CSE 097S - Penn State University - Spring 2006
Professor John Hannan
Lecture 1: Introductions - January 10&12, 2006

Outline

- Learn about Computer Science
- Learn about Computer Engineering
- Study and Solve some Problems
- Societal/Ethical Issues of Computing
- Anything Goes

Goals of the Course

- Learn a little of how a computer works  Software, Hardware, Communication
- Computer Science and Computer Engineering as Disciplines
- Introduction to Penn State
- Sharpen reasoning abilities

Introductions

- Name (& nickname)
- Something unusual that I’ll remember
- Where you are from
- Intended Major
- Interests

Background Check

- Computer Experience
- Libraries
- Diversions

What’s inside a Computer?

- Physical Parts

- Logical Parts
Computer Science

What is Computer Science?

What isn’t Computer Science?

Computer Engineering

What is Computer Engineering?

What isn’t Computer Engineering?

Logic

What is Logic?

What isn’t Logic?

Hardware -vs- Software

Suppose we assume the following:

Computer = Hardware + Software

Then:
Computer Engineers design the hardware
Computer Scientists design the software
Both use Logic

Diversion I

In a certain flower garden, each flower was either red, yellow, or blue, and all three colors were represented.
A statistician once visited the garden and made the observation that whatever three flowers you picked, at least one of them was bound to be red.
A second statistician visited the garden and made the observation that whatever three flowers you picked, at least one of them was bound to be yellow.
Two logic students heard about this and got into an argument.
The first student said: "It therefore follows that whatever three flowers you pick, at least one is bound to be blue, doesn’t it?"
The second student said: "Of course not!"
Which student was right, and why?
Computers in the Flower Garden
Could a computer have solved this problem?

How could we represent this problem in a way a computer could understand?

Logic plays an essential role in problem solving:

• the kinds of problems computers are good at solving
• the kinds that take (complex) reasoning ability

Computer scientists & engineers need to understand logic and how to use it.

Comp Sci & Comp Eng Majors
You might do the following:

• Design and implement a Database System
• Analyze the complexity of sorting algorithms
• Design arithmetic hardware circuits
• Learn new programming languages
• Implement parts of an Operating System
• Build a microprocessor system

Computer Science Problem
Consider the problem of implementing a search engine.

• We want to maintain information on over 8 Billion web pages
• We want to provide fast, accurate searching of these pages
• How do we store all the information in these web pages?
• How do we search for information in these web pages?

Computer Scientists think about such problems.

Computer Engineering Problem
Consider the problem of multiplying two numbers

• How does a computer do this?
• Special Hardware?
• How fast is it?
• How much power does it use?

Computer Engineers think about such problems.
The Elevator Problem
Assume we have a 100-floor building with one elevator
This is a unique elevator in that once a person gets on they immediately disappear
(or they magically appear in the elevator and request to get off at a certain floor)
As the elevator operator you keep track of all the pending requests (floors to visit)
Your job is to make sure that everyone is satisfied, no one waits too long, and you don’t overwork the elevator
How do you decide which floors to visit and when to visit them
I.e., Give an algorithm that lets you keep your job

Disk Scheduling
The elevator problem is actually a simple way of describing the real problem of Disk Scheduling
Data is read from/written to a computer’s disk much the same as a CD player reads music from a CD
A head moves along a straight line from the outer edge of the disk to its center while the disk is spinning
Typically, the disk spins much faster than the head moves
To read or write data to a certain location, the head must be moved over the proper track on the disk
The tracks are like floors; The head is like the elevator car
Reading data is like people getting on the elevator Writing data is like people getting off the elevator

Latency and Seek Times

The Elevator Problem
We could employ several different strategies
Some are better than others
We can compare them by considering how they behave on the same set of requests
For simplicity, assume we have the following fixed set of requests
5, 35, 2, 14, 12, 21, 3, 9, 22, 20
What’s the best sequence to follow?

The Library Problem
Assume you are in a library seated at a table
Books are shelved in the library stacks
The table has space for only 10 books at a time
If you want a book you have to go to the stacks, get it, bring it back, then place it on the table (Fetch)
If the table is full, you need to return a book before you can get a new one (Replacement)
Which book should be replaced?
We need to decide on a Book Replacement Policy
A good policy should minimize the number of trips to the stacks we need to make

Factors to Consider
• We might later need a book we put away
• If we haven’t used a book recently maybe we don’t need it anymore
• Likewise, if we have used a book recently, we’re likely to use it again
• When we fetch a book, we might also find books nearby (in the stacks) that we should fetch too

Temporal and Spatial Locality
Memory Management

- A computer’s memory can be divided into two parts:
  - Main Memory (RAM)
  - Secondary Memory (Hard Disk)

- Data & Programs reside permanently in Secondary Memory

- When a computer needs to access these, they must be transferred to Main Memory

- If there is no room in Main Memory, then something there must be copied back to Secondary Memory to make room

- Deciding what to replace is a decision made by the Operating System

- We want to minimize access to Secondary Memory

Hot Areas

- What are some hot topics in software?

- What are some hot topics in hardware?

- Other hot topics?

Hot Areas

- The Web
- Bio-Informatics
- Wireless Communication/Networking
- Pervasive/Persistent Computing
- Trusted/Secure Computing
- Power-Aware/Low-Power Computing
- Digital Rights Management
- Embedded Systems
- Multi-modal Interaction