

CS350: Data Structures Class Notes

Class 1: Monday, September 1, 2008

Today we began with the usual organizational stuff—went over the syllabus, talked about the web page and book, etc. In particular,

- You may want to pick up a copy of *Python Pocket Reference* by Mark Lutz, ISBN 0-596-00940-2. If you have an unmarked copy, you can use this during testing. Highlighting and your name are okay.
- Be sure to get the errata for your text.

Please read the syllabus.

Overview of the course

We tried to put the course in context by addressing the following four questions:

1. What is *computer science*? We didn't really attempt to answer this question. Everyone should have an idea and this is really a moving target. If pressed, a common definition is the study of algorithms.
2. What is computer engineering? If pressed, a common definition is applying engineering principles to computer hardware.
3. What is software engineering? If pressed, a common definition is applying engineering principles to computer software.
4. What is data structures? If pressed, a common definition is the study of both the organization of data and of the algorithms used to manipulate data. The separation of data structures and algorithms unclear at best.

Program Life Cycle

Different authors draw different distinctions, but basically we include:

1. Analysis—This includes identifying the relevant data and the operations on that data.
2. Design—This includes organizing the data and identifying algorithms that will be used.
3. Implementation or Coding
4. Testing & Debugging
5. Maintenance

Other possible items include specification, risk analysis, unit coding and unit testing as opposed to system coding and system testing, deployment, etc. For our purposes, we will

roll these steps into the listed steps. At any point the need to cycle back to the first step, analysis may be required. Typically, the bulk of the costs arise during maintenance.

Next time we will begin reviewing background material. Typically, this includes 1. Reviewing the program language being used, 2. Data abstraction and ADTs (abstract data types), 3. The analysis of complexity of algorithms, and, often , 4. Recursive programming.

I'm assuming folks know the basics of Python. We will review key points as they arise. So on Wednesday, we will begin with abstract data types and information hiding. Please read through page 34.