Debugger, Command Line, and Git

Welcome back to CS 2100!

Prof. Rasika Bhalerao

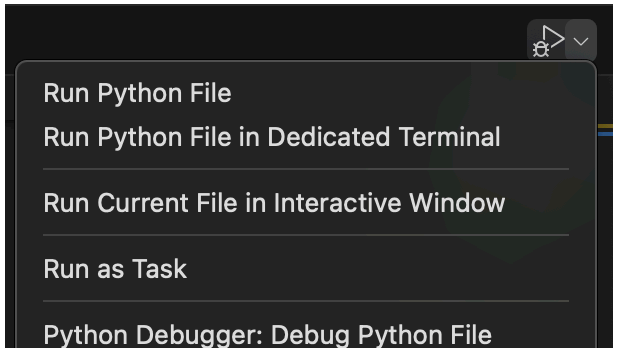
Poll: My code keeps printing the wrong thing. What should I do?

1. Run it again -- maybe it will magically work this time!
2. Stare at the code for a couple more hours trying random things
3. Use the debugger
4. Nothing -- this is unfixable

To use the debugger:

1. set a breaking point on at least one line
    - To set a breaking point, click next to the line number. A red dot will appear. (Click the same spot again to remove the breaking point.)
    - When we start the debugger, it will execute all of the code up to (but not including) that line.

2. To start the debugger, rather than clicking the "run" button, use the menu to select "Python Debugger: Debug Python File"

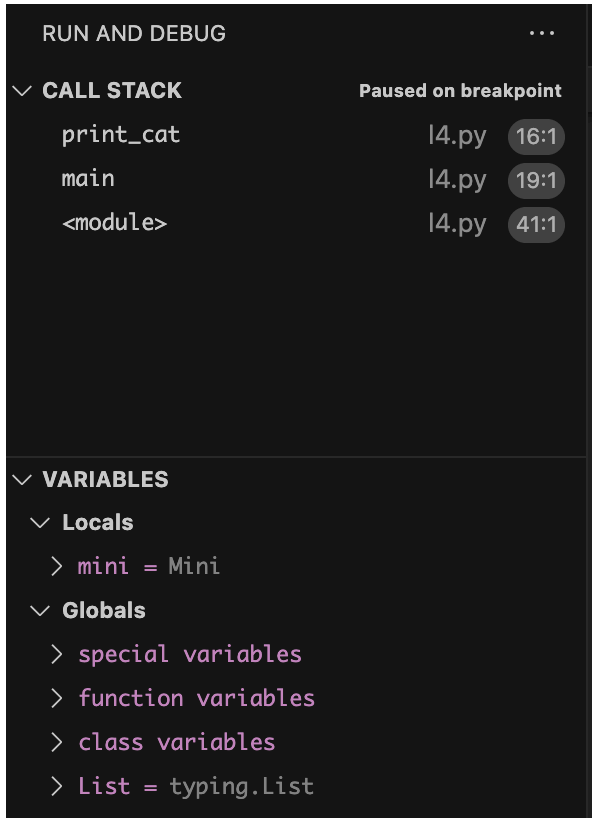


To use the debugger:

3. Two things will appear: the "Run and Debug" window, and the menu of controls

The "Run and Debug" window contains:

- "call stack": functions that are currently running
- current values stored inside all local and global variables



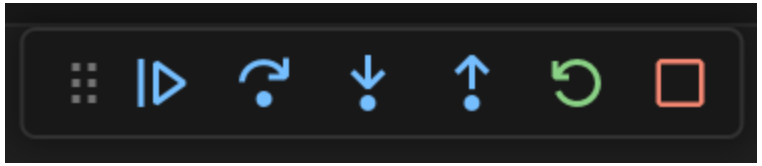
Called `main()`, and from there, called `print_cat()` (which is currently running, waiting for us to use the menu of controls)

Tip:

The debugger uses the method `__repr__(self) -> str` to get the string used to represent objects in the "Run and Debug" menu.

If you want the `__str__(self) -> str` method to be used to represent your object instead, you will need to add this to your class:

```
def __repr__(self) -> str:
    return self.__str__()
```

The menu of controls: 

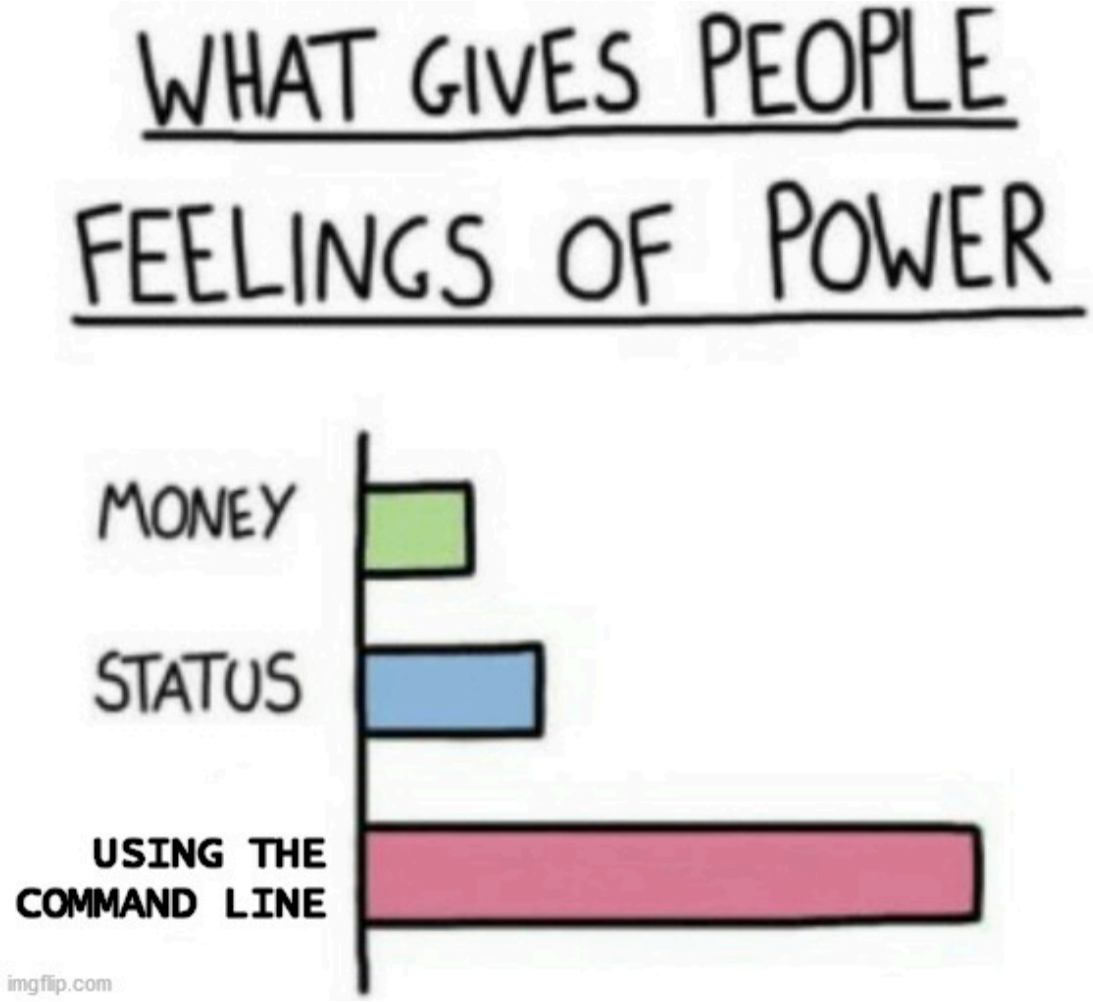
1. Six dots: drag the menu elsewhere on your screen
2. Continue running the program until the next breaking point (or the end if none left)
3. Run the current (highlighted) line of code.
    - If that line contains a function call, execute the entire function.
4. Run the current (highlighted) line of code.
    - If that line contains a function call, enter that function and run only its first line.
5. Continue running the code in the current function, until we return control to the function that called it. Then stop at the next line in the outer function.
6. Restart the debugger from the beginning of the program.
7. Stop the debugger and close it without running the remaining code.

Tip:

Use tests to debug buggy functions.

1. If there is a function with a bug, write a test that calls that function in a way that is likely to make the bug happen
2. The test will fail at first, which is good because if the test passes, then you know you have fixed the bug
3. Place a breaking point in the buggy function, and start the debugger on the test

The command line



The command line is a powerful way to navigate a computer.

In this course, we will use the command line to navigate assignment files.

- Mac or Linux: Terminal app
- Windows: Git Bash

Most popular commands:

Command	Description	Example
`ls`	short for “list”; prints the contents of the current working directory	`ls`
`cd`	short for “change directory”; moves you to the given directory	`cd ~/Desktop`
`pwd`	short for "print working directory"	`pwd`
`mkdir`	short for “make directory”; creates a new subdirectory at the current location with the given name	`mkdir nuresources`
`touch`	creates an empty file in the current directory with the given filename	`touch lecture8.py`

Changing the directory using `cd`

- Using the absolute directory, the complete path from the root to a given directory or file: `cd ~/Desktop/Lectures/2100`
- Using the relative directory, the path from our current working directory to a given directory or file: `cd Lectures/2100` (from the Desktop)

Poll: The command `cd fa25-hw3-lists-rasikabh/src/data` results in an error. What is NOT likely to be the cause of this error?

1. The current location is not the directory where I store my homework assignments
2. There is no directory called `src` in the directory `fa25-hw3-lists-rasikabh`
3. `data` is a single file, not a directory
4. The assignment submission for Homework 3 is closed

git

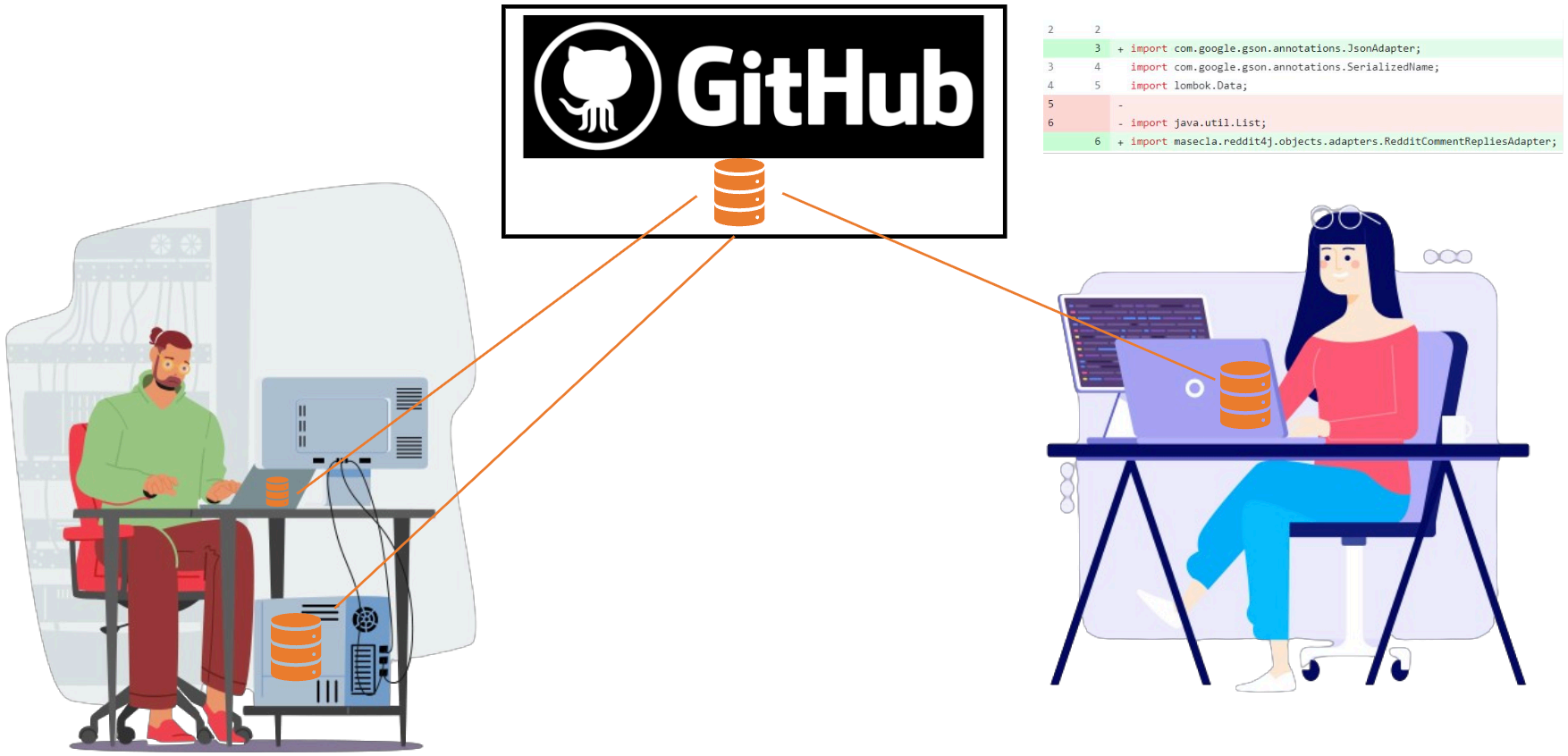
Have you ever wished you could...

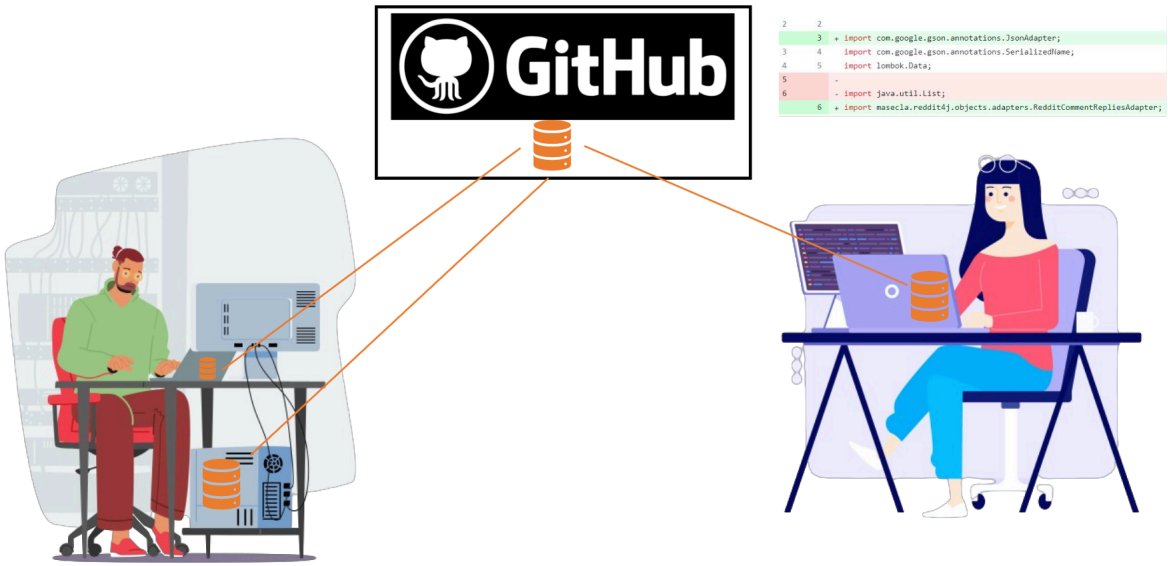
- go back to an earlier working version of a project?
- see what changed between versions of your code?
- tell who made a certain change, when and why?
- switch between using your laptop and desktop to work on a project?
- work with a friend without worrying about overwriting each other's changes?
- have someone review your changes before adding them to a project?
- make some experimental changes with the option of undoing them?
- create your own version of an existing project, change it, and incorporate updates to the original project?

(List source: Ellen Spertus)

Git is the leading distributed version control software, created by Linus Torvalds in 2005

GitHub is a website owned by Microsoft that hosts git repositories and has a web interface (plus other tools like automated testing and issue tracking)





When you "start" an assignment using Pawtograder, it creates a code repository on GitHub containing the "starter code" for that assignment.

You then use `git clone` to copy that code repository to your laptop, and open / edit the code using VSCode on your laptop.

Key git concepts

- Repository (repo): a set of code and its history
  - local: on your computer
  - remote: on another computer (like GitHub)
- Commit
  - the codebase at a given point in time (noun)
  - to add a set of changes to the repository (verb)
  - Push: to move code from a local to remote repository

Tip: git repositories have a directory in them called `.git` which is invisible by default. To get it to show up when you use the `ls` command, you must do this: `ls -al`

Locations of versions of code

Location	Definition	git command to put code there	Postal analogy
working area	code that you are currently writing / saving in VSCode		Writing on a paper
staging area	code that is ready to be commited	`git add .`	Add a stamp and put it in your backpack
local repository	code that has been committed	`git commit -m "description"`	Put all stamped cards in the mailbox
remote repository	code on GitHub	`git push`	Workers move cards to destinations

Most assignments will go like this:

- Accept the assignment on Pawtograder
- Pawtograder creates your GitHub repo.
- Using your command line, navigate to this course's directory and do `git clone <GitHub repo URL (ssh version, not html)>`
- You open the resulting files using VSCode and work on the assignment, saving as you go.
- After each significant chunk of progress on the assignment:
    - `git add .` to stage changes in all files in this directory
    - `git commit -m "descriptive message"` to commit the changes
    - `git push` to push the changes to the online repo in GitHub
        - Pawtograder will automatically take that as your submission (if the submission is still open)

Other commands

- `git pull` takes changes any that others pushed to the repo on GitHub, and copies them to your local repo
- `git status` reports which files have been changed and staged
- `git diff` shows every changed line
- `git diff --staged` shows the difference between staged and committed changes
- `history` shows the history of commands you typed into the command line

Poll: Why is this not an ideal commit message? `"Complete Homework 3"`

1. It doesn't describe the code changes
2. It implies that all changes to the entire assignment were submitted in a single commit
3. We can't change Homework 3 again, since we said we completed it
4. All of the above
5. (1) and (2) only

Poll:

1. What is your main takeaway from today?

2. What would you like to revisit next time?