Software Engineering I

SOFT 160

University of Nebraska–Lincoln

August 21, 2017
Learning Goals for Today

Be able to:

- Locate course objectives, roadmap, and resources.
- Explain acceptable types of homework collaboration according to the CSE Academic Integrity Policy.
- Explain to a friend what it means to “engineer software.”
Personnel

- **Instructors**
  - Suzette Person (Avery 362)
  - Brady Garvin (Avery 356)

- **Lab Instructor & Teaching Assistants – Avery 12; Office Hours posted on Piazza (Q&A platform)**
  - Niloofar Mansoor (Lab Instructor)
  - Taher Ahmed (Teaching Assistant)
  - Jasmine Boyer (Teaching Assistant)
  - Hallie Hohbein (Teaching Assistant)
  - Denis Komissarrov (Teaching Assistant)
Software Engineering Program

- Second year of the program — evolving and improving
- "Software engineering first" curriculum model
- Blended model of instruction
Course Format

• MWF class meetings — AVH 347
  • Active and collaborative learning
  • Technical and non-technical skills
  • Software engineering in practice
  • Software engineering in research

• Weekly labs — AVH 347
  • Hands-on activities

• Homework — expect to spend 8-12 hours per week on homework
Expectations

- Attendance is required at all class meetings and lab sessions
- Come to class on-time and prepared
  - Bring your laptop (battery charged)
  - Be ready to answer questions about assigned readings, etc.
- Maintain a supportive and constructive environment at all times
- Complete lab assignments within two weeks (if not completed during the lab session)
Grading

- Class participation: 5%
  - Initial credit: 4%
  - Providing mid-semester feedback: 1%
  - Each unexcused absence: -0.5%
- In-class activities and assignments: 40%
  - Lab assignments: 25%
  - Instructor-led in-class activities: 5%
  - Team-worked parts of capstone: 10%
- Homework assignments: 25%
  - Journal assignments: 5%
  - Non-capstone homework: 15%
  - Individual parts of capstone: 5%
- Quizzes: 10%
  - Ten short quizzes in lab: 1% each
- Exams: 20%
  - Two non-comprehensive: 5% each
  - Comprehensive final exam: 10%
Extra credit: Each instructor-endorsed question or response on Piazza is worth 0.25% up to a max of 2%.
Collaboration is permitted as specified in assignments. If not explicitly permitted, then you may not develop joint solutions, share work or copy anything (from anyone or anywhere).

- Note that we will use the Stanford Measure of Software Similarity (MOSS) tool to check for plagiarism in homework, etc.

- Communication: Piazza

- Technology: no phones, no web browsing, posting to social media, etc. during class or lab sessions
<table>
<thead>
<tr>
<th>Exam</th>
<th>Class Average</th>
<th>Laptop-User Average</th>
<th>Average Improvement after Quitting</th>
</tr>
</thead>
<tbody>
<tr>
<td>#1</td>
<td>83</td>
<td>71</td>
<td>—</td>
</tr>
<tr>
<td>#2</td>
<td>85</td>
<td>72</td>
<td>+16</td>
</tr>
<tr>
<td>#3</td>
<td>82</td>
<td>69</td>
<td>+11</td>
</tr>
</tbody>
</table>

—*Put Your Phone Down Please: Digital Devices and Student Performance.* D. Duncan. UC Boulder.
Digital Distractions

- Average difference in score for students who multitask: -11%
- Average difference in score for students with a multitasker in view: -17%

—Laptop Multitasking Hinders Classroom Learning for Both Users and Nearby Peers. F. Sana, T. Weston, and N. Cepeda. McMaster University and York University
Course Objectives

1. Write and modify code to organize data and solve computational problems.
2. Apply defensive programming techniques to improve maintainability and minimize faults.
3. Explain what it means to engineer a software solution.
4. Analyze computational artifacts.
Course Objectives

5. Apply iterative software development practices to incrementally improve solutions.

6. Use software engineering tools to support the problem solving process.

7. Solve problems using computational resources in a team environment.

8. Describe similarities and differences in techniques learned in class versus state-of-the-art techniques and state-of-the-practice techniques.
Roadmap

- Module 1: Getting Started as a Software Engineer (2 weeks)
- Module 2: Assessing Software Quality and Teamwork (3 weeks)
- Module 3: Constructs for Solving Computational Problems I (3 weeks)
- Module 4: Testing and Debugging (1 week)
- Module 5: Constructs for Solving Computational Problems II (2 weeks)
- Module 6: Course Capstone (4 weeks)
Advice

“Stay on top of journals and coding assignments, don’t wait until the end to do them. Also, if you can practice coding outside of the class assignments, it will help a lot.”

“Do not be afraid to speak up! Everyone in the course will most likely feel the same way you do.”

“Be passionate in programming, even if it seems nothing but a blur to you, have the motivation to strive through!”

”The first piece of advice...look at the labs before the class and to maybe even start working on them. The second piece of advice...create a dictionary of all the terms you learn at the start of the course as the terms come up constantly.”
Advice

• “Don’t procrastinate your projects...If you start the homework early, it gives you more time to potentially go in and see a professor or TA if you cannot figure out an unanticipated problem on your own.”

• “Don’t be afraid to ask for help and get comfortable doing so really fast. Chances are that you will need to get help on certain things throughout the years and the sooner you get comfortable asking, the more successful you will be.”

• “…always take the time to think through a project before you begin working on it. Planning is vital to the success of the project as it prevents having to start over and redesign things when they end up not working out.”