CIS 170: The Science of Computing

http://www.cs.uoregon.edu/classes/08S/cis170

What is “computer science”?
Course outline
lecture topics
projects
exams
Getting started

Course Information

- Professor:
  - John Conery
    conery@cs.uoregon.edu

- GTF:
  - Victor Hanson-Smith
    victorhs@cs.uoregon.edu

- All the information you need (announcements, syllabus, schedule, lecture notes) is online at the class website:
  - http://www.cs.uoregon.edu/classes/08S/cis170

- Use Blackboard to download labs, submit completed labs, and check grades:
  - https://blackboard.uoregon.edu

About the Instructors

- John Conery
  - professor of computer and information science
  - 25th year at UO
  - research: programming languages, parallel processing, bioinformatics
  - teaching: mostly 400/500/600 level CIS classes on computer architecture and scientific applications

- Victor Hanson-Smith
  - Ph.D. student in computer and information science
  - M.S. from UO
  - research interest in bioinformatics, parallel processing
  - was GTF for CIS 170 last term

Today’s Lecture

- CIS 170 is a general introduction to computer science
  - intended for non-majors and pre-CIS majors

- Today’s lecture:
  - what is computer science?
  - what parts of computer science will we study this term?
  - course organization
  - projects, papers, exams, grading policy
What is Computer Science?

- Computer science is
  - engineering
  - math
  - cognitive science
  - linguistics
  - business

Computer Science is Engineering

- Engineers design and build artifacts
  - based on scientific knowledge of the area
  - evaluate cost/benefit tradeoffs
  - use professional standards
- Examples
  - structural engineering (buildings, bridges, ...)
  - electrical engineering (power grids, electronics, ...)
- **Computer engineering**: design and construction of computer systems
  - logic chips, power supplies, fans, packaging, ...
- **Software engineering**: design and implementation of computer applications
  - analyze requirements, specify functionality, write code, test
  - what science supports this field of engineering?

Computer Science is Math

- Mathematicians deal with formal systems
  - logic; deductive and inductive proof
  - algebraic formulas, transformations
  - abstractions that may have many applications
- Algorithms are formal descriptions of methods for solving problems
- Computer scientists often prove properties of algorithms
  - ex: common methods for sorting \( n \) items can take \( n^2/2 \) steps or \( n \times \log_2 n \) steps

<table>
<thead>
<tr>
<th></th>
<th>( n^2/2 )</th>
<th>( \log_2 n )</th>
<th>( n \times \log_2 n )</th>
</tr>
</thead>
<tbody>
<tr>
<td>poker hand</td>
<td>5</td>
<td>12</td>
<td>12</td>
</tr>
<tr>
<td>bridge hand</td>
<td>13</td>
<td>85</td>
<td>48</td>
</tr>
<tr>
<td>full deck</td>
<td>52</td>
<td>1352</td>
<td>296</td>
</tr>
</tbody>
</table>

- Insertion sort
- Merge sort

Computer Science is Cognitive Science

- Cognitive psychologists study mental processes, e.g.
  - memory
  - perception
  - decision making
- Computer scientists use concepts from cognitive science when designing software
  - graphical user interfaces (organized so users find them easier to work with)
  - scientific visualization (summarizing, displaying large amounts of data)
- Artificial intelligence aims to build software to do many things humans do
  - machine learning (e.g. discriminate cancerous from normal cells)
  - planning (most effective allocation of scarce resources)
  - natural language processing (e.g. machine translation)
Computer Science is Linguistics

- The study of computer languages has been influenced by many ideas from linguistics
  - grammars and grammar rules
  - parsing (e.g. "diagramming" a sentence)
- Computer science applications include
  - regular expressions and pattern matching
  - analysis of DNA and protein sequences
  - natural language processing

Computer Science is Business

- Concepts from economics and business administration play a big role in computer science
- Software engineering:
  - understanding customers and their requirements
  - end-user support (training, documentation, ...)
  - group projects, cooperative design
  - OMSE: approximately 1/3 MBA style courses
- Custom vs commodity components
  - Linux "cluster" vs SMP multiprocessor

The Science of Computing

- The "science" in computer science includes
  - algorithms: what are the most efficient methods for solving problems?
  - languages: what are the best ways to express algorithms?
  - software engineering: how can we build useful and reliable programs?
  - computer engineering: how can we build cost-effective computer systems?
- Computer science helps people solve problems
  - science
  - engineering
  - medicine
- Computer science helps people be more effective or creative
  - architecture
  - communications
  - music and the arts

Computer Science Research

- Universities
  - UO, OSU, PSU, nearly every university in the US and around the world
- Government labs
  - Department of Energy (PNL, Sandia, Oak Ridge, ...)
  - National Institutes of Health (NLM, ...)
- Industry labs
  - IBM, Xerox, Bell Labs in the 1970s and 80s
  - Microsoft, Intel, IBM, Google, ...
  - Pharmaceutical companies, automobile industry, ...
CS in a Liberal Arts University

- At the University of Oregon the CIS Department is part of the College of Arts and Sciences
  - at many other universities CS is in Engineering (e.g. EECS, CSE, ...)
  - many universities have also established a College of Computing, where CS is just one of many departments in a separate college
- At UO CIS majors take
  - 54 credits CIS
  - 32 credits Math
  - 12 credits science (physics, chemistry, biology, or psychology)
  - Technical writing
- Our philosophy: critical thinking and an interdisciplinary perspective are just as important as technical skills

Why This Course?

- Where does CIS 170 fit into the computer science curriculum at UO?
- The goals for CIS 170 are to give students an introduction to computer science
  - a brief introduction to key ideas
  - projects on selected topics
- Motivation:
  - fun, interesting science course for non-CS majors
  - broad overview of the field for pre-CS majors

Theme for SQ’08

- Because CS is such a large and complex subject it would be hard to cover each area in a 10-week course
- Instead we choose a theme for each term
  - readings and projects are related to the common theme
  - SQ’06 and WQ’07 (Conery): bioinformatics
    - algorithms on strings, application to DNA and proteins, sequence databases, ...
  - WQ’08 (Fickas): the future of computing
    - DNA computers, quantum computing, computers and medicine, ...
- SQ’08 (Conery): electronic voting
  - a critical look at the technical issues surrounding proposals for voting machines
    - what is a fair election?
    - is there a role for computer technology to ensure fair elections?
    - what are the social impacts (pros and cons) of various alternatives?

Textbook

- The textbook this term is
  - Computer Science: An Overview, by J. Glenn Brookshear
  - a broad introduction to a wide variety of CS topics
  - chapters on core topics plus artificial intelligence, databases, graphics, and more
  - we won’t have time to go into all of these topics...
- One of the reasons I chose this book:
  - sections in each chapter on social impacts and ethical issues related to computer science
Course Outline

- Some of the topics from CSO we plan to cover include:
  - data representation
    - how are numbers, strings, songs, pictures and other things (votes?) stored in a computer?
  - algorithms
    - how do machines process the data stored in their memories?
    - are there limits to what a machine can do?
  - computer networks
    - how is data transferred between systems?
    - how does the internet work?
    - security, privacy, and other issues facing electronic voting
  - databases
    - how secure are voter registration databases?
  - user interface design
    - electronic voting machines must be accessible and usable by every voter

On-Line Resources

- In addition to the textbook, you will find a lot of course information on-line
  - the "resources" link on the class web page will take you to a page that has links to
    web sites with related information
  - there is also on-line reference material for the software we will be using
  - this web site is still evolving -- send me suggestions if you find anything interesting

- Blackboard has facilities for group discussions
  - for our class: you can post questions, answer other students’ questions, ...
  - I'm still learning how to use it -- if you post a question please e-mail me to remind me to log in....
  - I may also start a blog where you can post comments and get a discussion going

Labs

- There will be a set of homework assignments in the form of “labs”
  - paper-and-pencil assignments, similar to those in a math class
  - short projects that involve the use of software to explore that week’s topic
  - example: algorithms on strings
    - problem set will have questions about patterns and substring searches
    - project will use software that searches for specified patterns in strings (some from a local file, some from the internet)

- Analogy -- labs in chemistry classes
  - instructor prepares the materials and methods
  - you follow instructions, write up a report and submit it
  - an advantage of computer labs: no nasty smells or messy explosions...

Exams

- There will be a final exam

- The best way to study for the exam: do the problem sets and projects!
  - the “short answer” questions in labs will be very similar to the sorts of questions that will be on the exam
Grading Policy

- Your grade will be based on
  - score on the final exam
  - grades on labs
  - class participation
    - answering questions, participating in discussions, etc
    - posting to the on-line discussion is a good way to contribute
  - extra credit

- Policy for late work:
  - turn in whatever you have completed by the deadline
  - anything extra submitted after the deadline may be considered for extra credit
    (if we don’t discuss solutions in class)
  - no extra credit unless you submit something by the initial deadline

A Note About Class Participation

- The on-line discussion groups are one way to participate
  - post questions, comments, references to information you found, ...
  - think of the discussion group as a 10-week chat

- Asking questions in class is another way to contribute
  - do the assigned reading before class
  - come ready to ask questions
  - don’t be shy about asking questions in class

- Laptops in class are OK
  - but be considerate of others -- no web browsing or non-class related activities

Ruby

- CIS 170 does not require experience in computer programming
- This course is not an introduction to programming: there will be no “programming assignments”
- There are, however, lab projects using a programming language named Ruby
- Why do we say these are “labs” and not “programming”?
  - programming involves planning and preparation
  - a programming project typically involves design, coding, and testing
  - all but the simplest programming projects require weeks of concentrated effort
- CIS 170 Ruby projects will require between one and three hours
  - we will give you the program
  - you will run it, modify it, see what it does, understand how it works

Ruby (cont’d)

- Ruby is a general-purpose programming language
  - it can be used to write large applications
- In CIS 170 we will use Ruby as a “workbench”
  - load some data into the system
  - type in expressions that process the data, produce some results
  - write statements that implement algorithms, test them on the data
- Using a system like Ruby gives you a chance to experiment with concepts
- You have a choice for running Ruby:
  - download and install it on your own computer -- it’s free
    - you can find instructions on the class web page and Blackboard
  - use a machine in one of the UO computer center labs
    - Klamath B13
    - McKenzie 101
First Week

- Mon Mar 31
  - introduction

- Wed Apr 2
  - representing data in a computer system
  - introduction to Ruby
  - Lab 1: data representation in Ruby
    - due Fri Apr 11

- Fri Apr 4
  - voting: fair elections, history, technology, HAVA, proposals for electronic voting

Getting Started

- Get a copy of the textbook
  - read chapter 0 (Introduction) and sections 1.1 -- 1.6 (Data Storage)
  - Note: it’s OK to use Brookshear 9th edition
    - the (only?) change in the 10th edition is a new chapter on computer graphics

- Figure out how you will run Ruby
  - don’t wait until the first project is due!
  - download and install on your system, or go to one of the labs and make sure you can start and run Ruby (see next slide)

Getting Started (cont’d)

- Log on to Blackboard (http://blackboard.uoregon.edu)
- Make sure you can connect to the CIS 170 pages
- You will be using Blackboard to
  - download lab projects
  - submit completed labs
  - check on your grades
- Download and run Lab 0: “Hello, World”
  - a trivial project to verify that Ruby is installed and working
  - do the lab and submit it (make sure the homework submission process is working)
  - this project is due Apr 4 (this Friday)