What is software engineering?

- In groups of two or three people, take three minutes to write down a definition

Key points for definitions

- Full-lifecycle: “womb-to-tomb”
- Multiple people are necessary
  - Many people
  - And people with different skill sets and job descriptions
- Multiple versions of software will be developed
- Economics plays a key role: resources are constrained (cost, time-to-market, etc.)

Highest level goal of 503

- Develop a deep understanding of the fact that software engineering is not a mere matter of programming

What is software engineering research?

- Finding ways to identify and better understand problems that are faced in effectively engineering software
- Finding ways to solve these problems
- Neither the problems nor the solutions are cut-and-dried in software engineering research
- Both are much more contextual than in many other areas of computer science research

People play a key role

- Many aspects of software engineering focus on how to make the humans involved in engineering software more effective (as opposed to the computer itself)
- People use the software systems (even if indirectly) and this places pressures on the software itself
Assessment is complicated

- The contextual and human-oriented nature of software engineering makes it hard to assess proposed improvements
- Some graduate students view this characteristic of software engineering research as sufficiently disturbing to cause them to work in other areas

My view is different

- The problems of software engineering are real: really, really real!
- The “softness” of the problems and the solutions make it more interesting, challenging and exciting
  - In understanding the problems, in determining potential solutions, and in assessing their value
- This does cause the answer to many (perhaps most or even all) questions to include the phrase, “Well, it depends…”

CSE503: Technical focus

- Much work in software engineering touches on managerial issues: this is essential, since coordinating groups of people over time clearly relates to management
- In this course, we’ll focus on technical aspects of software engineering
- That said, it’s impossible to draw this line firmly and clearly
  - Near the end of the course, I may cover a few of the more managerial aspects of software engineering

CSE503: Two primary objectives

- Provide an overview of some of the most important techniques and approaches that can help in producing better software at more predictable costs
  - Understanding state-of-the-art, which may help you in your own system building
- To lay a foundation for performing research in software engineering
  - Even though not all of you actually will!
  - This means that we will often discuss the intention and effectiveness of techniques and approaches, as well as the techniques and approaches themselves

What is your background?

- What’s the largest software system you have ever worked on?
- Original developer or maintainer?
- Any products with significant user bases?
- What were the most difficult software engineering problems you faced?

What do you want from CSE503?

- Other than “it’s a quals course”
“Not a mere matter of programming”: an example

- Proving programs correct has a 30+ year history
  - Given a specification (in a formal logic) and
  - an implementation (in a programming language),
  - prove that the implementation satisfies the specification

- This has often been considered to be the key problem in software engineering

```
x: int;
read(x);
if (mod(x,2) = 1) then
  x := x + 1;
{ even(x) }
```

But it leaves open key software engineering problems

- Requirements engineering
  - Where did the specification come from? Does it satisfy the needs of the customer?
- Design
  - How does it interact with other parts of the program?
- Evolution
  - What happens if the specification is changed?
- Economics
  - What is the cost of proving correctness?
- Testing
  - Should we rely entirely on the proof?

...
Wednesday (1/9):  
a Michael Jackson video  
• I’m sure you’ll all be exhausted from the beginning of the quarter, so I’ve decided to give you a break and show you a video on Wednesday  
• Indeed, a Michael Jackson video  
  – No, not that Michael Jackson!  
• The intent of this video is to drive home a set of ideas, in particular that software engineering is more than a mere matter of programming

Friday (1/11) and Monday (1/14):  
program correctness  
• Basic material on proving programs correct  
  – Program specifications  
  – Semantics of programming language constructs  
  – Pre- and post-conditions  
  – Hoare triples and Dijkstra weakest preconditions  
  – Loop invariants  
  – Proving correctness of data structures  
• Intent  
  – Valuable material on its own  
  – Basis for understanding software specification work  
  – “Not a mere matter of programming”

Then: software specifications  
• Model-based specifications (e.g., Z)  
• Algebraic specifications  
• …