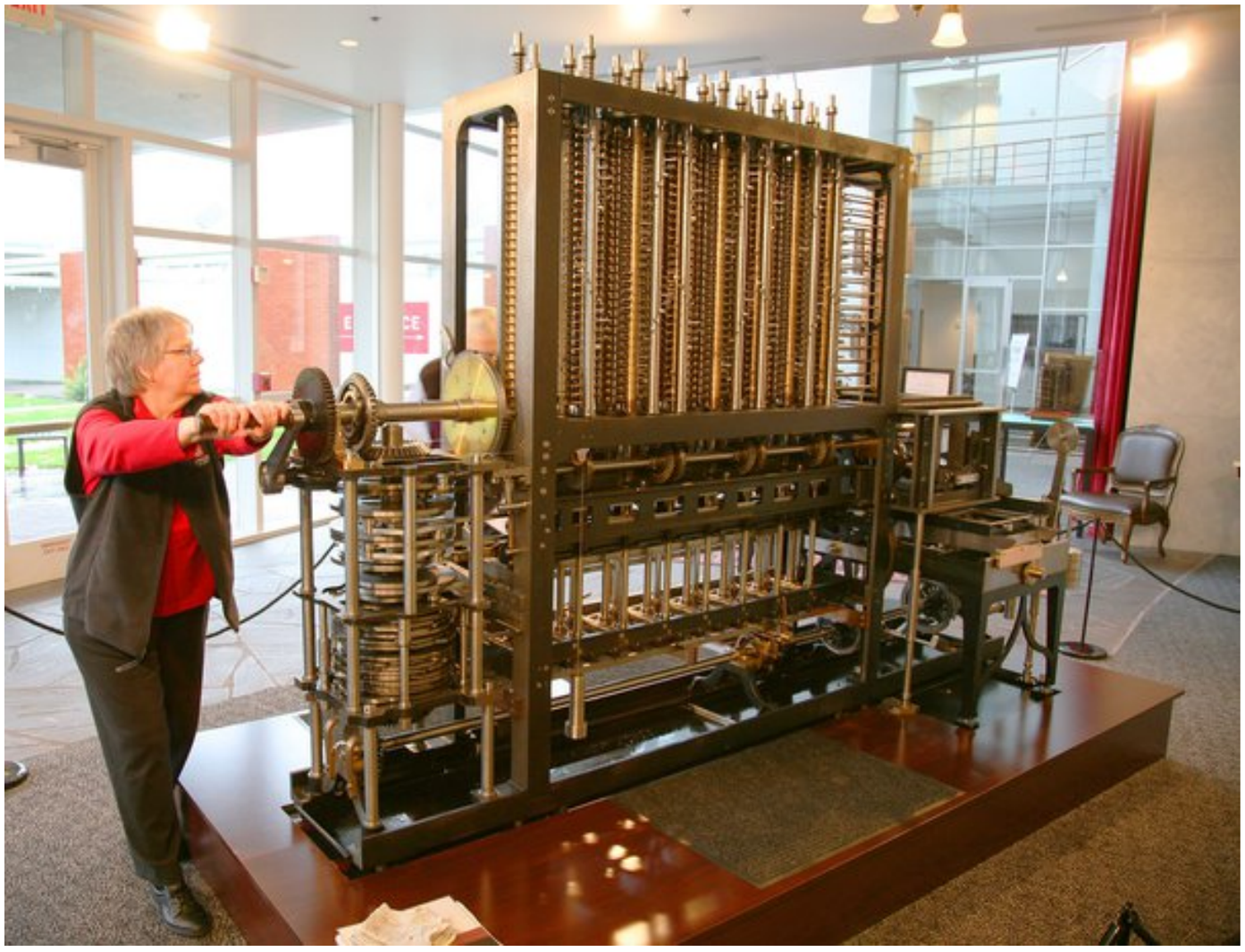
MCB 536: Tools for Computational Biology
Lecture 04: Intro to Command Line

Melody Campbell, Fred Hutch

Teaching Goals

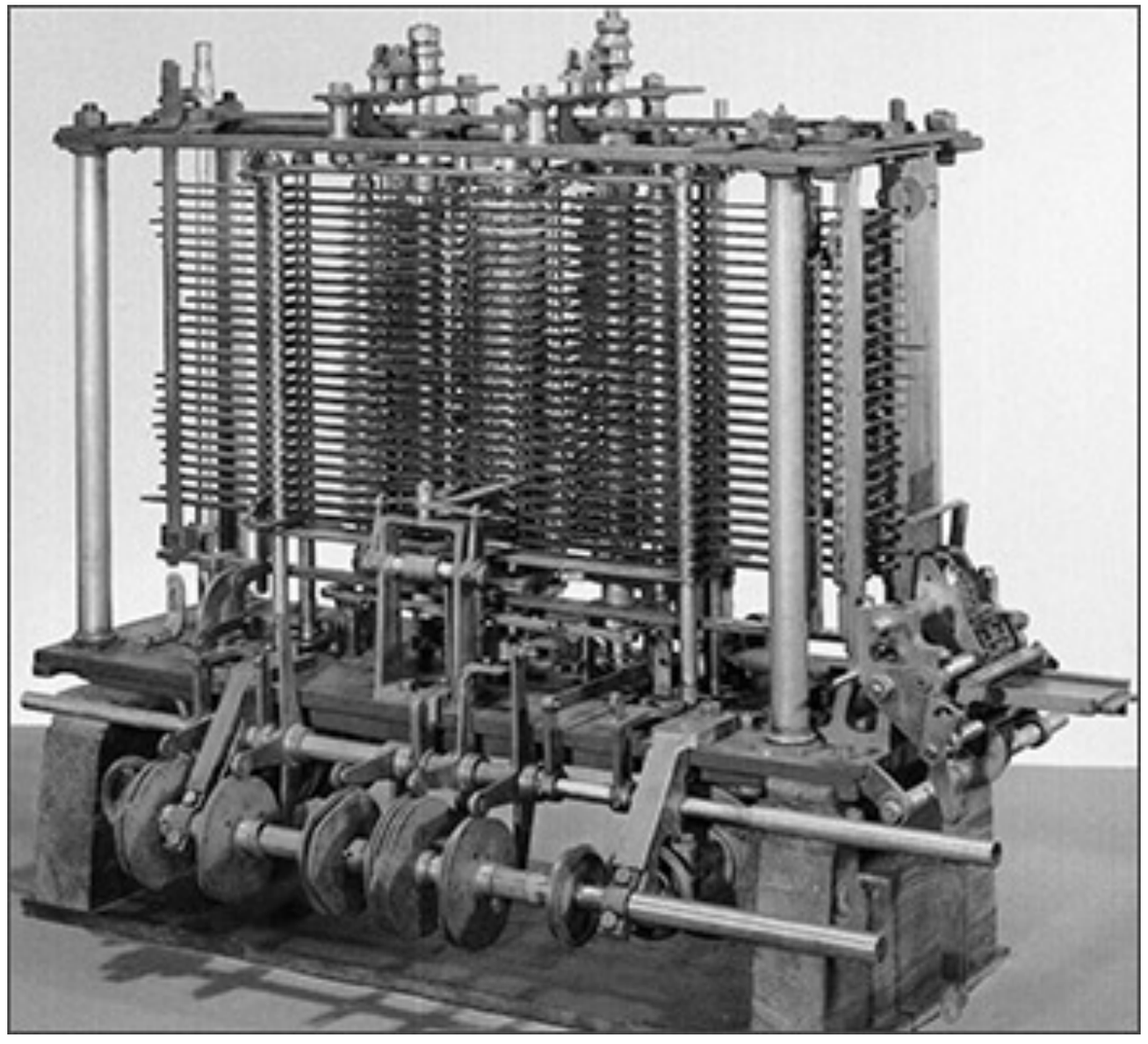
- History: how did we get to the command line?
- Interacting with the command line
  - Commands
  - Flags
  - Wildcards
  - Variables
  - Outputs
  - Autocomplete
  - Terminate

The Interface btwn Human & Computer



Early Interfaces

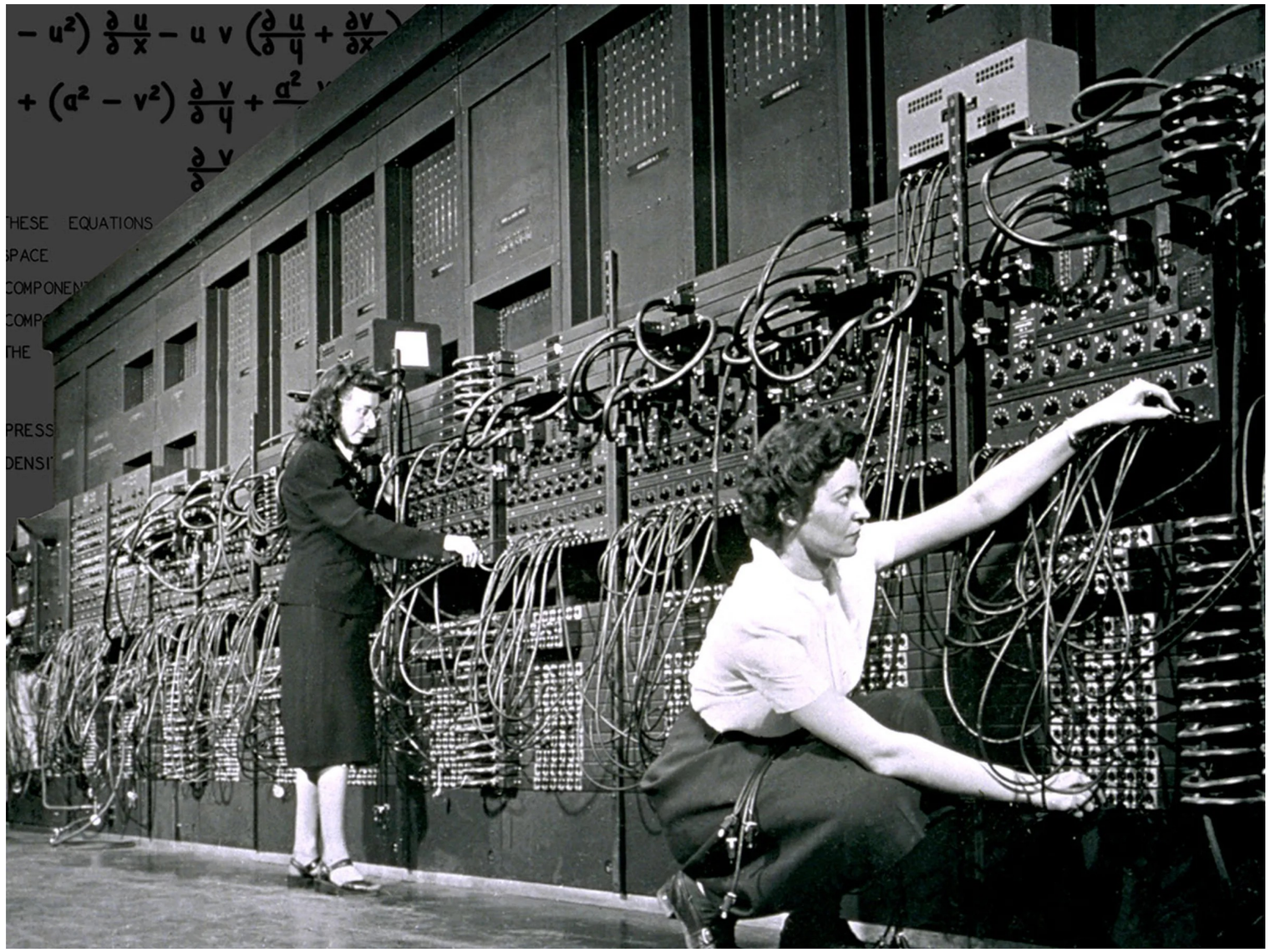
- Use physical controls for inputs and outputs: gears, knobs, switches



1837: Analytical Engine

First Electronic Computers

- Still relied on mechanical controls and patch wires
- Took weeks to input a program



1945: ENIAC

First Electronic Computers

- Took weeks to run a program
- Output is paper

Early 1950's: Updated Inputs / Outputs

- Inputs: Punch cards and magnetic tape replace mechanical
- Outputs: Still paper

Log files were lights!

Optimized for Computers (not Humans)

- Rolls of punch tape easy to read
  - continuous
  - holes can be read optically or mechanically

also, not interactive!

Programmers have to convert their language and programs to a format that is easy for computers to understand

Late1950's: Back & Forth

- Computers get smaller & more sophisticated
- Multitasking and Timesharing Systems
- Start using keyboards!

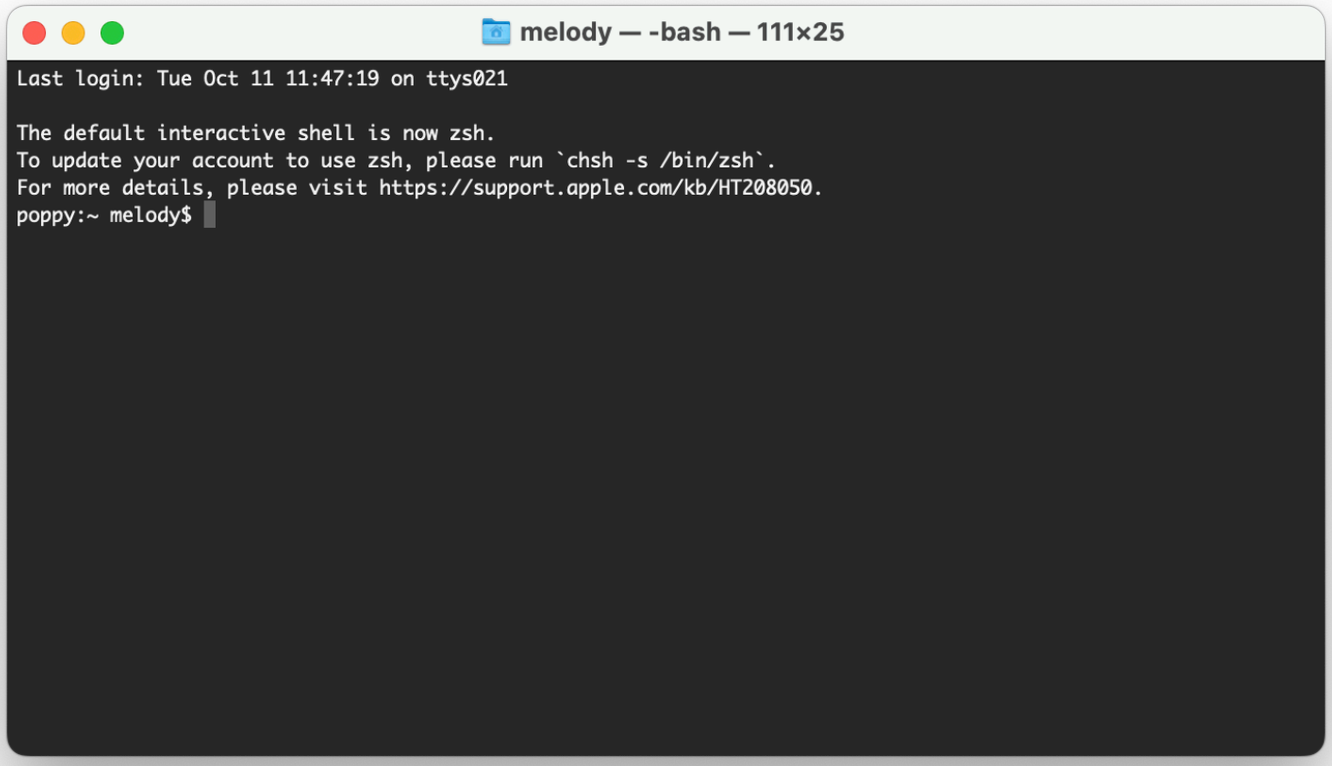
60’s & 70’s: Electronic Teletype

- A electro mechanically augmented typewriter used for telegraphs: one person types, electronic signal transferred via telegraph line, transmitted to another teletype machine

- This was easy to adapt to computers: user types, computer ‘types’ back

Command Line Interface

- ^ The name for the adapted teletype human/computer interface
- This was the most prevalent form of human/computer interaction until the 1980's
  - Screens became cheaper in the 1970's

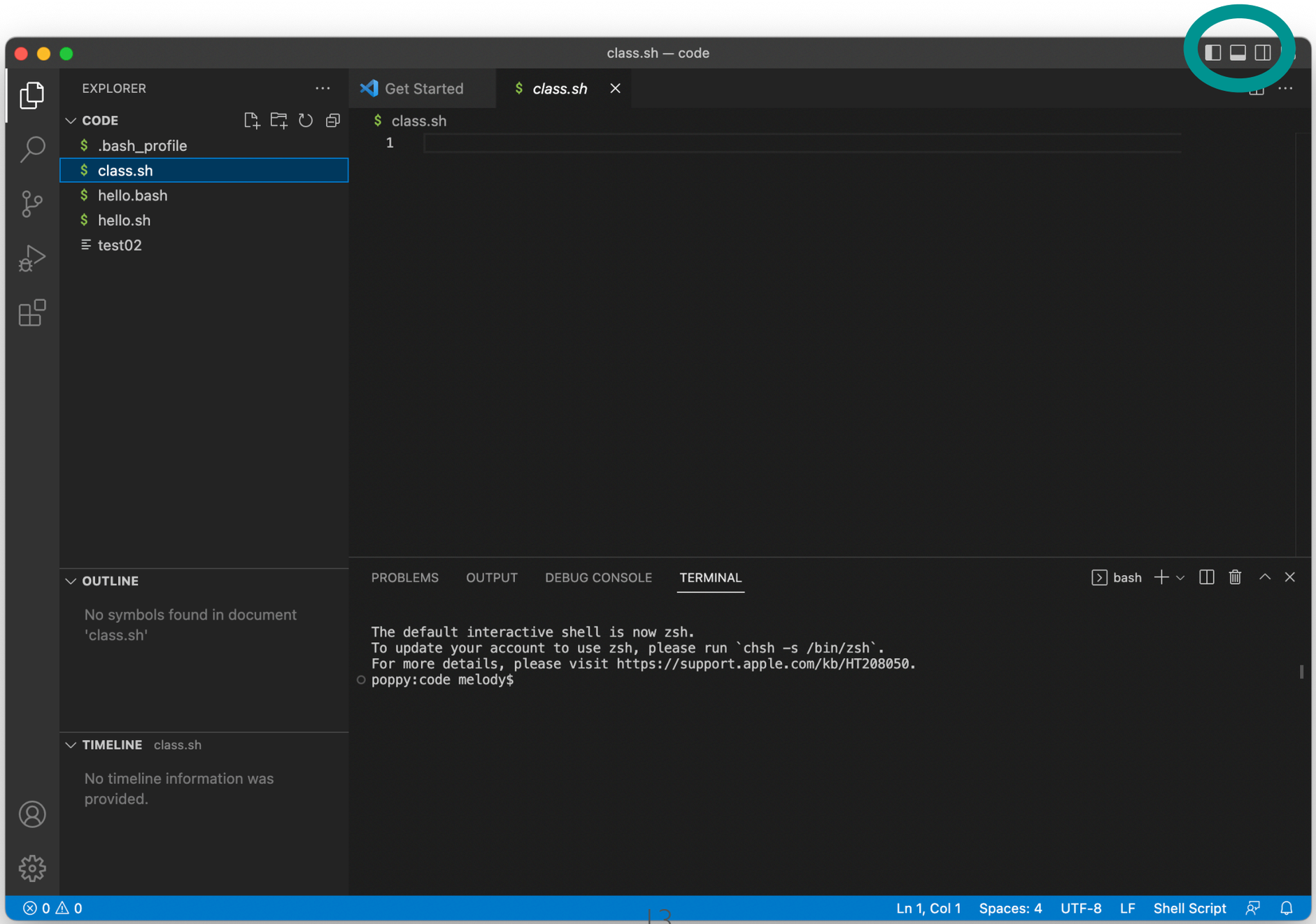


Add a Screen -> Terminal

- We have screens (infinite paper), we still use the ‘electronic teletype procedures’
  - This way engineers didn’t have to design a whole new way for humans to interact with computers
    - text in > text out
- Known as “Glass teletype machines”
- Now called terminals

What can you do with a terminal?

- Open Visual Studio Code
- Go to Terminal



pwd

- pwd: print working directory
  - aka where am i?
  - what is a directory?
    - it's just a folder
  - what is a working directory?
    - it's the folder you're in
- you are probably here:
  - /users/$USER/(the name of the folder (directory) you made when you set up VS)
- what does $USER mean?
  - try echo $USER
    - what does echo mean?
      - … what is a variable??
        - what is a command??????

Optimized for Computers (not Humans)

- Rolls of punch tape easy to read
  - continuous
  - holes can be read optically or mechanically

also, not interactive!

Programmers have to convert their language and programs to a format that is easy for computers to understand

You are learning a new language.

Translation: English to Linux

What folder am i in? → pwd

Which user am i logged in as? → echo $USER

pwd

- pwd: print working directory
  - aka where am i?
  - what is a directory?
    - it's just a folder
  - what is a working directory?
    - it's the folder you're in
- you are probably here:
  - /users/$USER/(the name of the folder (directory) you made when you set up VS)
- what does $USER mean?
  - try echo $USER
    - what does echo mean?
      - … what is a variable??
        - what is a command??????

Let’s jump in first & explain later

- This is a reasonable way to approach coding: just try stuff!
- It’s very difficult to break things
  - mash the keyboard: what happens?

Let’s jump in first & explain later

- This is a reasonable way to approach coding
- It’s very difficult to break things
  - mash the keyboard: what happens?

```
poppy:tfcb_2022 melody$
poppy:tfcb_2022 melody$ KSEJHDFISZH IS HFYW A478RYJBS jiwhey83rwy
bash: KSEJHDFISZH: command not found
```

nothing bad!

Let’s jump in first & explain later

- Figure out where you are: pwd
- If you are not in /Users/$USER get there
- cd
  - change directory
  - this is what lets you move in and out of folders (like double clicking on a normal computer)
    - cd $HOME
    - pwd
- mkdir
  - make directory
  - this makes a new folder (like command + shift + N on a normal computer)
  - make a folder called tfcb_2022
    - mkdir tfcb_2022

Let’s jump in first & explain later

- cd tfcb_2022
- Let’s make three simple text files 1,2,3 with a single word:
  - echo hello01 >file01.txt
  - echo hello02 >file02.txt
  - echo hello03 >file03.txt
    - It is annoying to repeat this command 3 times, but we will learn a better way soon
- The arrow > takes the text “hello01” and outputs it as a file called file01.txt
  - Be careful! > will overwrite any current file!
  - If you want to *add* you need two: > >
- echo hello04 >file04.txt
- echo goodbye04 >>file04.txt

Let’s jump in first & explain later

- How do we check what is written in each file?
- cat
  - concatenate
    - try: cat file01.txt
    - cat file02.txt
    - cat file04.txt
      - tab complete to make your life easier!

Commands

- When you type something into a terminal, the computer assumes it is a command
  - try typing your name
- Commands you've used so far: pwd, cd, mkdir, echo, cat

Argument

- "A value that is passed between programs, subroutines, or functions"
    - cat file01.txt
  - If cat is a verb, than file01.txt is a noun
  - You can have multiple arguments (this depends on the command)
    - cat file01.txt file02.txt

* Wild Cards!

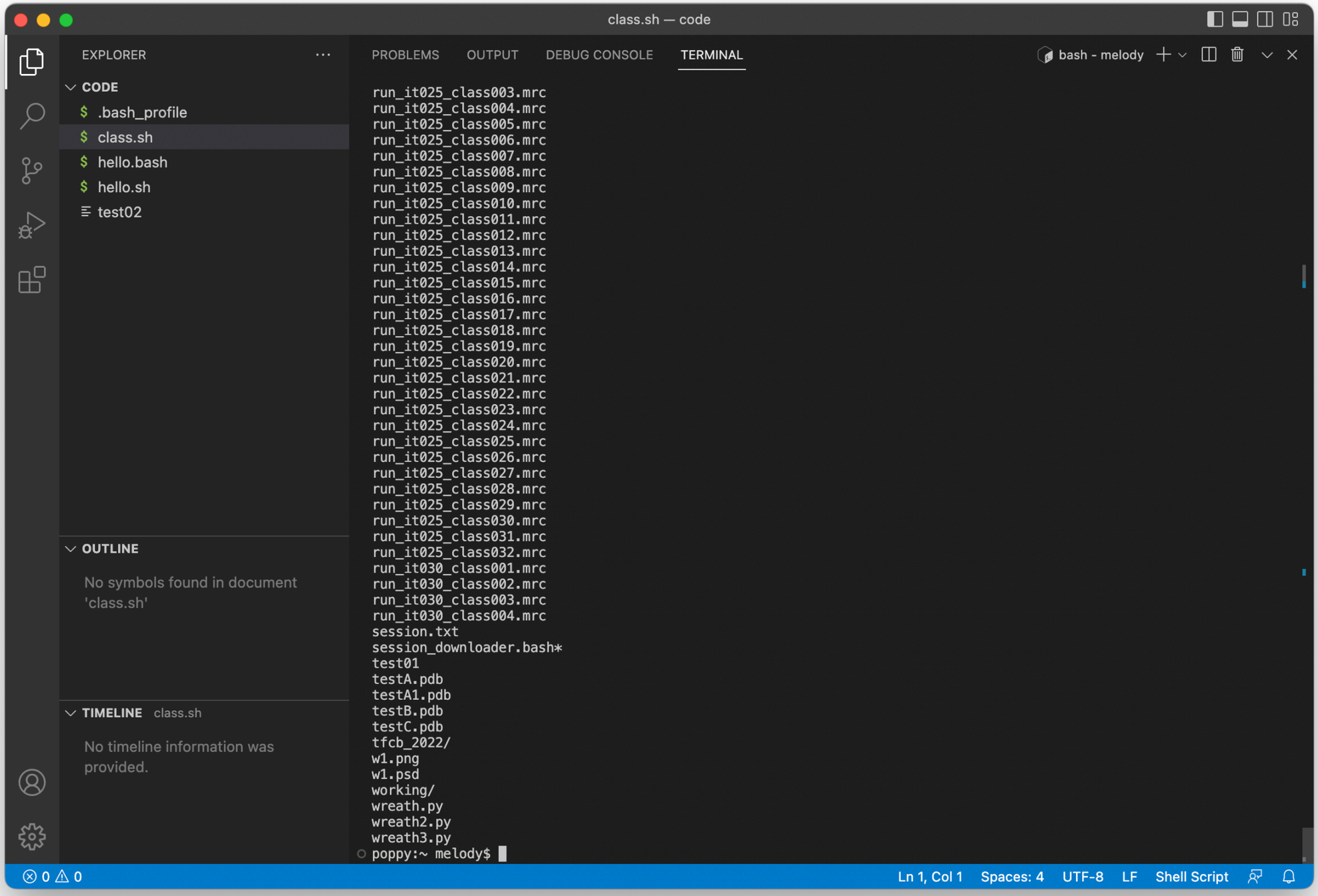
- You can have multiple arguments (this depends on the command)
    - cat file01.txt file02.txt
    - cat file01.txt file02.txt file03.txt file04.txt
- Sometimes you want to read more than one file at once, but you don't want to type it all in. try:
    - cat *.txt
- Also very useful for listing files only of a certain type:
    - ls *.txt
    - ls *.doc
    - ls *.ppt

ls

- ls
  - list directory contents
- cd ..
  - .. means move “up” one directory
  - cd ../.. means move up two directories
    - / delineate folders (e.g. don’t put a / in a file name)
  - . is the directory you’re in
  - shortcut: try cd alone
    - this brings you back to your home dir
- ls
  - what is in this new directory? (/Users/$USER)

Flags

- Flags modify the command
- default ls:

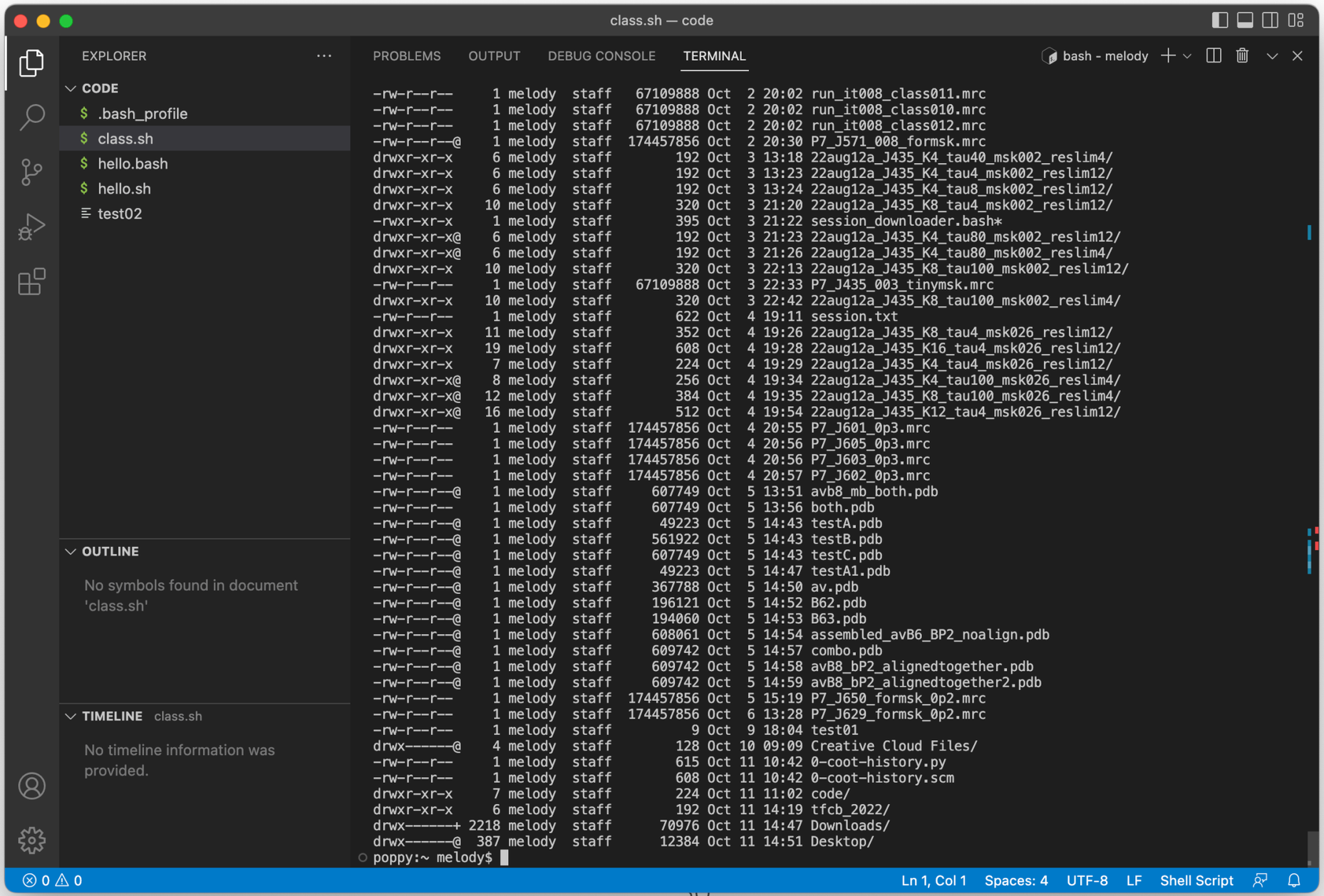


Flags

- Flags modify the command
- type ls
  - default is short format, alphabetical order
- type ls -l
  - -l is a flag
    - it stands for long format
- my favorite: ls -ltr
  - -t sort by depending time (most recently modified files first)
  - -r reverse time (most recently modified files last)

Flags

- my favorite: ls -ltr
  - note how the output has changed



How do I find flags?

- man
  - This commands stands for manual and it will tell you different options of flags that can be used on a command
  - man ls
    - note: in this example, ls is now the argument, not the command
    - type ‘q’ to end this command
- flags are different for every command
  - what works on ls will not work on all commands

Syntax (Structure)

command -flag(s) argument

ls -ltr tfcb_2022

what do you want me to do?

what options do you want?

what should i perform it on?

verb adverb noun

english: list out time sorted backwards and fully what is in this folder

Variables

- Variables are shown by having a dollar sign
- Some are set by most systems ($USER $HOME)
- Others you can set on your own to personalize your computer ~OR~ for writing simple scripts
  - They can update and change!
    - they can be commands or flags or arguments
- Example: today_is=october ; echo $today_is
- questions…
  - what is that vertical line?
  - why didn’t you use the space?

; and (escape character)

- This vertical line basically takes any info from the prior command and passes it on to the subsequent command.
- You can also do this on two lines
  - today_is=october
  - echo $today_is
- Try not to use spaces in file names!
  - Spaces delineate commands, flags and argument
    - If the computer sees a space it thinks it is should move to the next command or argument
  - If you have to use a space or a special character, put a backslash \ first
    - echo hello06 > space\ file.txt
    - this tells the computer that whatever comes next should only be interpreted as text, not the next command

Terminate & Cursor shortcuts

- Say you accidentally typed a bunch of things in the terminal that you don't want anymore…
  - Control + C
  - this will not run the command, and give you a fresh new line
- The cursor can be annoying. You have to move via arrows
- Say you typed a bunch of things in the terminal and you made one mistake
  - at the beginning
    - Control + A
  - at the end
    - Control + E

Why are we doing this?

- Many programs and computing clusters don't have nice looking files that you can click and drag to get around
- You now have many of the tools you will need to write scripts and routine and thus automate your work
  - for example: it is much nicer to write a few lines of code that to manually change .jpgs to .jpgs 10,000 times
- Shell scripting helps you get files into the correct format to feed into more sophisticated programs

October 11, 2022 Additions

- How to rename:
  - mv input.txt output.txt
- Copy files
  - cp file.txt file2.txt
- How to delete a file
  - rm file.txt
- How to delete a dir
  - rm -r directory
- Print out contents of dir into new text file
  - ls * > contents.txt