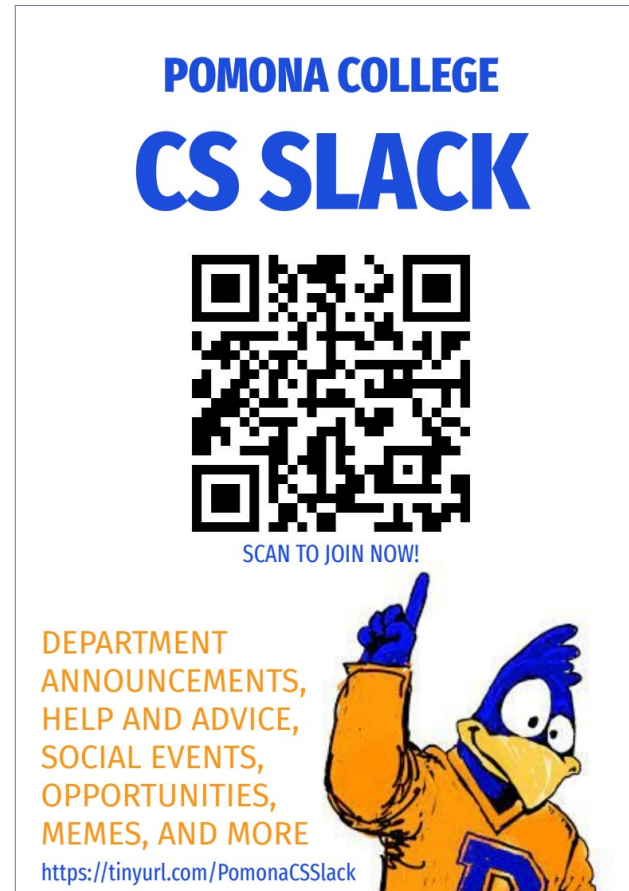


some things to know about



<https://listserv.pomona.edu/scripts/wa.exe?SUBED1=CSCOLL0Q&A=1>

<https://listserv.pomona.edu/scripts/wa.exe?SUBED1=CSALL&A=1>



csci54 – discrete math & functional programming
introduction (to everything)

-
- ▶ procedural/object-oriented (and not functional)
 - ▶ write a function `maxInt` in Java/Python that takes an array/list of integers and returns the value of the largest element. you may assume the array/list is not empty.
 - ▶ continuous (and not discrete)
 - ▶ given a (continuous) function $f(x)$, how do you find the maximum value on the closed interval $[a,b]$?



discrete math & functional programming

- ▶ discrete math is concerned with structures that can be counted
- ▶ functional programming requires solving problems by just applying functions to arguments, rather than by updating state.
- ▶ "thinking functionally" is very similar to "thinking mathematically"

```
maxInt [x] = x
maxInt (x:xs) = max x (maxInt xs)
```



Calendar, course website

<https://www.cs.pomona.edu/classes/cs54>



structure of the class

- ▶ basic weekly structure:
 - ▶ Tuesday: lecture
 - ▶ Thursday: lecture
 - ▶ Thursday/Friday: small group meetings
 - ▶ Friday 10pm: turn in small group assignment
 - ▶ Sunday 10pm: weekly problem set due
- ▶ two in-class midterms and an in-class final during finals week



small groups

- ▶ groups of ~6 that meet with an assigned TA for an hour once a week on either Thursday or Friday.
 - ▶ more on this, including first-pass on group formation, later this lecture.
- ▶ weekly low-stakes small group assignment available on gradescope/Canvas Wednesday
- ▶ due in gradescope Friday 10pm
- ▶ no extensions (make sure your group knows who's turning in!)
- ▶ first group assignment due this Friday



weekly assignments

- ▶ available on gradescope/Canvas by Thursday
- ▶ due in gradescope Sunday 10pm
- ▶ coding and/or written
- ▶ will initially be done in assigned groups of 2-3 with all members from the same small group. about halfway through the semester will add the option to choose own partner(s) as well as the option to work alone.
- ▶ if you need an extension let me know
- ▶ first problem set due this Sunday (done individually)



grading

- ▶ If you don't like your grade on a homework or test...
 - 📖 Or if you have lingering questions...
- ▶ Meet with me or a TA to talk through what you tried!
 - 📖 “Oral retakes” like this can regain points
- ▶ Even if you didn't finish the assignment...
 - 📖 It's worth talking through so you can stay on pace



Some rules and regulations

- ▶ don't look for solutions to problem set questions on the web (including ChatGPT or copilot), from students who have taken the class previously, or other sources not specifically distributed for this class this semester.
 - ▶ don't share (non-publicly-available) materials from the class with anyone not in the class this semester
 - ▶ but please discuss concepts with other people involved with the class this semester: me, the TAs, other students!
 - ▶ do not look at other people's (working) code
 - ▶ write code/proofs/etc on your own: do not, for example, take a photo of code/notes on the whiteboard. If you understand the idea, you should be able to write the code/proof/etc on your own without referring to notes.
-



resources

- ▶ Course webpage: syllabus, calendar, logistics
- ▶ Canvas: information, announcements, resources, problem sets, lecture notes
 - ▶ Discussion forum
 - ▶ for questions/comments potentially of interest to the entire class and that do not contain private/confidential information
 - ▶ email me directly about things specific to you (e.g. grades)
- ▶ Gradescope: for turning in assignments
- ▶ readings:
 - ▶ calendar at <https://www.cs.pomona.edu/classes/cs54> includes relevant chapter/sections



resources – readings

- ▶ "Learn you a Haskell for Great Good!" by Miran Lipovača

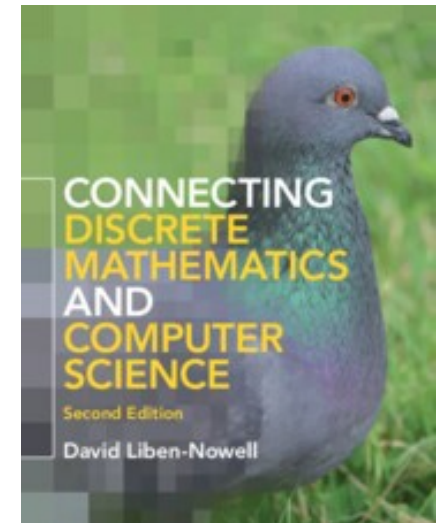
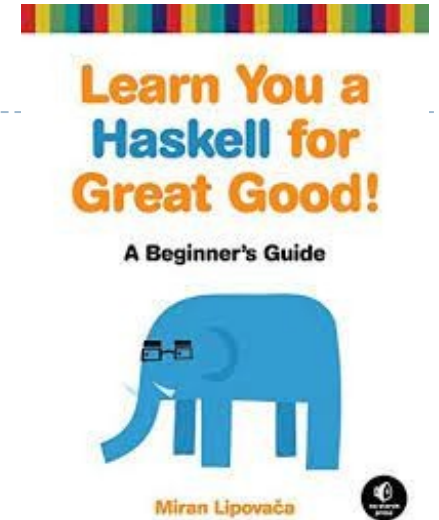
- ▶ <https://learnyouahaskell.com>

"Connecting Discrete Mathematics and Computer Science" by David Liben-Nowell

- ▶ <https://cs.carleton.edu/faculty/dln/book/>

Some other books, videos, etc listed on course webpage

- ▶ Let us know if you run across anything you find helpful for learning the material!

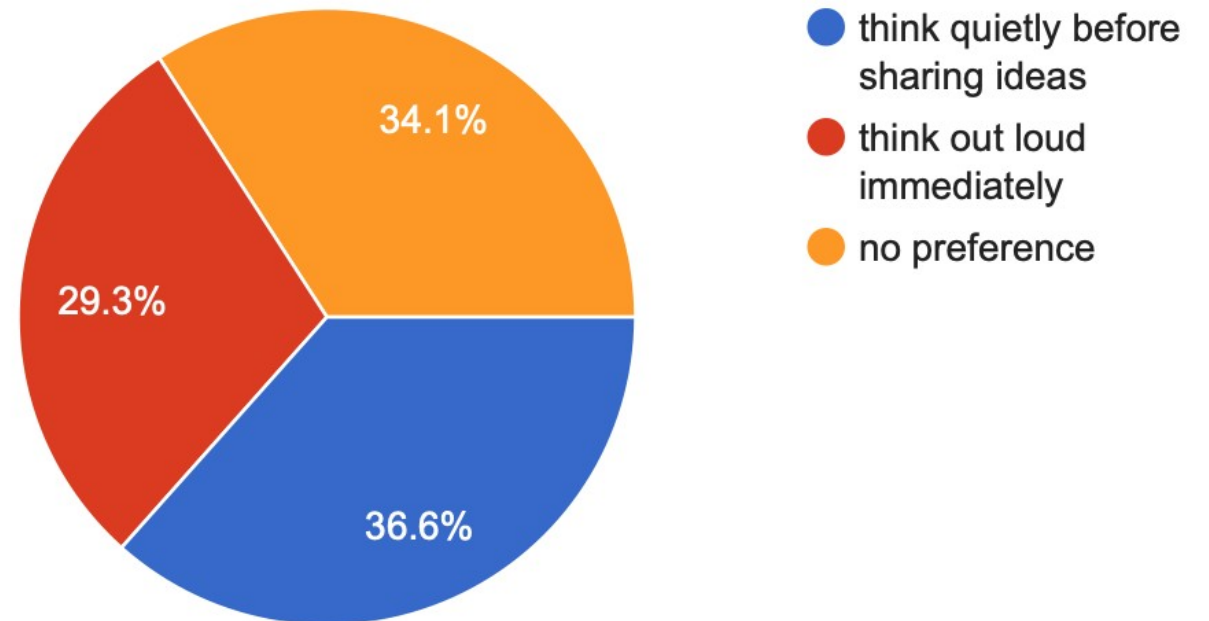


resources

- ▶ me (Prof. Osborn)
 - ▶ Edmunds 113
 - ▶ office hours: TBD
 - ▶ meal sign-up on office door by next week
- ▶ TAs:
 - ▶ Kendrick Dahlin
 - ▶ Tara Mukund
 - ▶ Dylan Yang
 - ▶ Francisco Morales Puente
 - ▶ Asya Lyubavina
 - ▶ Alyssa Coleman
 - ▶ Kellie Au
 - ▶ Emily Zhu
- ▶ QSC: pomona.mywconline.com, qsc@pomona.edu

small groups

- ▶ ~6 students in each group
 - ▶ assigned using results of survey that you should have filled out by noon yesterday
 - ▶ same group for entire semester, responsible for choosing a 1-hour weekly meeting time on Thursday or Friday (**try to avoid 4:15-5:30 Thursday**)
- ▶ first meeting this week



small groups

- ▶ ~6 students in each group
 - ▶ assigned using results of survey that you should have filled out by noon yesterday
 - ▶ same group for entire semester, responsible for choosing a 1-hour weekly meeting time on Thursday or Friday (**try to avoid 4:15-5:30 Thursday**)
 - ▶ first meeting this week
- ▶ There will be a google form shared on Canvas tonight for any significant group logistical issues/concerns. Based on responses, I **may** make minor adjustments to group membership between now and Wednesday's lecture.



grading

- ▶ 30% checkpoints (15% each)
- ▶ 30% / 25% final exam
- ▶ 40% problem sets
- ▶ 0% / 5% group work



► Questions?



Introduction to Haskell – getting set up

- ▶ Problem 2 on week01-group (due Friday) asks each of you to set up Haskell on whatever machine you plan on doing your assignments on.
- ▶ The week01-ps asks you to write a few functions in Haskell and for each of you to turn it in (individually) on gradescope.
 - This assignment must be turned in individually on Gradescope. However **for this assignment only** you may collaborate as much as you want with the other students in your small group.
 - If you ever have any questions about what is an acceptable amount of collaboration (or an acceptable use of ChatGPT or similar programs!), you must discuss with the professor in advance!!!



Introduction to Haskell – getting set up

- ▶ Problem 2 on week01-group (due Friday) asks each of you to set up Haskell on whatever machine you plan on doing your assignments on.
- ▶ The week01-ps asks you to write a few functions in Haskell and for each of you to turn it in (individually) on gradescope.
- ▶ Two ways of interacting with Haskell:
 - ▶ type commands in the interpreter
 - ▶ edit a file and run in the interpreter



Haskell basics

- Some things will feel familiar:

```
ghci> 2 + 15
ghci> 49 * 100
ghci> 1892 - 1472
ghci> 5 / 2
ghci> True && False
ghci> True && True
ghci> False || True
ghci> not False
ghci> not (True && True)
ghci> 1 == 0
ghci> 5 /= 5
```



Haskell basics

- ▶ Some things may feel familiar-ish but are worth thinking about a little more
- ▶ Defining functions

```
ghci> add1 x = x + 1  
ghci> addxy x y = x + y
```

- ▶ Conditionals

```
ghci> cap n = (if n > 100 then 100 else n)
```



Introduction to Haskell – getting set up

- ▶ Problem 2 on week01-group (due Friday) asks each of you to set up Haskell on whatever machine you plan on doing your assignments on.
- ▶ The week01-ps asks you to write a few functions in Haskell and for each of you to turn it in (individually) on gradescope.
- ▶ Interacting with Haskell:
 - ▶ type commands in the interpreter
 - ▶ edit a file and run in the interpreter
- ▶ For the week01-ps assignment due Sunday:
 - ▶ Haskell
 - ▶ editor for writing/modifying code: VSCode, emacs



Running Haskell code

▶ sample interaction

```
Week01 — -zsh • Emacs-arm64-11_2
/Users/tyc04747/Desktop/Reorganized/Classes/2022-2023/cs54-sp23/Content/Week01 >
ghci
GHCi, version 9.2.5: https://www.haskell.org/ghc/  :? for help
[ghci> :l week01-code.hs
[1 of 1] Compiling Main                ( week01-code.hs, interpreted )
Ok, one module loaded.
[ghci> cap 124
100
[ghci> cap 47
47
[ghci> cap -2

<interactive>:4:1: error:
• No instance for (Show (Integer -> Integer))
  arising from a use of 'print'
  (maybe you haven't applied a function to enough arguments?)
• In a stmt of an interactive GHCi command: print it
[ghci> cap (-2)
-2
[ghci> :q
Leaving GHCi.
/Users/tyc04747/Desktop/Reorganized/Classes/2022-2023/cs54-sp23/Content/Week01 >
```

```
week01-code.hs
{-
Week01 sample code.
This is how you get a multi-line comment in Haskell.
-}

-- cap n
-- caps the value of n at 100
cap n =
  if n > 100
  then 100
  else n

-:--- week01-code.hs All L10 (Haskell +2 Interactive ElDoc)
(No changes need to be saved)
```

```
ghci> :l <filename>
ghci> :q
```

Practice questions

```
cap n =  
    if n > 100  
    then 100  
    else n
```

- ▶ Write a function `cap'` that not only caps the upper limit at 100, but additionally evaluates to 0 if `n` is less than or equal to 0.
- ▶ Write a function `pow` that takes two parameters `n` and `k` and returns `n` to the `k`th power. (assume that `k` is guaranteed to be a non-negative integer. do not use the `**` operator)

