H241 Project Assignments

Who's Doing Which Topic:

Chuck Deurschle: NP-Completeness
Ben Dixon: Parsing
Andrew Kipp: Dynamic Programming
Robert Lyon: AI Knowledge Representation
Christopher Pease: AI Search
Matt Smith-Daniels: Program Verification
Kevin Ruble: Network Routing and Robustness
Clarence Duncan: Greedy Algorithms
Wesley Pettyjohn: Operating Systems

References:

For NP-Completeness, Dynamic Programming and Greedy Algorithms:
Cormen, Thomas H.; Leiserson, Charles E.; Rivest, Ronald L.; Stein, Clifford (2001). *Introduction to Algorithms (2nd ed.)*. MIT Press and McGraw-Hill. ISBN 0-262-53196-8

For Operating Systems, Deadlock:
Silberschatz, Abraham; Gagne, Greg; Galvin, Peter B. (2002). *Operating Systems Concepts (6th ed.)* Wiley. ISBN 978-0471250609

For Network Robustness/Routing:
Peterson, Larry L.; Davie, Bruce S. (1999). *Computer Networks: A Systems Approach (2nd ed.)* Morgan Kaufmann. ISBN 978-155860514

For AI Search and Knowledge Representation:
Nilsson, Nils (1998). *Artificial Intelligence: A New Synthesis* Morgan Kaufmann. ISBN 978-1558604674

For Program Verification:
Hamburger, Henry; Richards, Dana (2002). *Logic and Language Models for Computer Science*. Prentice Hall Professional Technical Reference . ISBN: 0130654876

For Compilers, Parsing:
Cooper, Keith; Torczon, Linda (2003). *Engineering A Compiler*. Morgan Kaufmann. ISBN: 978-1558606982

Relevant IU Courses:

NP Completeness: B401

Greedy Algorithms and Dynamic Programming: B403

Networks: P438

Compilers/Parsing: B401, P423

Program Verification: P415

AI Search: B351, B355

AI Knowledge Representation: B351, C311 (Logic Programming)

Operating Systems/Deadlock: P436

Grading

(Detailed guidelines and advice for each item will be posted to the H241 site well before the due date.)

15% Present reading to H241 recitation

30% Problem set over reading material.

25% A small, self-defined "try-it-out-yourself" exercise experimenting with concepts from the reading. (This can be pretty simple, or more ambitious, whatever you want).

30% 1-2 page handout and 15 minute lecture over your topic during class the last week of C241.

Important Dates (Last Updated 10/14/09)

9/11/09: Sneak Preview of Relevant Discrete Topics (important for all projects except program verification).

10/26/09 (Monday): First week presenters email me their example problems (or hand them to me in lecture if pencil and paper is simpler): Robert Lyon (AI-Knowledge Based Systems), Chuck Deurschle (NP-Completeness), Matt Smith-Daniels (Program Verification), and Christopher Pease (AI-Search).

10/30/09 (Friday): First week presentations: Robert Lyon (AI-Knowledge Based Systems), Chuck Deurschle (NP-Completeness), Matt Smith-Daniels (Program Verification), and Christopher Pease (AI-Search) (15%)

11/2/09 (Monday): Second week presenters email me their example problems (or hand in during lecture): Ben Dixon (Parsing), Kevin Ruble (Network Routing and Robustness), Andrew Kipp (Dynamic Programming), Clarence Duncan (Greedy Algorithms), Wesley Pettyjohn (Operating Systems).

11/6/09 (Friday): Second week presentions: Ben Dixon (Parsing), Kevin Ruble (Network Routing and Robustness), Andrew Kipp (Dynamic Programming), Clarence Duncan (Greedy Algorithms), Wesley Pettyjohn (Operating Systems) (This will involve bending space-time, or more likely, shifting to a neighboring room for the last presentation. If you are not intending to use the projector and are interested in going last, please let me know.) (15%)

11/20/09: Problem set due, along with a paragraph describing your plans for the "try-it-yourself" problem (30%).

12/04/09: Turn in and get comments on your first draft 1-2 page hand-out for doing 15min teaching during C241. Also show off your "try-it" problem to your fellow H241 students.

12/07/09-12/09/09: Distribute your hand-outs and teach class for C241 (30%).

12/18/09 (final exam): Last day to turn in your work on the "try-it" problem (25%).

Detailed Milestone Descriptions

H241 Reading Presentation (10/30-11/4)

You will have roughly 30 minutes to present your favorite selection of key concepts/techniques that you've learned about in your reading. You're welcome to consult outside sources if you like, but you should be able to learn everything you need from your reading packet. You should be prepared to: (1) Give a short 5 minute summary of your topic area, including a very brief outline of what problems in what applications it's intended to solve (why/where/how is your topic useful?) and what types of discrete math are necessary for it (logic? time complexity?). (2) Give a sequence of roughly 3-5 example problems that illustrate key concepts and techniques in your area. You may choose examples from the reading, or create your own; however, you must thoroughly understand both the problems and the solutions, and be ready to answer questions about them. You should try to begin with very simple examples and build to more complex ones.

Your audience is your fellow students and you should do your best to make sure your explanations are completely clear to them (you can assume basic knowledge of time complexity, and anything we've already covered in class). You are welcome to create a power point presentation (this is especially useful for displaying important definitions, graphs/visual aids, or outlines), but be very careful that this does not limit your ability to demonstrate solutions in an easy to follow manner. Simply showing one slide with the problem and the next with the solution, with little explanation as to how one gets between the two, is not a useful presentation style. However, we're lucky enough to have access to a room in which you can simultaneously use both the projector and the whiteboard for your presentation, if you desire.

In general, there is a lot of information in your readings, and it's unlikely that you'll be able to cover all of it in 30 minutes. You should choose an approachable subset of interesting and useful points to discuss (of course, making sure that you do cover several of the main/key concepts from the packet). You'll have an audience consisting of your fellow h241 students along with any c241 students who are interested in donuts and extra-credit... and good, interesting talks can be a lot of fun for everyone. You should pick out which topics and example problems you want to cover, think carefully about how you're going to explain them (work on figuring out explanations that are both simple and clear; feel free to use smaller sub-examples, questions, analogies, images, etc. to illustrate your points), and then time yourself to make sure the presentation won't take too long. If it does, consider dropping or changing an example problem, or refining your explanations more. Please let me know if you have any problems with either your reading or presentation.

H241 Problem Set (Friday 11/20)

Please choose a good, representative set of 5-9 problems from your text to solve and turn in. Do not use the same problems you used as example problems for your reading presentation (unless there are few problems available in your text–please contact me if you think this will be an issue). If your problems tend to be fairly simple, I'd like to see you do closer to 9; if your problems tend to be very dense/complex/challenging, then closer to 5 is fine. The overall problem set should be of a reasonable difficulty level and cover the key concepts of the material (if you choose all very simple problems, or only ones from a single topic, I may ask you to do a few additional problems). A sub-problem (as in "part(a)", "part(i)", etc...) can be counted as it's own problem if it's... substantial enough to be counted as its own problem. Use your own best judgement, ask me if you have questions.

Your write-up should be very neat and easy to read (extra-credit for LaTeX!). Please turn it in *directly* to my mailbox in the main office, room LH215 (walk into the office, go left around the corner with the chairs, find my mail slot and place it there). Do *not* use the homework drop box. It occasionally eats things, and they are not heard of again. The office will close at 5pm.

In addition, please include a brief description of what you'd like to do for the "try-it-yourself" problem. Think of a way to try out the material in the text for yourself. You could program an algorithm from the text and try running it (maybe compare it to a less efficient algorithm), you could make up your own variation on an algorithm in the text (how might we want to change this algorithm if the conditions or assumptions were a bit different? could we make it more efficient if we knew certain constraints held? what sort of conditions make the algorithm run really inefficiently and could we treat those cases differently?), you could try using the ideas/tools in the text on an interesting example problem that you come up with, you could find an interesting related problem in another text and try solving it, etc... Anything that lets you play with some of the ideas in the text to make sure you really understand them. It doesn't need to be very complex or work-intensive, but you should try to look for something that sounds a bit fun and interesting to you. You'll have until the final exam to turn it in to me, although you'll have a chance to show it off to your fellow H241 students on 12/4. If you'd like any help thinking of possibilities, just bring your packet by my office hours.

Two-Page Handout (Friday 12/4)

As you know, you will be giving a 15 minute presentation to the general C241 class during the last week of classes. Along with your presentation, you'll be passing out a simple two-page handout. These will give students a concrete introduction to your topic which they can read after the presentation, along with information about where to go if they want to learn more. It will likely also be used to introduce future C241 classes to your project topics.

The handout will include an overview of your topic, one solved example problem, and one unsolved homework question. Everyone in the class (including you) will get extra credit for each homework question they solve correctly, and so long as you don't make your question inanely trivial, it's up to you how difficult you want it to be. The handout can be type-set, or *neatly* handwritten. Feel free to use text/example problems taken from your presentation or problem set. We will meet during H241 recitation on Friday 12/4 (this friday) for a peer-review of your handouts.

Important: ***On the day you present, you will be responsible for bringing 42 printed copies of your final-draft handout with you. It is very important that you do this. Do not underestimate the potential chaos inherent in printing out a handout. Please either print double-sided, or plan on doing some stapling.***

Handout Components:

Topic: Write your topic at the top center of the first page.

Overview Information: Include this information, in this order, with these labels, in the upper half of your first page.

Presenter: [Your Name]

Relevant Discrete Math Concepts: [A list of concepts we've covered in class that are applicable to your topic–possibly logic, combinatorics, tree traversals, graphs, time-complexity, induction, relations, etc...]

Description: [An very general description of what your topic is and why/where it's useful; this is the same information you start your presentation with. Should be one medium paragraph long.]

Reference: [The name, author, and date of the book your packet came from. I'll get those to you.]

Learn More: [a list of IU courses which cover your topic and/or similar material (I'll give you this too)], [also any outside references you've found useful that you'd like to pass on].

Example Problem Statement: On the lower half of your first page, you'll write the problem statement for your example problem. Pick a medium-difficulty problem which you think will give future students a good taste of your topic. Pick something that you can give a detailed, complete solution for in a single page (so it can't be *too* complex.) If your problem statement includes any words which the students wouldn't recognize, make sure you define them clearly (or figure out how to avoid using them).

Example Problem Solution: On the second page, write a clear detailed solution for the example problem. Remember that this isn't like solving a problem on your

homework; here you're trying to teach other people how problems like this are solved. This should be what you wished someone had handed you at the beginning of the semester, instead of just the giant packet.

Homework Question On the bottom of the second page, write a short homework question that should be answerable just by reading the rest of your handout. For example, you could ask students to solve another problem which is similar to the example problem, or ask them some question about how the solution works or how the solution might be different if the problem were changed slightly. Everyone in the class will get extra credit for each homework question they answer correctly (so long as they didn't write it). I'm trusting you not to tell your friends the solution to your problem.