Course Information

Instructor	Keith Schwarz (htiek@cs.stanford.edu) Office: Gates 178 Office Phone: (650) 723-4350
TAs	Andy Nguyen (tanonev@stanford.edu) Julie Tibshirani (jtibs@cs.stanford.edu) Kostas Kollias (kkollias@stanford.edu) Phillip Chen (pcchen@stanford.edu) Sean Choi (yo2seol@stanford.edu)

Email The course staff can be reached cs161-sum1213-staff@lists.stanford.edu. Please don't hesitate to send us emails! We're here because we genuinely love this material and want to share it with you. If you have any questions on the material, or if you're interested in exploring more advanced content, please get in touch with us through the staff mailing list. We'd be happy to help out.

Lectures Mondays, Wednesdays, and Fridays, 2:15PM – 3:30PM in Gates B03. Lectures will also be recored and posted online at http://myvideosu.stanford.edu.

Units If you are an undergraduate or SCPD student, you should enroll in CS161 for five units. If you are a graduate student, you may enroll for anywhere between three and five units, depending on what best fits into your schedule. Regardless of how many units you are enrolled for, the course content and requirements will be the same. The unit flexibility is simply to make it easier for graduate students to enroll without exceeding unit caps.

Prerequisites This course has CS103, CS109, and (since it's a prerequisite for CS109) CS106B as prerequisites. These are not “hard” prerequisites – you're welcome to take this course even if you haven't taken these exact courses – but you should be comfortable with the material from these classes.

From CS103, you should feel comfortable writing a formal mathematical proof. You should have exposure to discrete mathematics (graphs, sets, functions, ands relations), and should be familiar with undecidability, the complexity classes **P** and **NP**, and the $\mathbf{P} \overset{?}{=} \mathbf{NP}$ question. From CS109, you should be comfortable working with random variables and their properties – you should know about linearity of expectation, the union bound, and conditional probabilities. Finally, from CS106B, you should feel comfortable writing recursive functions, using fundamental data structures, and basic algorithmic analysis.

If you are unsure whether you have these prerequisites, please feel free to contact the course staff – we'd be happy to help you determine whether this course is appropriate for you!

Website

The course website is http://cs161.stanford.edu and it's loaded with resources for this course. There, you'll find all the handouts for this course, lecture slides, and additional links that you may find useful. I would suggest periodically polling the website to stay abreast of any important developments in the course.

Office Hours

We will try to hold office hours throughout the week and will announce a schedule soon. Additionally, we will try to hold remote help hours for SCPD and remote students, which will be announced soon.

Readings

In addition to lecture handouts, notes, and slides, we will be using the book *Algorithm Design* by Kleinberg and Tardos. A few copies of this book are on reserve at the Stanford Libraries.

Grading

There will be six problem sets given out over the course of the quarter. We hope that these problem sets will give you an opportunity to play around with the ideas and techniques we'll be exploring in lecture. We strongly recommend that you read through all the problem sets as soon as they go out so that you can take the time to think through them.

In addition to the problem sets, there will be a final project that goes out during the final week of class and will be due on **Saturday, August 17** at **12:15PM**. We will provide more information about the final project later in the quarter.

Overall, your grade for this course will be determined as

Problem Sets: 70%
Final Project: 30%

Late Policy

Because of the diversity of the material we'll be exploring, you may find that some homework assignments are harder than others. To give you some extra flexibility, you may submit up to **two** assignments **one class period** past the due date without penalty. You don't need to let us know that you'll be turning in the assignment late, though we'd appreciate a heads-up just so that we know to expect it. While you may use these extensions on any assignments that you would like, **you may use at most one extension per assignment**. Submitting the assignments later than this will make it harder for us to give feedback in time for the next assignment. After you have used up your two automatic extensions, any assignment submitted late will be assessed a 25% penalty. Regardless of whether you've used an extension, **no assignments will be accepted more than one class period past the assignment due date**.

If you have any extenuating circumstances, such as a family or medical emergency, and need extra time to complete the assignments, please contact us. We're more than happy to accommodate. However, **all requests for extensions must be received before the assignment due date**.

Regrades

If you believe that we made an error when grading one of your assignments, please stop by office hours and ask one of the TAs to look over the assignment. If we made an error during grading, we will regrade your assignment as soon as possible. Before arriving, please write a one-paragraph explanation as to why our grading was incorrect.

All regrade requests must be received within **one week** of the date in which the problem set is returned. No regrade requests will be considered after this point, regardless of the severity of the error. (Our intent here is to make sure that you are reviewing the assignment feedback around the time that we get your problem sets back to you.)

It seems strange that I need to write this here, but please treat the course staff civilly. The TAs and I sometimes make honest mistakes when grading, and if that happens we're happy to correct them. However, if you think we've made a mistake, please be courteous when asking for a regrade. **We reserve the right to refuse to regrade your assignments if you do not treat the course staff with respect, even if we have made an error**.

Honor Code

One of our goals in CS161 is to teach you the art of algorithm design. Many algorithms are "nondeterministically trivial:" they are very easy to understand, but difficult to come up with. Consequently, when doing the homework assignments, it is extremely important that you do as much of the work as possible on your own without consulting anyone else or any other outside resources. We hope that the problem sets will give you an opportunity to learn how to devise your own algorithms.

That said, I understand that you may want to work on the problem sets in groups. If you'd like to do this, that's totally fine. However, if you do, **you must write up your own solutions to all of the problems**, and **you must cite all people you worked with on the problem set**. If you do not do so, you may be guilty of plagiarism. Additionally, if you consult any resources other than the course textbook, lecture slides, or lecture notes, **you must cite your sources**. We reserve the right to assign a penalty if your answers are substantially derived from other sources, but as long as you provide citation you will not be committing plagiarism.

Some of the questions on the problem sets may have been used in past quarters (especially if they're really cool problems!) Because of this, **it is a serious violation of the Stanford Honor Code to consult graded problem sets or solution sets from previous quarters**.

Incomplete Policy

If you have a serious medical or family emergency and cannot complete the work in this course, you may contact Keith to arrange for an incomplete. Except for medical or family emergencies that arise in the last week of the quarter, no requests for incompletes will be considered during or after the final week of class. Note that we do not grant incomplete grades for poor performance on the assignments, nor do we grant incomplete grades for busy work schedules.

In order to receive an incomplete in the course, you must have completed all assignments in the course at the time at which you request an incomplete, except for possibly the most-recently-due assignment.

CS161
Summer 2013

Handout #01
June 24, 2013

CS161 Syllabus

This handout contains the tentative syllabus for CS161. Depending on how quickly we're able to cover various topics, we may proceed more quickly or more slowly than the syllabus indicates.

Readings are taken from *Algorithm Design* by Jon Kleinberg and Eva Tardos.

Date	Topics	Readings	Assignments
M June 24	*How do we reason about algorithms?* Course Overview Analyzing an Algorithm: Insertion Sort O, Ω, and Θ notation	2.1 – 2.2, 2.4	
Part One: Fundamental Graph Algorithms			
W June 26	*How do we find shortest paths in a graph?* Breadth-First Search Graph Representations Dijkstra's Algorithm I	3.1 – 3.3, 4.4	**Problem Set 1 Out**
F June 28	*How do we explore graphs?* Dijkstra's Algorithm II Depth-First Search Directed Acyclic Graphs	3.5 – 3.6	
M July 1	*How do we order prerequisites?* Topological Sorting Strongly-Connected Components Kosaraju's Algorithm I	Notes	
W July 3	*What are the applications of graph theory?* Kosaraju's Algorithm II Applications of Strongly-Connected Components	Notes	**Problem Set 1 Due** **Problem Set 2 Out**
Part Two: Divide-and-Conquer Algorithms			
F July 5	*How do we analyze recursive algorithms?* Mergesort Recurrence Relations Binary Search	5.1	
M July 8	*How can we efficiently implement priority queues?* Binary Heaps Heapsort A Lower Bound on Sorting	2.5	
W July 10	*How do we solve divide-and-conquer recurrences?* The Master Theorem Applications of the Master Theorem	5.2, Notes	
F July 12	*How do we solve non-uniform recurrences?* The Median-of-Medians Algorithm The Substitution Method I	Notes	**Problem Set 2 Due** **Problem Set 3 Out**

Part Three: Randomized Algorithms			
M July 15	*Can random guesses improve performance?* The Substitution Method II Quickselect Linearity of Expectation	13.3, 13.5	
W July 17	*How do we precisely analyze randomized algorithms?* Quicksort Indicator Random Variables Finding Large Cuts	13.5	
F July 19	*Can we guess the right answer with high confidence?* Monte Carlo Algorithms Karger's Min-Cut Algorithm	13.2	
M July 22	*How can we efficiently store associative data?* Hash Functions Hash Tables	13.6	**Problem Set 3 Due** **Problem Set 4 Out**
Part Four: Greedy Algorithms			
W July 24	*Can locally optimal decisions be globally optimal?* Interval Scheduling "Greedy Stays Ahead"	4.1 – 4.2	
F July 26	*How do you link up nodes at a low cost?* Minimum Spanning Trees Prim's Algorithm	4.5	
M July 29	*Can we locally modify solutions to build better ones?* Exchange Arguments Dijkstra's Algorithm Revisited Kruskal's Algorithm	4.4 – 4.6	**Problem Set 4 Due** **Problem Set 5 Out**
Part Five: Dynamic Programming			
W July 31	*Can we reuse work across subcomputations?* Linear Independent Set Definition of Dynamic Programming Sequence Alignment I	6.1 – 6.2, 6.6	
F August 2	*How do Internet routers find paths?* Sequence Alignment II The Bellman-Ford Algorithm	6.8	
M August 5	*Can we precompute shortest paths everywhere?* The Floyd-Warshall Algorithm Johnson's Algorithm	6.8 – 6.9	**Problem Set 5 Due** **Problem Set 6 Out**
Part Six: Intractable Problems			
W August 7	*How do you approach intractable problems?* **NP**-Hardness The Knapsack Problem Pseudopolynomial-Time Algorithms	8.1 – 8.5, 6.4	

F August 9	*Can we find simple cases of otherwise hard problems?* Independent Set on Trees	10.2	
M August 12	*Can we guess the answer to hard problems?* Randomized Approximation Algorithms MAX-3SAT	13.4	**Problem Set 6 Due** **Final Project Out**
W August 14	*Where do we go from here?* Further Topics in Algorithms		
Sa August 17	**Final Project Due at 12:15PM** **No Late Submissions**		