Welcome to Basic Algorithms

People

Your instructor

Name: Victor Shoup

email: shoup@cs.nyu.edu

Office hours: ONLINE; Tuesdays, 1:15-2:30pm

Your recitation leader

Name: Alex Bienstock

email: afb383@nyu.edu

Office hours: ONLINE; Fridays, 3-4pm

Your tutor

Name: Albert Liu

Office hours: ONLINE; Mondays, 7-9pm; Tuesdays, 3-5pm;
Fridays, 7-9pm; Sundays, 3-5pm

. . . and a host of graders

Grades

There will be about

- 9 problem sets
- 6 programming assignments
- 4 or 5 quizzes

Weights for final grades:

- problem sets: 30%
- programming assignments: 20%
- quizzes: 50%

Course home page

The course home page is

`https://cs.nyu.edu/courses/fall20/CSCI-UA.0310-001/`

Please **bookmark this page**

This is where you will access and download

- lecture slides
- problem sets
- programming assignments
- other handouts

NYU Classes

You will use NYU Classes for

- uploading problem sets
- accessing your grades
- accessing announcements

Instructions for uploading problems sets

- Each individual problem on a problem set will correspond to *one* assignment on NYU Classes
  - For example, on PS 1 (Problem Set 1), there are problems 1, 2, 3, . . .
  - These will correspond to assignments PS1.1, PS1.2, PS1.3, . . . on NYU Classes
- You must upload a *single* ***PDF*** file for *each* problem
  - To repeat: a **single file for each problem**, and
  - that file must be a **PDF file** (and *not* JPG or *anything else*)
- Scanned handwritten (but *legible*) solutions are perfectly acceptable
- You may also use LaTex or other text preparation tools (but this is *not* required)
- Submissions will **not receive any credit** if:
  - they are not submitted correctly as a *single PDF* file,
  - they are not legible, or
  - they are not submitted by the due date (i.e., **late policy = no late problem sets accepted**)

Programming Assignments and Hackerrank

Programming assignments will be done on *Hackerrank*

You should go to `hackerrank.com` and create a user account with a **hackerrank ID** that is **equal to your NYU Net ID**

e.g., “abc123”, and not “N12345678”

You *must* have a Hackerrank ID of this form, even if you already have another Hackerrank ID

very few, if any, exceptions will be made to this policy

The first programming assignment will be due soon, so please get this set up right away

No late programming assignments will be accepted

Grades will be based on how well your program objectively performs: no partial credit for trying

Online quizzes and Gradescope

Instead of a midterm and final exam, we will have several *online quizzes*

These quizzes will be hosted on `gradescope.com`

You will automatically be registered for our class on Gradescope

Each quiz will be *timed*: you will have a fixed amount of time to complete the quiz after you start

Students who have permission for extra time on exams will be given that time

You may take the quiz at any time within a specified 24 hour window

Academic Integrity Policy

Problem sets and programming assignments: students are allowed to discuss general ideas and strategies with each other

But. . . each student should write up their *own* solutions and code

Any blatant copying of solutions or code from other students *or any other source* will result in

- at a minimum, an automatic zero for that work,
- possibly more serious academic consequences

Warning: automated tools will be used that are very good at detecting copying (even if you try to "cover your tracks")

Quizzes: there is a **ZERO TOLERANCE POLICY** of collaboration of any kind, or of the use of any resources other than those explicitly allowed

- *Violations of this policy will be **SEVERE***

Pseudocode

On problem sets and quizzes, you will often be asked to write *pseudocode*

This is usually a mixture of

- high-level English, and
- more low-level algorithmic descriptions, closer to a programming language like Java or Python

Examples of high-level English

- “Sort the input”
- “swap $A[i]$ and $A[j]$”

Recommended style for lower level pseudocode

// Given an array $A[0\,..\,n)$ of integers
// compute largest index i such that $A[i] = 0$
// (or -1 if no $A[i] = 0$)

$lastZero \leftarrow -1$
for i in $[0\,..\,n)$ do
    if $A[i] = 0$ then
        $lastZero \leftarrow i$

// Alternative implementations

$i \leftarrow n - 1$
while $i \geq 0$ and $A[i] \neq 0$ do
    $i \leftarrow i - 1$

$lastZero \leftarrow i$

$lastZero \leftarrow -1$
for i in reverse $[0\,..\,n)$ do
    if $A[i] = 0$ then
        $lastZero \leftarrow i$
        exit for loop

Getting started . . .

First topic: *2-3 trees*, a special kind of balanced search tree

From the course home page:

- *Handouts:* 2-3 trees (notes) [read this]
- *Slides:* 2-3 trees

Second topic: Math Facts (review/intro to some basic math that we'll need)

- *Handouts:* Math Facts (notes) [read this]