Project 1

Preliminary handin by October 17th
Final handin: 10AM, October 24

The Assignment

This assignment will allow you to explore an interesting topic or problem in computer science in more detail. You will write on one (not all!) of the topics outlined below. Included with each topic are some ideas to get you started and some suggested readings. We encourage you to expand the bibliography as you see fit; the preliminary handin meetings will be your opportunity to get input and approval on your topic.

The expectation is that you will make good use of resources to help you form an informed opinion, but that you will go beyond that and produce a paper that reflects both research and independent thought.

Expectations

Your paper must be 3-5 pages, double spaced, not including bibliographic information. You must include a bibliography with at least two sources. Web pages are acceptable sources, provided you can establish that the author has credibility on the subject. Be sure to include inline citations whenever you provide a direct quote or use specific facts, concepts, or ideas presented by another author. Come see a TA if you have questions about how to appropriately reference your sources.

Preliminary Handin

In class next week, you will sign up for a preliminary handin meeting with a member of the course staff. You may bring a draft of your paper to this meeting if you like. At minimum, you must bring a detailed one-page outline of your paper and a copy of your bibliography.
Grading

The preliminary handin will count for 15 percent of your grade. Beyond that, your paper will be graded on content, clarity, and style. Late papers will lose a full letter grade per day late, including weekends. Late preliminary handins will receive no credit.

Topics

Graphical Programming Languages

Is it possible to write a purely graphical language? What advantages might such a language have over traditional programming languages? Disadvantages? In what situations might such a language be useful or appropriate?

Recommended Reading:

- Logical circuit designs can be thought of as a type of program. Look into graphical systems for logical circuit design and simulation. An example is Diglog, which is used in CS31.
- the following gives a survey of visual programming languages: http://www.nickerson.to/visprog/ch2/ch2.htm
- research Visual Basic, which was developed by Microsoft.
- Research some of the visual languages developed by MIT.
- Lego Mindstorms: a product line of programmable robotic legos. Lego Mindstorms is used in CS148: Building Intelligent Robots (which has been taught by Tom in the past). There also exists visual programming software that ships with Mindstorms. To find out more about Mindstorms, check out the resources page of last year’s CS148 website: http://www.cs.brown.edu/courses/cs148/2003/resources.html

Industry Standards

A number of software standards (e.g., those concerning C, HTML, Java, and protocols that make the web possible) have been set since the birth of
Computer Science. Why are standards necessary? Who is (and who should
be) involved in setting industry standards? What are some of the challenges
in setting standards?

Recommended Reading:

- The IEEE (Institute of Electrical and Electronics Engineers) sets many
  hardware and software standards - www.ieee.org

- You may wish to learn how time standards were established (and why)
  and compare with software standards. Find out about time standards
  (and other standards) from the NIST (National Institute of Standards
  and Technology):

  - NIST home: www.nist.gov

- networking protocols: consult the section entitled “Smart Milk Cartons”
  of Chapter 11: “Under the Hood” of “Talking with Computers”

- The ISO (International Organization for Standardization) also sets
  many standards: www.iso.ch

Open Source

What is it? Why is/isn’t it a good idea? Do you think all software will
be open source in the future? Why or why not? What about small-scale
business software? Are certain software markets more conducive to the open
source model than others? If so, which ones and why?

Recommended Reading:

- www.gnu.org

- www.redhat.com/about/

- “The Cathedral and the Bazaar” by Eric S. Raymond

- “The Business and Economics of Linux and Open Source” by Martin
  Fink
Code Libraries

While there are free code libraries available from open source organizations like GNU, there are many, many more libraries of code whose authors usually stipulate that licences be bought by developers who wish to use their libraries. What are the licensing issues at stake?

Recommended Reading:

- try Googling “Java licenses”. Sun has been in the news in recent years regarding licenses for its Java language and other software.

Application-Specific Programming Languages

Does it ever make sense to write a new programming language for a specific application? Such languages are commonly referred to as application-specific, domain-specific, or application-oriented languages. In what cases is it appropriate, and why? What are the risks of such an undertaking? What are the possible benefits?

Recommended Reading:

- Think about special scripting languages (some times called macro languages) for apps like Excel, Photoshop; also consider the languages (extensions of SQL) used to program databases.

- Consider typesetting languages like \TeX and \LaTeX, which is used to produce most of the documents handed out in this class and others in the CS department (including this handout).

- Investigate languages used for designing VLSI circuits.

- Also consider languages aimed at automating parts of Computer Aided Design (CAD), which is used by engineers and architects to design physical objects in the real world.

- How would you categorize mathematical apps like Mathematica and Matlab that have their own scripting languages?

- the history of and motivation behind Java: http://java.sun.com/features/1998/05/birthday.html
Teaching Recursion

Consider the first article mentioned below. Do you think it is appropriate to teach recursion before loops? Why or why not? Why might recursion be considered a difficult concept?

Recommended Reading:

- you may wish to use CiteSeer.com to find papers that cite the above paper: http://citeseer.nj.nec.com/cs

Artificial Intelligence

AI encompasses a tremendously large area of computer science, so large, that many special sub-fields have developed into their own areas (e.g., machine learning, computer vision, genetic programming, etc.). Some examples of AI that you may have seen before include online driving directions generators, spam email filters, computer opponents in video games, and commercial websites that attempt to make (educated) product recommendations to you based on your consumer habits (e.g. Amazon.com), to name a few. Much of what is out there today can be thought of as the automation of a complex process or method of decision-making that is so complex that we call it "intelligent." Choose a particular area of AI and elaborate on the economic, cultural, political, and/or philosophical implications of incarnations of the technology.

Recommended Reading:

- “The Mind’s I,” composed and arranged by Douglas R. Hofstadter and Daniel C. Dennett. “The Mind’s I” consists of many short essays by various experts and academics in CS, Psychology, and Philosophy. Many essays focus on AI efforts to emulate human cognitive abilities. Most of the perspectives are philosophical or psychological.
- consult Chapter 8: “You’ve got (Junk) Email” of “Talking with Computers”
• consult Chapter 15: “Darwin’s Dangerous Algorithm” of “Talking with Computers”