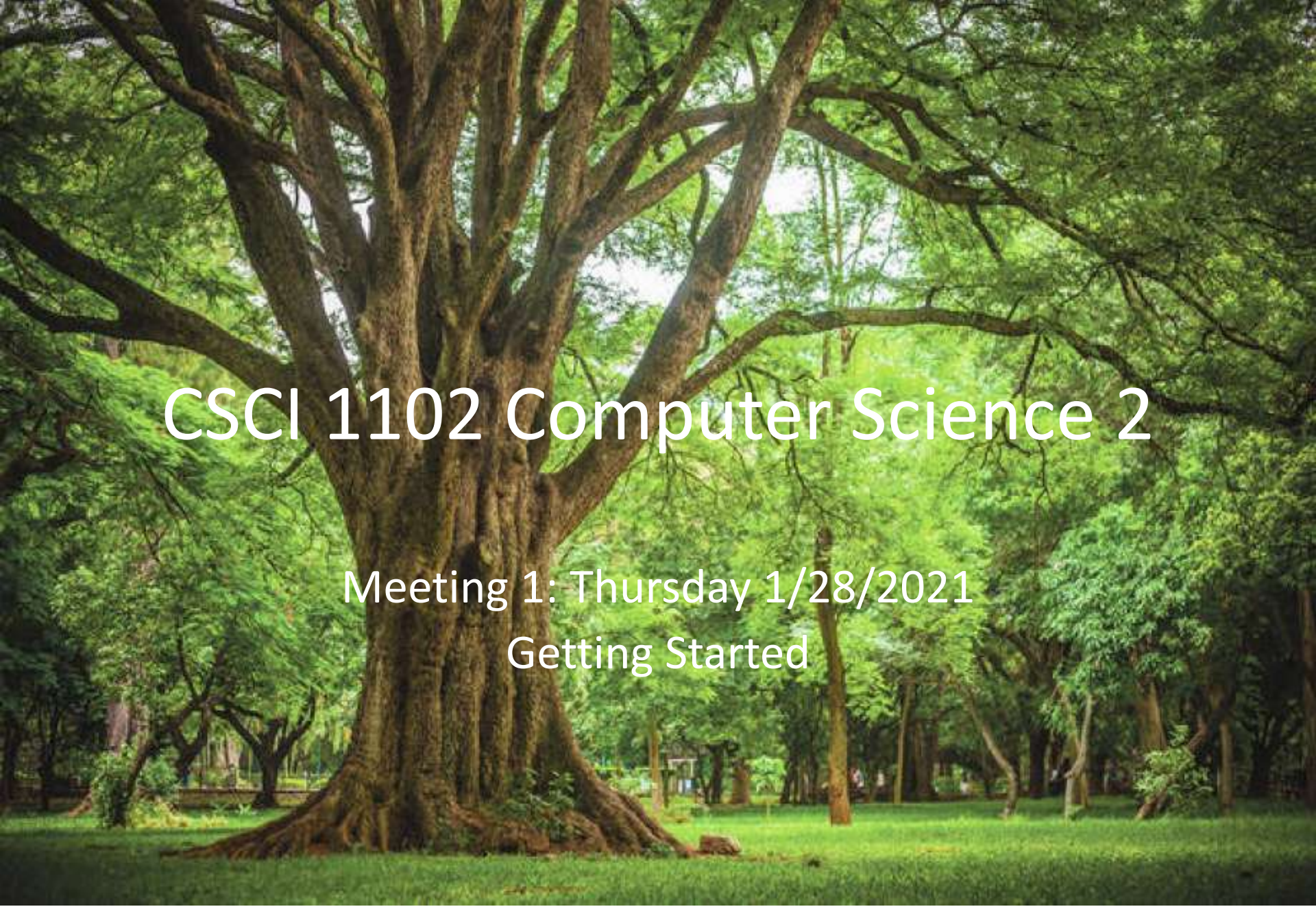


CSCI 1102 Computer Science 2

Meeting 1: Thursday 1/28/2021

Getting Started


CS2
CS1

Two Main Themes

1. Learning classic data structures & algorithms

2. Craftsmanship of software

Data Structures & Algorithms

Example: Data Compression

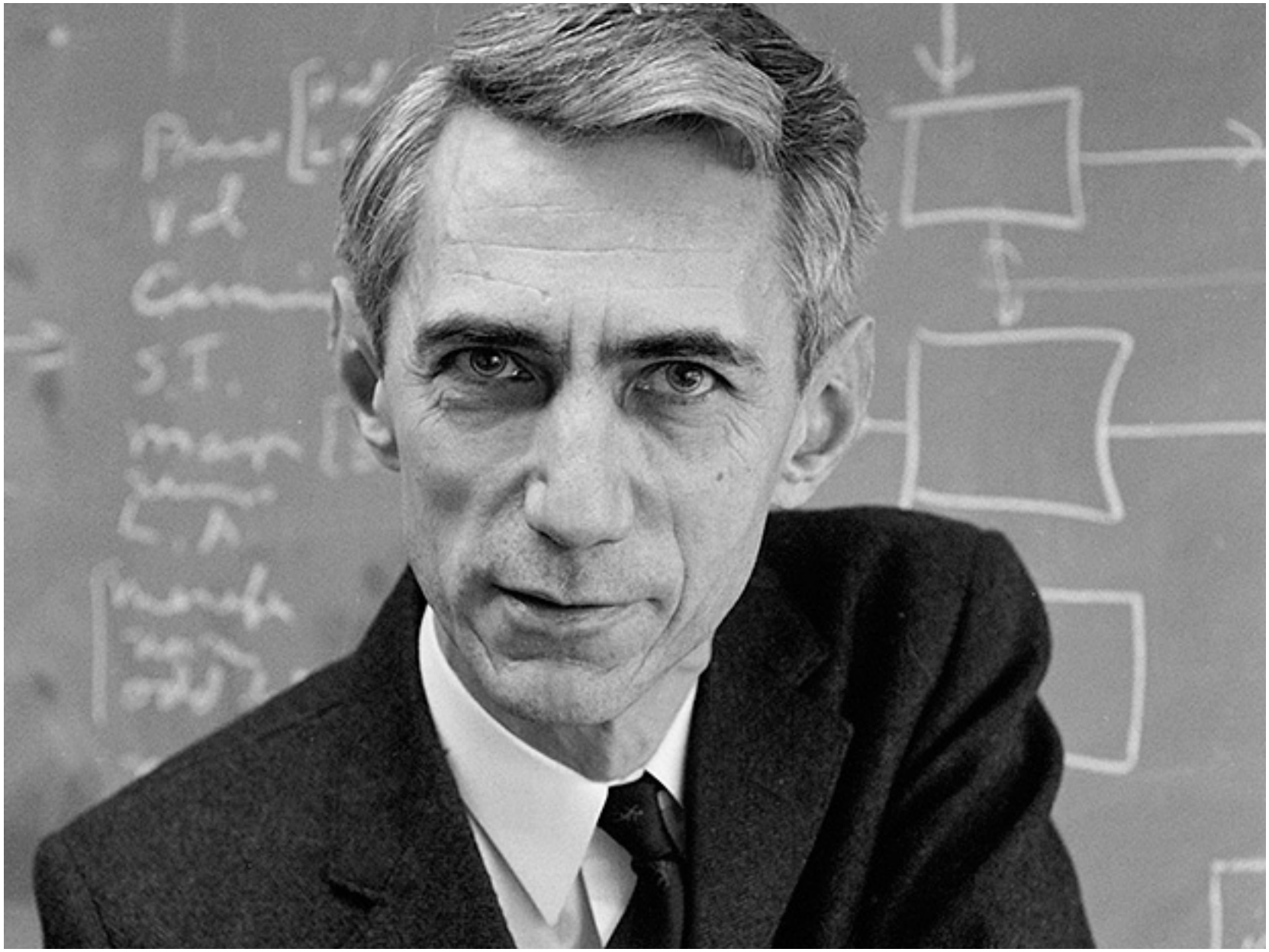
“ALABAMA”

“ALABAMA”

7 letters x 8-bits per letter => 56 bits

“ALABAMA”

‘A’ is run of the mill, ‘B’, ‘L’, & ‘M’
are more *interesting,* i.e., contain more
Information than ‘A’.



Reprinted with corrections from *The Bell System Technical Journal,*
Vol. 27, pp. 379–423, 623–656, July, October, 1948.

A Mathematical Theory of Communication

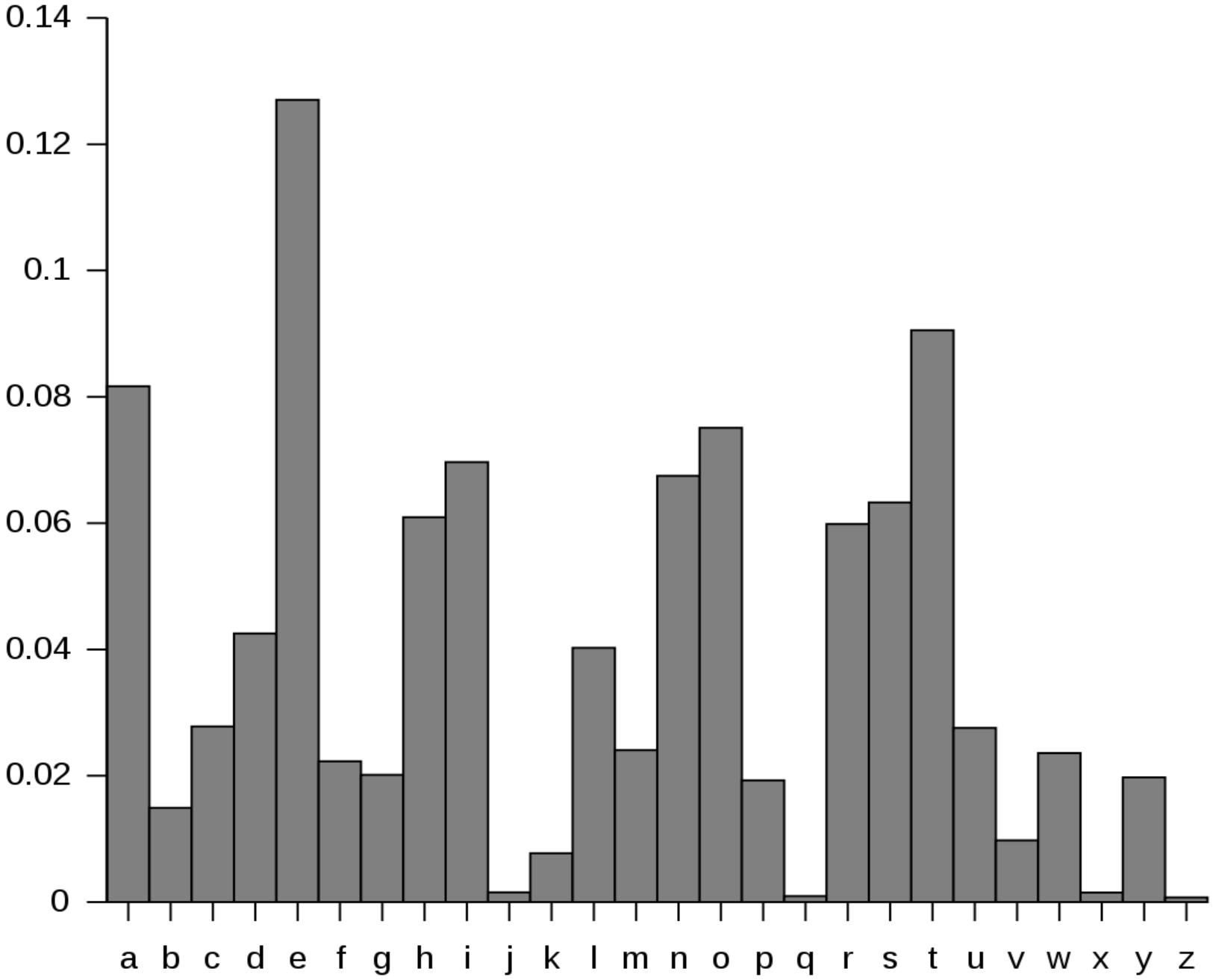
By C. E. SHANNON

INTRODUCTION

THE recent development of various methods of modulation such as PCM and PPM which exchange bandwidth for signal-to-noise ratio has intensified the interest in a general theory of communication. A basis for such a theory is contained in the important papers of Nyquist[1] and Hartley[2] on this subject. In the present paper we will extend the theory to include a number of new factors, in particular the effect of noise in the channel, and the savings possible due to the statistical structure of the original message and due to the nature of the final destination of the information.

The fundamental problem of communication is that of reproducing at one point either exactly or approximately a message selected at another point. Frequently the messages have *meaning*; that is they refer to or are correlated according to some system with certain physical or conceptual entities. These semantic aspects of communication are irrelevant to the engineering problem. The significant aspect is that the actual

Instead of an 8-bit fixed-length code for a letter, choose a variable-length code reflecting the amount of information in the letter.


0.14
0.12
0.1
0.08
0.06
0.04
0.02
0
a b c d e f g h i j k l m n o p q r s t u v w x y z

“ALABAMA”

$A_4\ L_1\ B_1\ M_1$

“ALABAMA”

A_4 L_1 B_1 M_1

A *Frequency Table*

A Queue

A *Priority* Queue

B_1 L_1 M_1 A_4

Low frequency
=> High Priority, head of the queue!
(letters arrived in alphabetical order)

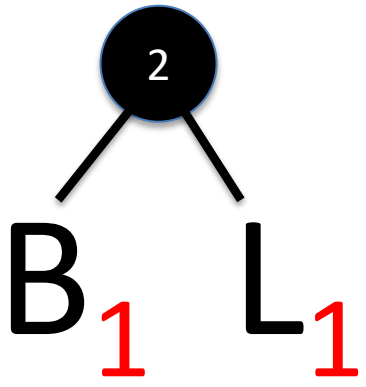
High frequency
$\Rightarrow$ Low Priority, back of the queue!

$B_1\ L_1\ M_1\ A_4$

Algorithm step: while the priority queue has more than one entry, remove two entries, combine them and reinsert the result back into the priority queue.

$B_1\ L_1\ M_1\ A_4$

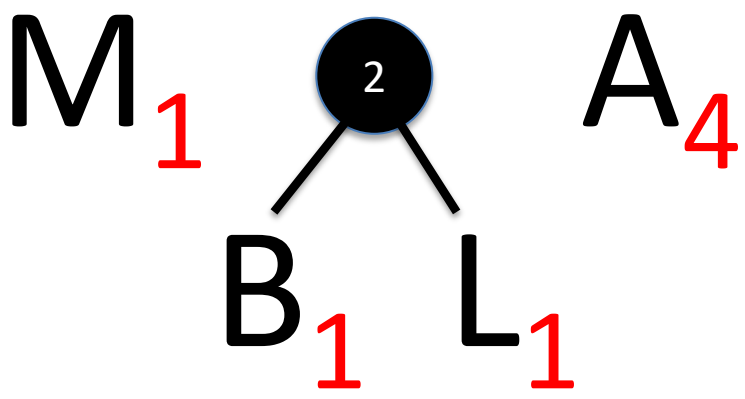
B_1 L_1 M_1 A_4



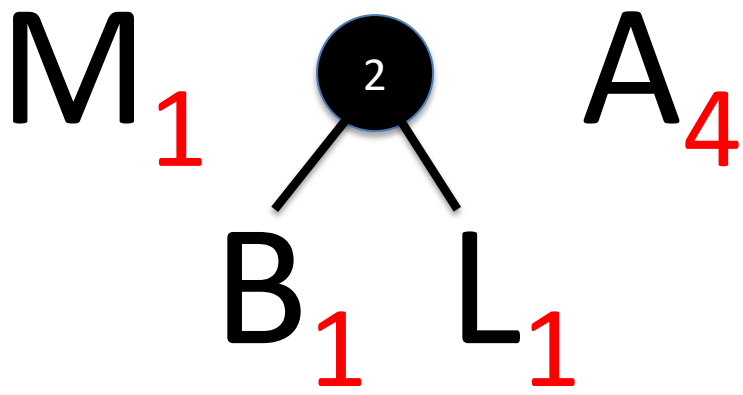
M_1 A_4

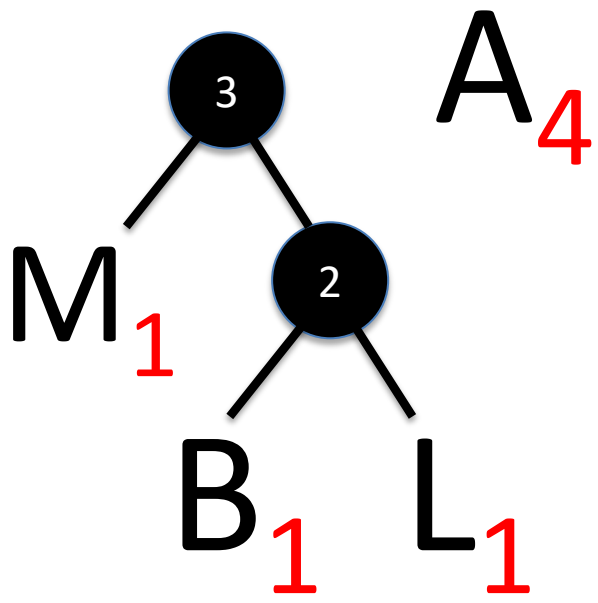
The items in the priority queue are *weighted trees*.

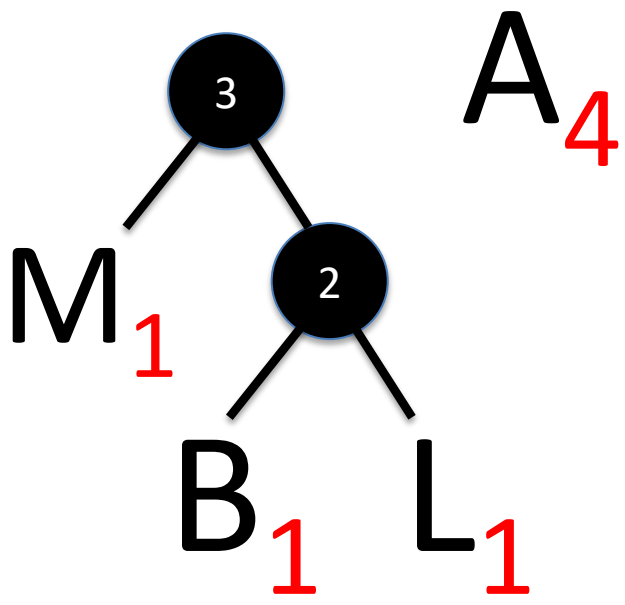
B_1 L_1 M_1 A_4

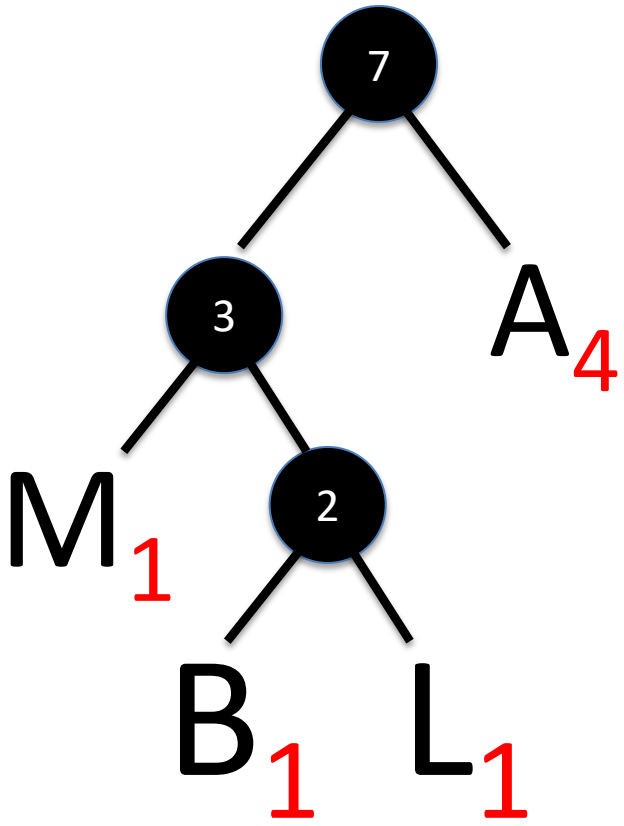


The items in the priority queue are *weighted trees*.

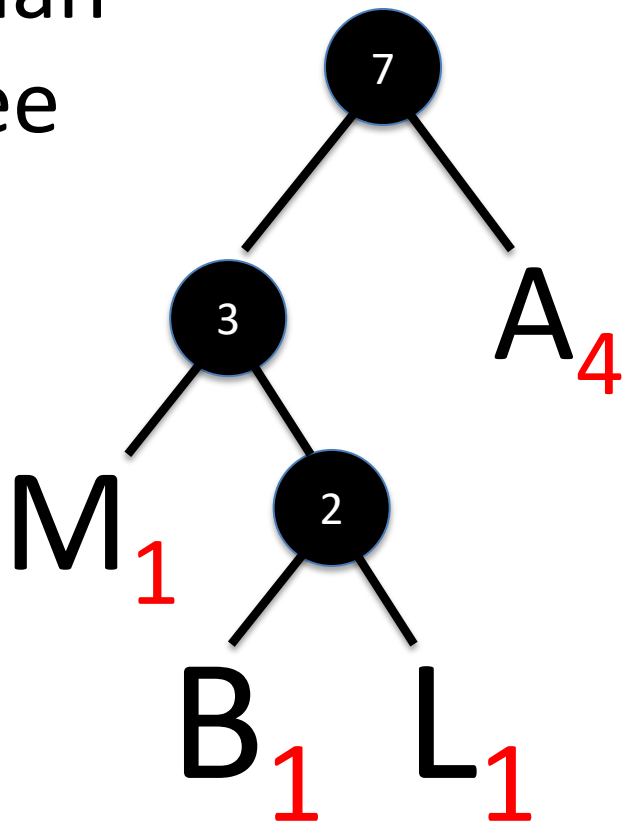

M1
2
A4
B1
L1


3
A4
M1
2
B1
L1

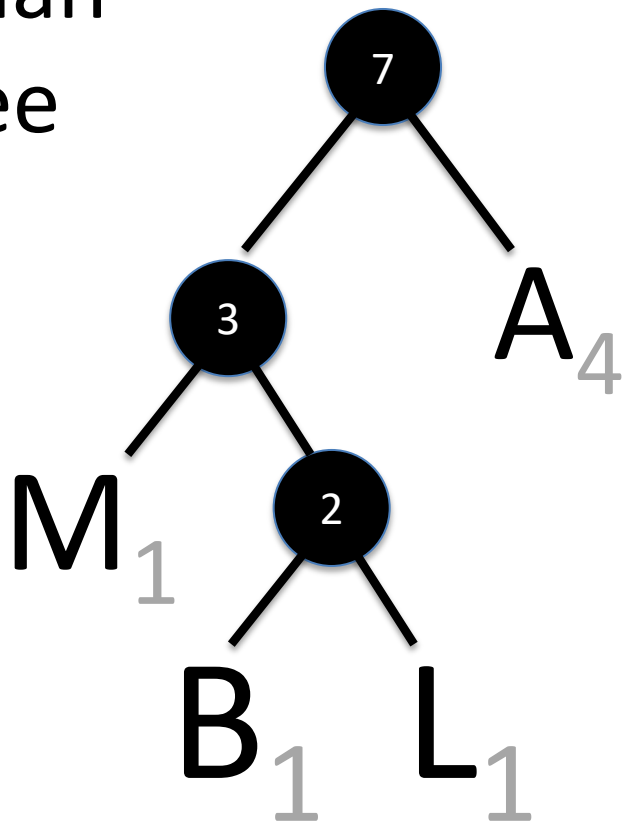

3
2
A4
M1
B1
L1


7
3
A4
M1
2
B1
L1

The Huffman Coding Tree

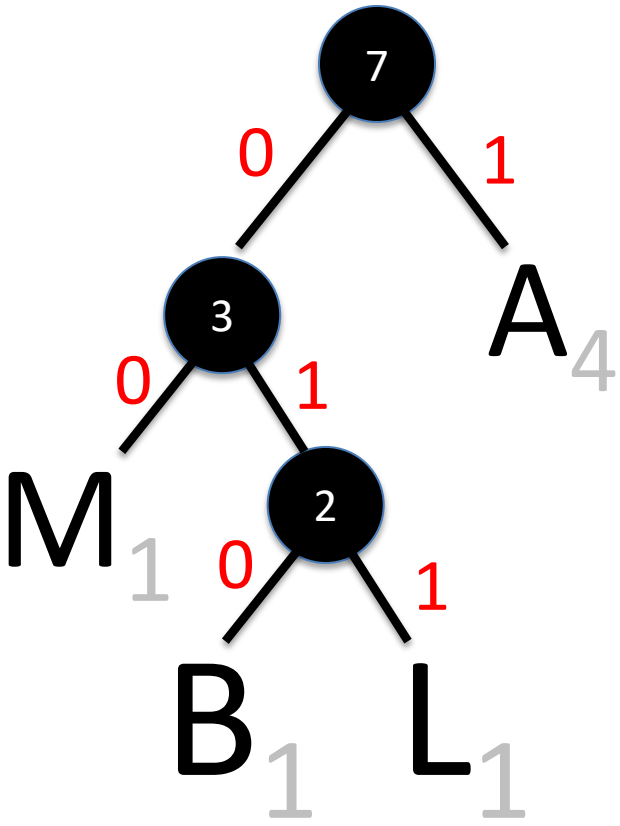


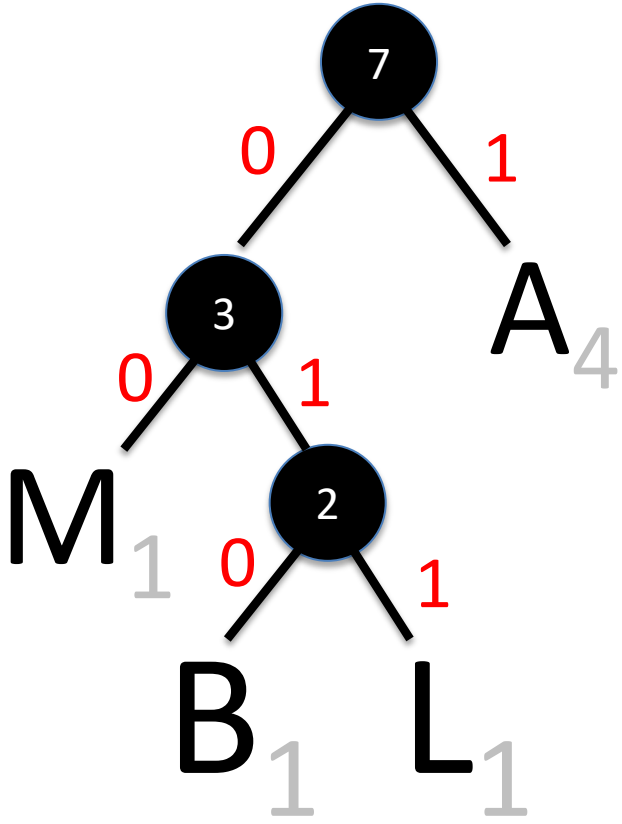
The Huffman Coding Tree



Left = 0

Right = 1


7
0
1
3
A 4
0
1
M 1
2
0
1
B 1
L 1



M = 00, B = 010; L = 011; A = 1

A L A B A M A

1 011 1 010 1 00 1

M = 00, B = 010; L = 011; A = 1

ALABAMA

101110101001

12 bits << 56 bits

M = 00, B = 010; L = 011; A = 1

2. Craftsmanship of software

How to write good code

- Correct
- Performant
- Easy to read
- Easy to write
- Modular
- Maintainable

Data Abstraction

E.g., *Frequency Table*

- Our programming language has types `int`, `real`, `string`, etc, but there is no frequency table type.

- *How to design and develop a new type?*

Abstract Data Types

- Decide what a client ought to be able to do with values of the new type, the operations.
- Decide how to represent the new type so that the operations can be *implemented efficiently*.

Subplot

Trade-offs between block-allocation and linked allocation

Standard Data Structures & Algorithms

- Stacks, Queues, Deques, Priority Queues, Binary Heaps, Binary Search Trees, Balanced Trees, Hash Tables, etc

- Searching and Sorting

What this course is not about

This course uses the Java programming language as a tool, Java is *not* the subject of the course.

This course is not about and does not cover object-oriented programming; in fact we don't use it at all.

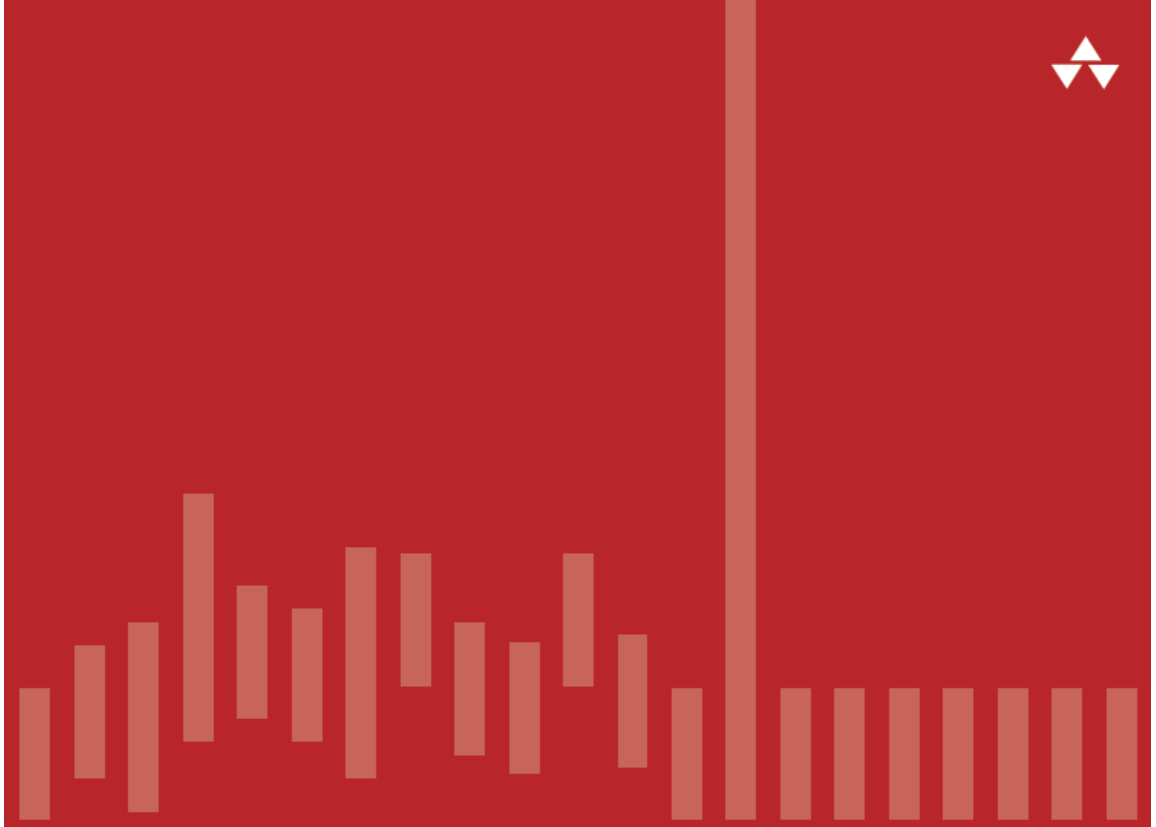
Pragmatics

Who are We?

- Instructor: Robert Muller

- TAs:
  - Callie Sardina, Head TA
  - Kristen Bayreuther
  - Emma Huang
  - Liam Murphy

  We're here to help.



Algorithms

FOURTH EDITION

ROBERT SEDGEWICK | KEVIN WAYNE

algs4.cs.princeton.edu/home/

ALGORITHMS, 4TH EDITION

essential information that
every serious programmer
needs to know about
algorithms and data structures

Online content. This booksite contains tens of thousands of files, fully coordinated with our textbook and also useful as a stand-alone resource. It consists of the following elements:

- *Execepts.* A condensed version of the text narrative, for reference while online.
- *Lectures.* Curated studio-produced online videos, suitable for remote instruction via CUvids.
- *Java code.* The algorithms and clients in this textbook, along with the standard libraries they use.
- *Exercises.* Selected exercises from the book and "web exercises" developed since its publication, along with solutions to selected exercises.
- *Programming assignments.* Creative programming assignments that we have used at Princeton.

You can explore these resources via the sidebar at left.

Textbook. The textbook *Algorithms, 4th Edition* by Robert Sedgewick and Kevin Wayne [Amazon · Pearson · InformIT] surveys the most important algorithms and data structures in use today. We motivate each algorithm that we address by examining its impact on applications to science, engineering, and industry. The textbook is organized into six chapters:

- *Chapter 1: Fundamentals* introduces a scientific and engineering basis for comparing algorithms and making predictions. It also includes our programming model.
- *Chapter 2: Sorting* considers several classic sorting algorithms, including insertion sort, mergesort, and quicksort. It also features a binary heap implementation of a priority queue.
- *Chapter 3: Searching* describes several classic symbol-table implementations, including binary search trees, red–black trees, and hash tables.
- *Chapter 4: Graphs* surveys the most important graph-processing problems, including depth-first search, breadth-first search, minimum spanning trees, and shortest paths.
- *Chapter 5: Strings* investigates specialized algorithms for string processing, including radix sorting, substring search, tries, regular expressions, and data compression.

Joshua Bloch

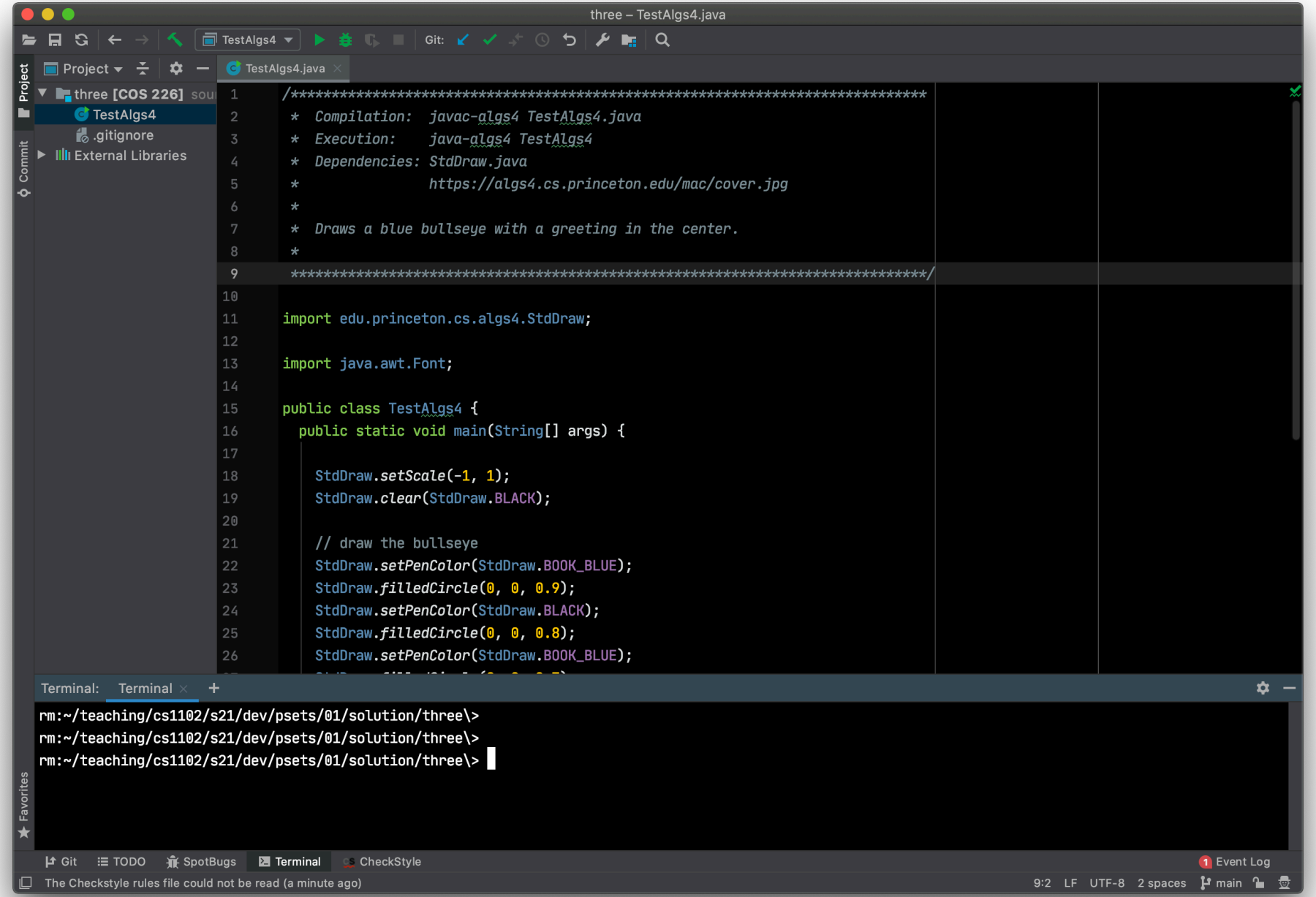
Updated for Java 9

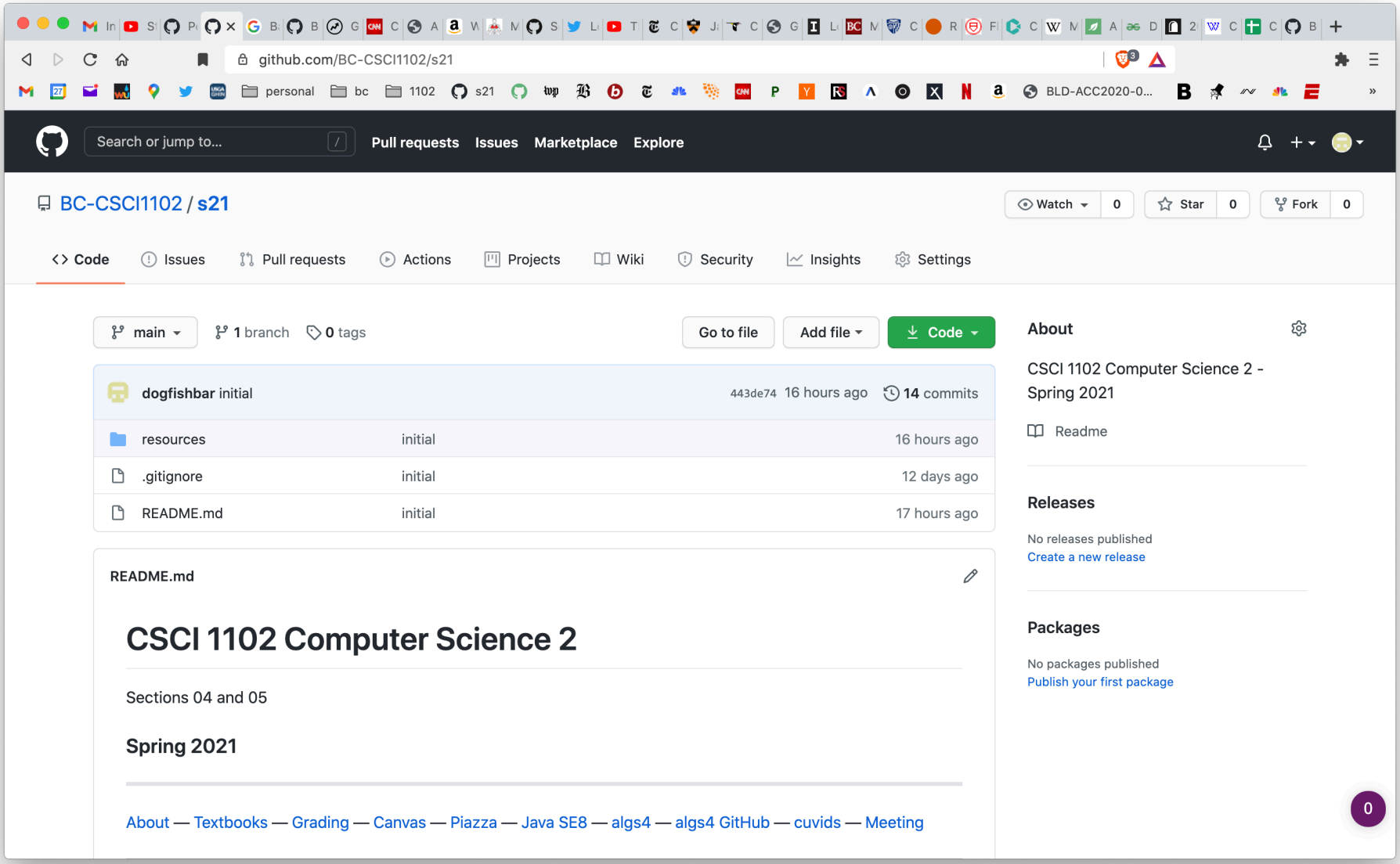
Effective Java

Third Edition

Best practices for

...the Java Platform


three – TestAlgs4.java
TestAlgs4
Git:
Project
TestAlgs4.java
three [COS 226]
TestAlgs4
.gitignore
External Libraries
/***
* Compilation: javac-algs4 TestAlgs4.java
* Execution: java-algs4 TestAlgs4
* Dependencies: StdDraw.java
* https://algs4.cs.princeton.edu/mac/cover.jpg
*
* Draws a blue bullseye with a greeting in the center.
*
***/
import edu.princeton.cs.algs4.StdDraw;
import java.awt.Font;
public class TestAlgs4 {
public static void main(String[] args) {
StdDraw.setScale(-1, 1);
StdDraw.clear(StdDraw.BLACK);
// draw the bullseye
StdDraw.setPenColor(StdDraw.BOOK_BLUE);
StdDraw.filledCircle(0, 0, 0.9);
StdDraw.setPenColor(StdDraw.BLACK);
StdDraw.filledCircle(0, 0, 0.8);
StdDraw.setPenColor(StdDraw.BOOK_BLUE);
Terminal:
Terminal
rm:~/teaching/cs1102/s21/dev/psets/01/solution/three\>
rm:~/teaching/cs1102/s21/dev/psets/01/solution/three\>
rm:~/teaching/cs1102/s21/dev/psets/01/solution/three\>
Favorites
Git
TODO
SpotBugs
Terminal
CheckStyle
Event Log
The Checkstyle rules file could not be read (a minute ago)
9:2
LF
UTF-8
2 spaces
main

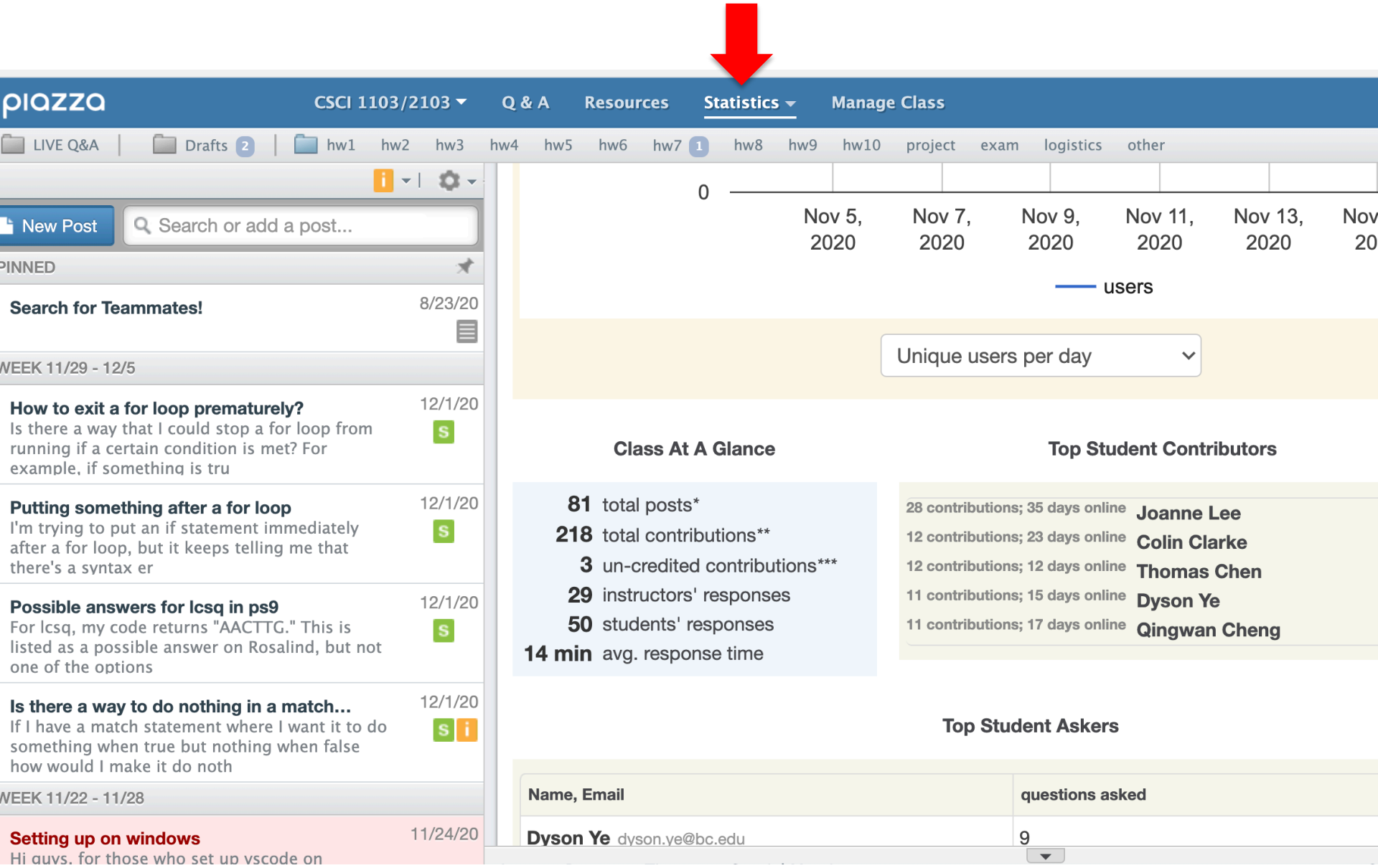

github.com/BC-CSCI1102/s21
Search or jump to...
Pull requests
Issues
Marketplace
Explore
BC-CSCI1102 / s21
Watch 0
Star 0
Fork 0
Code
Issues
Pull requests
Actions
Projects
Wiki
Security
Insights
Settings
main
1 branch
0 tags
Go to file
Add file
Code
dogfishbar initial
443de74 16 hours ago
14 commits
resources initial 16 hours ago
.gitignore initial 12 days ago
README.md initial 17 hours ago
README.md
CSCI 1102 Computer Science 2
Sections 04 and 05
Spring 2021
About — Textbooks — Grading — Canvas — Piazza — Java SE8 — algs4 — algs4 GitHub — cuvids — Meeting
About
CSCI 1102 Computer Science 2 - Spring 2021
Readme
Releases
No releases published
Create a new release
Packages
No packages published
Publish your first package

Work

- Ten software projects
- Three short quizzes
- Reading
- Participation – class & piazza forum

Grading

- Ten software projects – 140 points
- Three short quizzes – 30 points
- Reading
- Participation – 30 points


piazza
CSCI 1103/2103
Q & A
Resources
Statistics
Manage Class
LIVE Q&A
Drafts 2
hw1 hw2 hw3 hw4 hw5 hw6 hw7 1 hw8 hw9 hw10 project exam logistics other
New Post
Search or add a post...
PINNED
Search for Teammates!
8/23/20
WEEK 11/29 - 12/5
How to exit a for loop prematurely?
Is there a way that I could stop a for loop from running if a certain condition is met? For example, if something is tru
12/1/20
Putting something after a for loop
I'm trying to put an if statement immediately after a for loop, but it keeps telling me that there's a syntax er
12/1/20
Possible answers for lcsq in ps9
For lcsq, my code returns "AACTTG." This is listed as a possible answer on Rosalind, but not one of the options
12/1/20
Is there a way to do nothing in a match...
If I have a match statement where I want it to do something when true but nothing when false how would I make it do noth
12/1/20
WEEK 11/22 - 11/28
Setting up on windows
Hi guys, for those who set up vscode on
11/24/20
0
Nov 5, 2020
Nov 7, 2020
Nov 9, 2020
Nov 11, 2020
Nov 13, 2020
users
Unique users per day
Class At A Glance
81 total posts*
218 total contributions**
3 un-credited contributions***
29 instructors' responses
50 students' responses
14 min avg. response time
Top Student Contributors
28 contributions; 35 days online Joanne Lee
12 contributions; 23 days online Colin Clarke
12 contributions; 12 days online Thomas Chen
11 contributions; 15 days online Dyson Ye
11 contributions; 17 days online Qingwan Cheng
Top Student Askers
Name, Email
questions asked
Dyson Ye dyson.ye@bc.edu
9

Challenges

- This damnable pandemic and all that entails.
- Pythonistas: coming to terms with types
- OCamlers:
  - No either/or types or pattern matching;
  - Imperative style, mutability is the default.

First couple of weeks

How to Succeed

- Actually read the Algorithms textbook!
- Start problem sets right away.
- Persist! --- If you're not confused and uncomfortable most of the time, you aren't doing it right.
- Work a little harder to stay engaged.