Slide 1

Administrivia

- One purpose of the syllabus is to spell out policies, especially about:
  - Course requirements and grading.
  - Exam dates.
  - Late work.
  - Academic integrity.
- Most other information will be on the Web, either on my home page (here, office hours) or the course Web page (here).

  A request: If you spot something wrong with course material on the Web, please let me know!

Slide 2

More Administrivia

- Part of my job is to answer your questions outside class, so if you need help, please ask! in person or by e-mail or phone.
- Some of my office hours are designated as “open lab”. At those times I will be in one of the labs (probably HAS 340) ready to answer questions.

Slide 3

More Administrivia

- Probably the easiest (though not the only) option for doing the assignments is to use the Linux lab machines.
- You should have physical access (via your TigerCard) to four rooms containing such machines any time the building is open. You should have remote access to any that are booted into Linux.
- Returning students should already have accounts set up. (If you've forgotten your password, go to the ITS help desk and ask for it to be reset.) Accounts are being set up for new students. Username is the same as your Windows/ITS username; password will be mailed to you soon.

Slide 4

What Is This Course About?

- It's an introductory course in programming, with a focus on problem-solving and logic.
- "Programming" — ? solving problems with computers, which requires expressing ideas in a way the computer can understand.

Slide 5

Who Should Take This Course?

- Computer science majors who do not have credit for a similar course.
- Students majoring in subjects that require an understanding of programming.
- Students who want to meet the “Using Scientific Methods” understanding by learning programming.
- Course content tries to meet the needs of all three groups.
- No background in programming is assumed. (But be prepared to spend some time on homeworks. In the words of colleague Dr. Maury Eggen: Programming is not a spectator sport.)

Slide 6

Solving Problems on Computers

- Appearances to the contrary, computers are not smart. What they do well is perform sequences of simple math/logic operations very fast and very accurately.
- What makes them useful is that people have figured out how to break complicated tasks down into sequences of simple operations — i.e., how to “program” them.
- This requires a mindset not quite like that required for any other activity — and can involve a lot of creativity.
- It also involves a form of experimentation (which is why this course meets the CC requirement it does).

Steps in Solving Problems on a Computer

- Understand the problem — what do you want the computer to do, exactly?
- Design a solution suitable for a computer (“develop an algorithm”).
- Implement the solution (“write the program”) and test it. This requires expressing your ideas in “a programming language” — of which there are many! In this course, we will use C under Linux.

Slide 7

Minute Essay

- Tell me about why you are taking this course — as a prospective CS major? for another major (what)? to meet a CC requirement? what is your major?
- Tell me about your background:

  Have you had any exposure to programming? If so, in what language?

  Have you had any exposure to a Linux/Unix command-line interface?
- What are your goals for this course? Anything else you want to tell me?

Slide 8