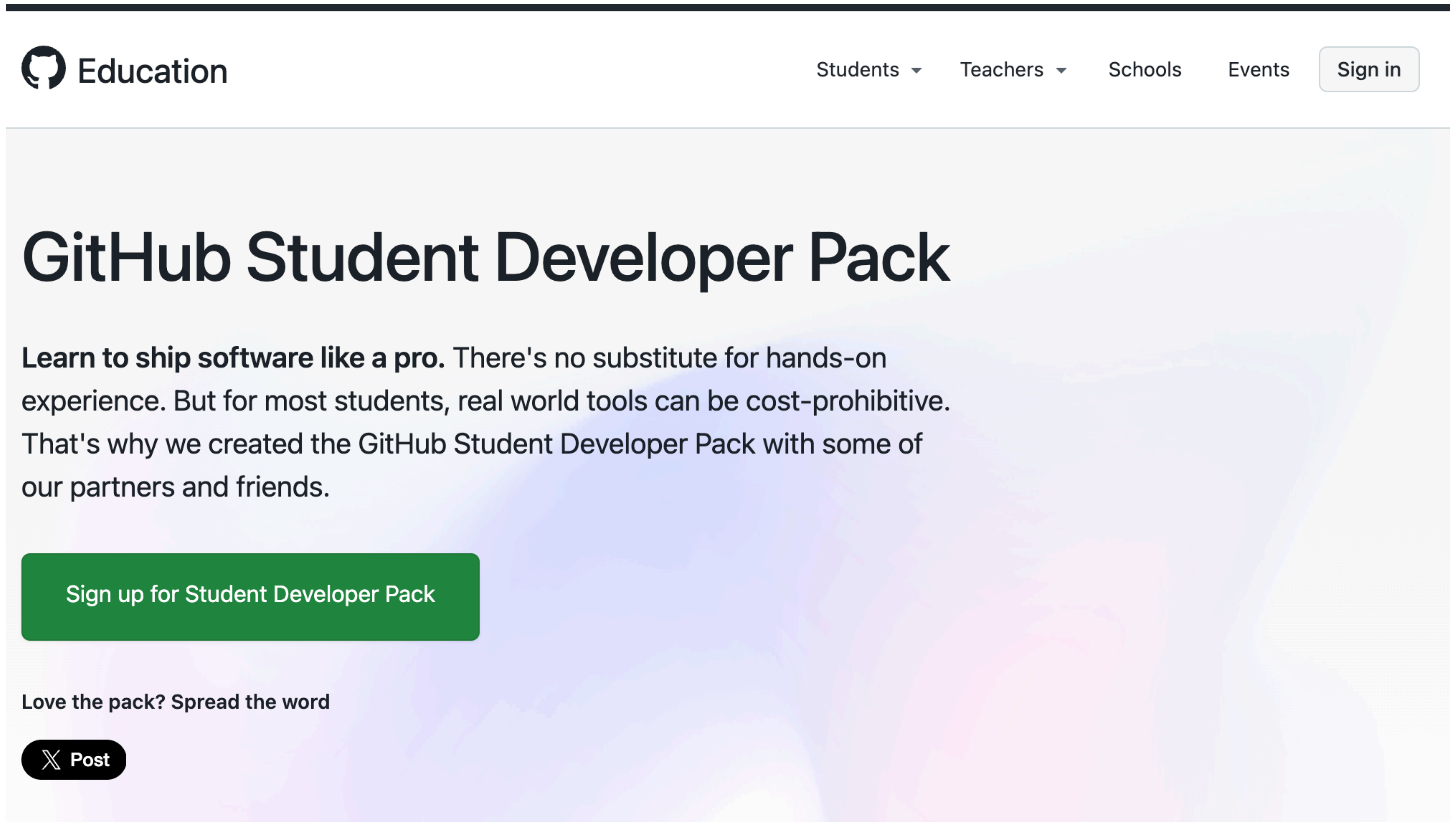
> Make your branch, make your changes

> push your branch

> pull request for code review

> merge into main/master




Education
Students
Teachers
Schools
Events
Sign in

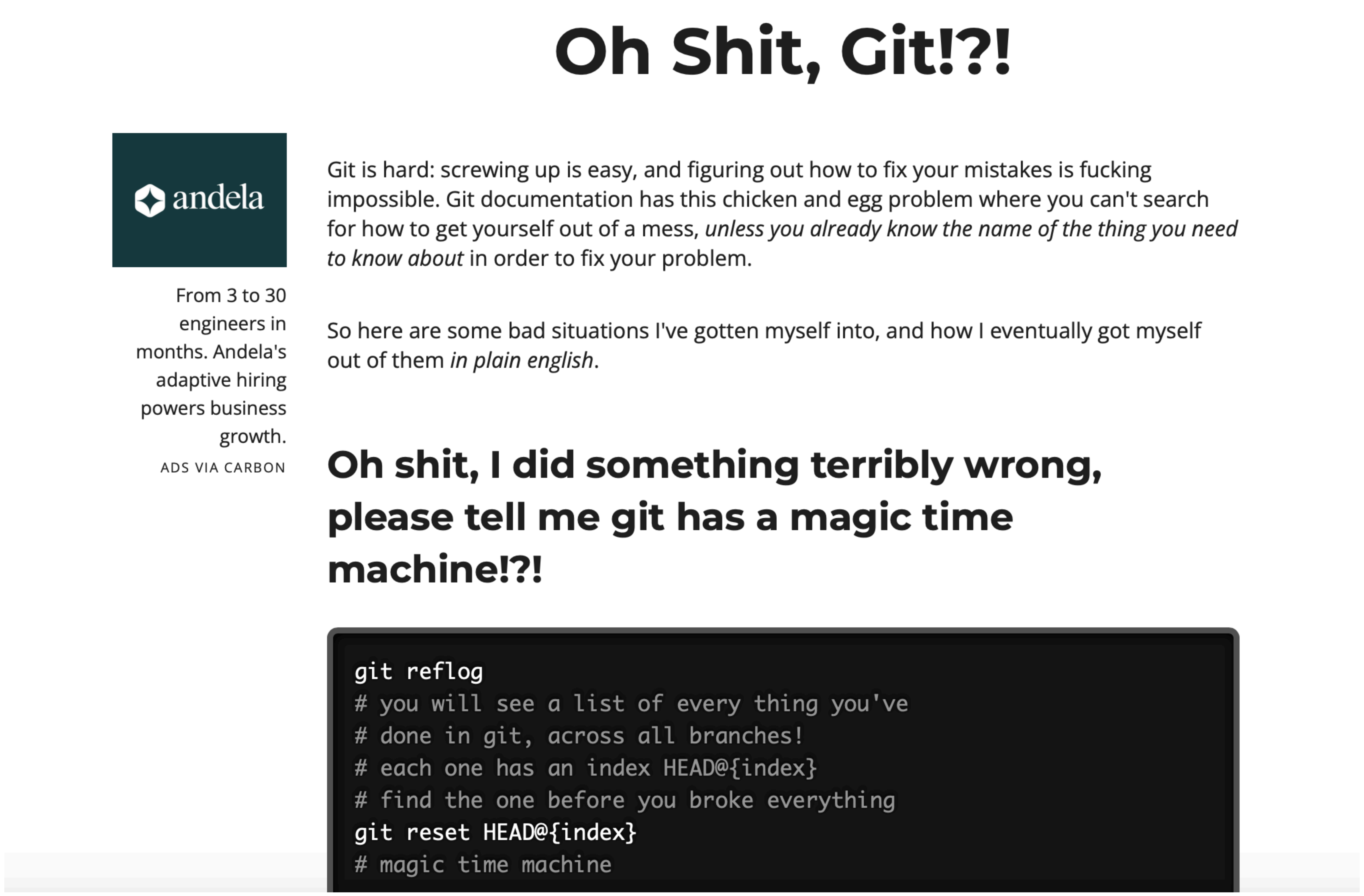
GitHub Student Developer Pack

Learn to ship software like a pro. There's no substitute for hands-on experience. But for most students, real world tools can be cost-prohibitive. That's why we created the GitHub Student Developer Pack with some of our partners and friends.

Sign up for Student Developer Pack
Love the pack? Spread the word
Post

https://education.github.com/pack

More help?



Oh Shit, Git!?!

andela

From 3 to 30 engineers in months. Andela's adaptive hiring powers business growth.

ADS VIA CARBON

Git is hard: screwing up is easy, and figuring out how to fix your mistakes is fucking impossible. Git documentation has this chicken and egg problem where you can't search for how to get yourself out of a mess, *unless you already know the name of the thing you need to know about* in order to fix your problem.

So here are some bad situations I've gotten myself into, and how I eventually got myself out of them *in plain english*.

Oh shit, I did something terribly wrong, please tell me git has a magic time machine!?!

```
git reflog
# you will see a list of every thing you've
# done in git, across all branches!
# each one has an index HEAD@{index}
# find the one before you broke everything
git reset HEAD@{index}
# magic time machine
```

Debugger

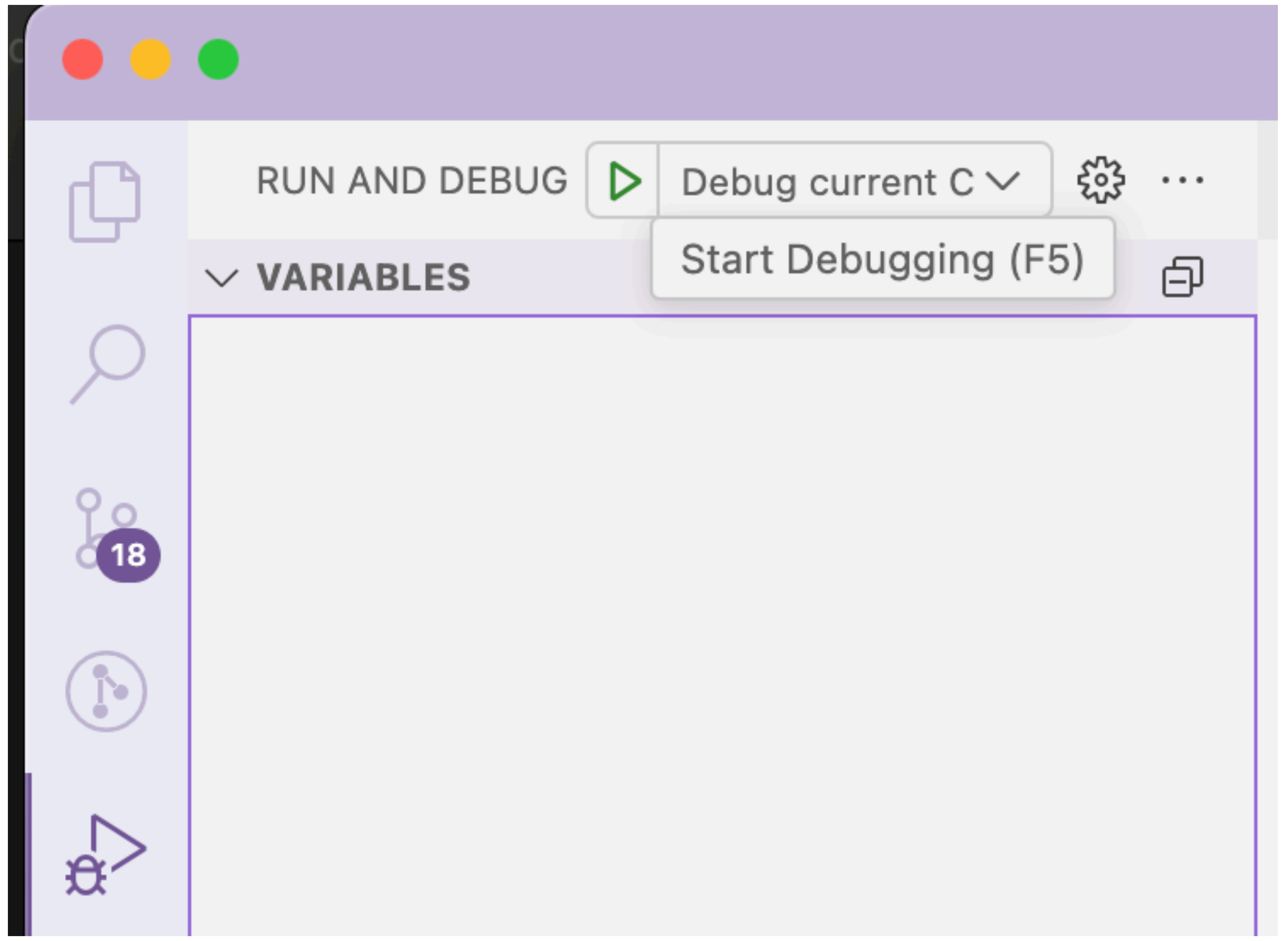
- **Pause** your program at any point
- **Inspect** what’s happening internally (variables, memory, etc.)
- **Step through** your code line by line
- **Fix bugs** without just guessing or 20 printfs

.vscode/launch.json

```
{
    "version": "0.2.0",
    "configurations": [
      {
        "name": "Debug current C file",
        "type": "lldb", // CodeLLDB extension
        "request": "launch",
        "program": "${fileDirname}/${fileBasenameNoExtension}",
        "args": [],
        "cwd": "${fileDirname}",
      }
    ]
  }
```

clang -g program.c -o program

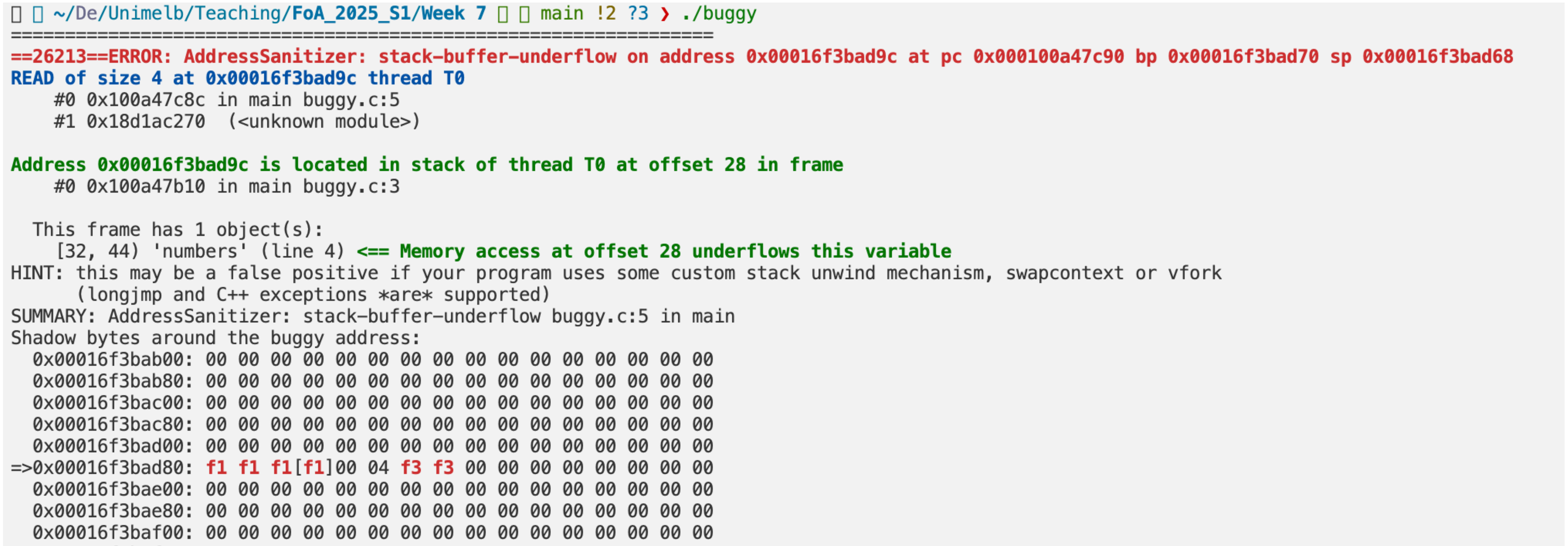
Visually in VSCode, or in terminal with **lldb**



Sanitizers

- Runtime introspection tools to detect memory errors, e.g. buffer overflows, null pointer dereferences, …

- clang -g -fsanitize=address buggy.c -o buggy



```
▯ ▯ ~/De/Unimelb/Teaching/FoA_2025_S1/Week 7 ▯ ▯ main !2 ?3 ❯ ./buggy
=================================================================
==26213==ERROR: AddressSanitizer: stack-buffer-underflow on address 0x00016f3bad9c at pc 0x000100a47c90 bp 0x00016f3bad70 sp 0x00016f3bad68
READ of size 4 at 0x00016f3bad9c thread T0
    #0 0x100a47c8c in main buggy.c:5
    #1 0x18d1ac270  (<unknown module>)

Address 0x00016f3bad9c is located in stack of thread T0 at offset 28 in frame
    #0 0x100a47b10 in main buggy.c:3

  This frame has 1 object(s):
    [32, 44) 'numbers' (line 4) <== Memory access at offset 28 underflows this variable
HINT: this may be a false positive if your program uses some custom stack unwind mechanism, swapcontext or vfork
      (longjmp and C++ exceptions *are* supported)
SUMMARY: AddressSanitizer: stack-buffer-underflow buggy.c:5 in main
Shadow bytes around the buggy address:
  0x00016f3bab00: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x00016f3bab80: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x00016f3bac00: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x00016f3bac80: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x00016f3bad00: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
=>0x00016f3bad80: f1 f1 f1[f1]00 04 f3 f3 00 00 00 00 00 00 00 00
  0x00016f3bae00: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x00016f3bae80: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x00016f3baf00: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```