CS 173: Discrete Structures

Viraj Kumar Margaret Fleck

Lucas Cook, Samer Fanek, Chen Li, Lance Pittman, Dan Schreiber

Rick Barber, Efe Karakus, Adair Liu, Andrew Pikler

What is CS 173 about?

- We're going to learn:

1. Basic mathematical objects and techniques
   - sets, functions, relations, graphs, trees, …
   - stuff you will encounter again and again in later CS classes

2. How to read and write mathematical proofs
   - particularly proofs by induction

- Why?

1. "I wrote a program to solve the problem"

   Are you sure it *always* works? Is it *efficient*?

2. "There's a bug in my program"

   Can you reason your way to a solution?

3. "Here's what my program does"

   Can you describe your code clearly?

Am I ready for CS 173?

- You’re ready for this course if you have:
  - At least a C– in both CS125 and Calculus I (Math 220, 221 or 234)
  - ECE190 or CS101 are good in place of CS125
  - A really good grade in CS105 or INFO103 might be OK too

- A quick check: can you read pseudo-code?

```
transmogrify(integer n)
  if n = 0 then
    return 0
  else
    return 1 + transmogrify( $\lfloor n/2 \rfloor$ )
```

- If you’re unsure, please CHECK with us (kumar@illinois.edu)

Can I proficiency out of CS 173?

- Do you know proofs?
  - Can you write a proof by contradiction?
  - An inductive proof?

- Do you know these mathematical concepts?
  - What does it mean for a function to be one-to-one?
  - For a relation to be an equivalence relation?

- Look through the on-line Lectures page from Fall 2009

- If you want to take the proficiency exam, send email to Margaret Fleck (mfleck@cs.uiuc.edu)
  - Exam will be 2-4pm this coming Saturday, in 3401 Siebel
- You may be interested in taking the honors add-on (CS 196)
  - see the web page for instructions

Website and Newsgroup, Textbook

- Routine announcements (homeworks, reading assignments, etc.) are made on:
  - the web page (http://www.cs.uiuc.edu/class/sp10/cs173) and
  - the class newsgroup (class.sp10.cs173)
- Only critical matters (e.g. exams) are announced in lecture
  - Read the newsgroup and watch the website on a regular basis
- Discussion sections and Office hours start next week
- Textbook: *Discrete Mathematics and its Applications*, by Kenneth Rosen, 6th edition (earlier/international editions should be ok)
- The website lists the sections we'll be focusing on
  - Just the stuff we cover in lecture/homeworks

Homeworks

- Homeworks released Friday, due the following Friday (4pm, no later!!)
  - HW 1 out this Friday, most of the material will be familiar
  - some new stuff will be covered on Friday

- Feel free to works in groups (2-3 students, any section/lecture)
  - but you must submit **your own work**

- Cheating Policy:
  - HW solutions can be similar, especially within a group
  - You must identify your group members clearly
  - Its OK to use references (books, Wikipedia, etc.) but cite them
  - Cheating on a HW will result in a zero AND an overall grade penalty

- Put your name and section at the top of Page 1, and submit in the dropboxes (basement of Siebel)

Exams and Quizzes

- Two Midterm exams (evening?) and one cumulative Final

- Three in-class quizzes

- If you have special needs, please let us know as soon as possible

- All scores will be posted on Compass

- You can attend *any* of the CS173 office hours
  - details posted on the website by next week

CS 173 in action

- "Here are a bunch of numbers, I want to know the largest."

- Idea: Pretend the first number is the largest *L*, then compare *L* against all other numbers, updating *L* as necessary

```
largest($a_1$, $a_2$, $a_3$, …, $a_n$ : array of n numbers)
  L = $a_1$
  for i = 2 to n
    if $a_i$ > L then
      L = $a_i$
  return L
```

- We'll see how to prove that this will always work
  - mathematical guarantee

CS 173 in action, again

- "Here are a bunch of numbers, I want to know the second largest."

- Idea: Find the largest *L*, find the largest of numbers that aren't *L*
  - What if the largest number appears twice in the "bunch"?
  - Is the "bunch" a set or a multi-set?

```
secondLargest($a_1$, $a_2$, $a_3$, …, $a_n$ : array of n numbers)
  L = largest($a_1$, $a_2$, $a_3$, …, $a_n$)
  S = $a_1$
  for i = 2 to n
    if ($a_i \neq$ L) and ($a_i$ > S) then
      S = $a_i$
  return S
```

- Is this correct?
- Can we find the second largest faster?