JOHNS HOPKINS
WHITING SCHOOL
of ENGINEERING

Syllabus
Computer Science 600.363/463
Introduction to Algorithms / Algorithms I
Fall, 2014
(3 credits, EQ)

Description

This course concentrates on the design of algorithms and the rigorous analysis of their efficiency. Topics include the basic definitions of algorithmic complexity (worst case, average case); basic tools such as dynamic programming, sorting, searching, and selection; advanced data structures and their applications (such as union-find); graph algorithms and searching techniques such as minimum spanning trees, depth-first search, shortest paths, design of online algorithms and competitive analysis.

Prerequisites

Data Structures (EN.600.226)
Discrete Mathematics (EN.550.171 or equivalent)

Instructor

Professor Michael Dinitz, `mdinitz@cs.jhu.edu`, `http://www.cs.jhu.edu/~mdinitz/`
Office: Malone 217, 410-516-7185
Office hours: Mondays 10:00am–12:00pm, and by appointment

Teaching Assistant

TBD

Meetings

Tuesday, Thursday, 1:30–2:50 pm, Shaffer 303

Textbook

Required: Cormen, Leiserson, Rivest, and Stein, *Introduction to Algorithms, Third Edition*, MIT Press (2009).

Online Resources

Course webpage: `http://www.cs.cmu.edu/~mdinitz/IntroAlgorithms/`
Online discussion: `https://piazza.com/jhu/fall2014/600363600463/home`
Email list: `http://www.cs.jhu.edu/mailman/listinfo/cs363`
Blackboard will be used for grades and homework submissions

Course Objectives

(1) Students will learn the basic definitions of algorithmic complexity, and how to analyze the complexity of algorithms.
(2) Students will learn basic algorithmic tools used to design efficient algorithms.
(3) Students will learn how to design efficient algorithms and to recognize situations where this is not possible.

Course Topics

- Basic definitions of algorithmic complexity (worst case, average case).
- Basic tools such as dynamic programming, sorting, searching, and selection.
- Advanced data structures and their applications (such as union-find).
- Graph algorithms and searching techniques such as minimum spanning trees, depth-first search, and shortest paths.
- Design of online algorithms and competitive analysis.

Course Expectations & Grading

There will be homework assignments approximately every week, two in-class midterms, and a final. Grades will be calculated as follows:

Homeworks: 40%
Midterm 1: 15%
Midterm 2: 15%
Final exam: 30%

This class will be graded on a curve, but not a strict one. That is, the correspondence between numeric and letter grades will be determined by the final distribution of numeric grades, but there is no specific letter grade distribution that will be targeted.

You are free to work on the homework in groups of up to 3, but you must write up your solutions entirely on your own. That is, collaboration is limited to discussing the problem, and does not include writing down the solution. Please list the members of your group on your submission.

Key Dates

Midterm 1: September 25
Midterm 2: October 30
Final Exam: December 15

The final exam date is set by the registrar and will not change, but the dates of the midterms are subject to change. If the dates do change, new dates will be announced online and in class.

Assignments & Readings

These will be posted on the course webpage.

Ethics

The strength of the university depends on academic and personal integrity. In this course, you must be honest and truthful, abiding by the *Computer Science Academic Integrity Policy*:

> Cheating is wrong. Cheating hurts our community by undermining academic integrity, creating mistrust, and fostering unfair competition. The university will punish cheaters with failure on an assignment, failure in a course, permanent transcript notation, suspension, and/or expulsion. Offenses may be reported to medical, law or other professional or graduate schools when a cheater applies.
>
> Violations can include cheating on exams, plagiarism, reuse of assignments without permission, improper use of the Internet and electronic devices, unauthorized collaboration, alteration of graded assignments, forgery and falsification, lying, facilitating academic dishonesty, and unfair competition. Ignorance of these rules is not an excuse.
>
> Academic honesty is required in all work you submit to be graded. Except where the instructor specifies group work, you must solve all homework and programming assignments without the help of others. For example, you must not look at anyone else.s solutions (including program code) to your homework problems. However, you may discuss assignment specifications (not solutions) with others to be sure you understand what is required by the assignment.

If your instructor permits using fragments of source code from outside sources, such as your textbook or on-line resources, you must properly cite the source. Not citing it constitutes plagiarism. Similarly, your group projects must list everyone who participated.

Falsifying program output or results is prohibited.

Your instructor is free to override parts of this policy for particular assignments. To protect yourself: (1) Ask the instructor if you are not sure what is permissible. (2) Seek help from the instructor, TA or CAs, as you are always encouraged to do, rather than from other students. (3) Cite any questionable sources of help you may have received.

On every exam, you will sign the following pledge: "I agree to complete this exam without unauthorized assistance from any person, materials or device. [Signed and dated]". Your course instructors will let you know where to find copies of old exams, if they are available.

In addition, the specific ethics guidelines for this course are:

(1) Homeworks may be done in groups of up to three, but you must list your group members on the first page of your submission.
(2) On all assignments each person should hand-in their own writeup. That is, collaboration should be limited to talking about the problems, so that your writeup is written entirely by you and not copied from your partner. In addition, list all members of your group.

Report any violations you witness to the instructor.
You can find more information about university misconduct policies on the web at these sites:

- Undergraduates: `e-catalog.jhu.edu/undergrad-students/student-life-policies/`
- Graduate students: `e-catalog.jhu.edu/grad-students/graduate-specific-policies/`

Students with Disabilities
Any student with a disability who may need accommodations in this class must obtain an accommodation letter from Student Disability Services, 385 Garland, (410) 516–4720, `studentdisabilityservices@jhu.edu`.

ABET Outcomes

- An ability to apply knowledge of computing and mathematics appropriate to the discipline (a)
- An ability to analyze a problem, and identify and define the computing requirements appropriate to its solution (b)
- An ability to design, implement, and evaluate a computer-based system, process, component, or program to meet desired needs (c)
- An ability to function effectively on teams to accomplish a common goal (d)
- An ability to use current techniques, skills, and tools necessary for computing practice (i)
- An ability to apply mathematical foundations, algorithmic principles, and computer science theory in the modeling and design of computer-based systems in a way that demonstrates comprehension of the tradeoffs involved in design choices (j)