Learning objective
To help computer science students be better prepared to be an immediate contributor to a software project

Grading
The objective in the grading structure is to match what will be expected of you in the workplace.
50% group programming projects
   Grading will be based on the following criteria
   - Features meet customer requirements
   - Project docs: Stories, Estimates, Commit Messages, Burn down charts
   - Code cleanliness metrics (largely based on Code Complete by McConnell)
25% quizzes on reading and in-class discussions
   Grading based on understanding the material and being able to articulate it in writing. Each quiz will receive a grade of + (understood), √ (partial understanding), or 0 (absent for quiz or unprepared for material). Quizzes will happen at the beginning of class, so being on time will be important. There will be no make-up for absences not communicated before the start of class.
20% pick 2 mini-assignments
   Grading will be based on performing the assigned task and writing skills.
5% Ignite-style class presentation
   Each presentation will receive a grade of + (researched, prepared, practiced, engaging), √ (researched, prepared), or 0 (unprepared).

Texts
The Art of Agile Development by Shore & Warden
The Five Dysfunctions of a Team by Lencioni
Peopleware:Productive Projects and Teams by DeMarco & Lister

Topics discussed in class

<table>
<thead>
<tr>
<th>Class Dates</th>
<th>Tuesday</th>
<th>Thursday</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aug 26, 28</td>
<td>Expectations for class, Introduction of Jon and Michelle, Introduction of Agile</td>
<td>Getting to know other people in class</td>
<td>Software teams chosen</td>
</tr>
<tr>
<td>Sept 2, 4</td>
<td>Iteration Zero</td>
<td>Creating User Stories (IC)</td>
<td>Completion of 5 Dysfunctions</td>
</tr>
<tr>
<td>Sept 9, 11</td>
<td>Estimating</td>
<td>Estimating (IC)</td>
<td>20 pages of Peopleware and 20 pages of Art of Agile expected every week from here on out</td>
</tr>
<tr>
<td>Date Range</td>
<td>Topic</td>
<td>Subtopic</td>
<td>Notes</td>
</tr>
<tr>
<td>-------------</td>
<td>--------------------------------------------</td>
<td>----------------------------------------</td>
<td>--------------------------------------------</td>
</tr>
<tr>
<td>Sept 16, 18</td>
<td>Test Driven Development</td>
<td>Design</td>
<td>Ignite presentations start, 2 per class period Programming on projects starts</td>
</tr>
<tr>
<td>Sept 30, Oct 2</td>
<td>Databases</td>
<td>Crucial Conversations (IC)</td>
<td></td>
</tr>
<tr>
<td>Oct 7, 9</td>
<td>QA your work - minimize bugs, finding the source</td>
<td>Team Dynamics Player vs. Victim Working with non-devs</td>
<td>Iteration 1 ends, Fall Break on the 10th</td>
</tr>
<tr>
<td>Oct 14, 16</td>
<td>Code reviews (IC)</td>
<td>Retrospectives (IC)</td>
<td></td>
</tr>
<tr>
<td>Oct 21, 23</td>
<td>Refactoring</td>
<td>Working with users to pick priorities, defer features Innovation Games</td>
<td>First mini-assignments due</td>
</tr>
<tr>
<td>Oct 28, 30</td>
<td>Integration Testing</td>
<td>Build lights, storyboards, other physical space Working with JIRA &amp; QA</td>
<td></td>
</tr>
<tr>
<td>Nov 4, 6</td>
<td>Errors, Exceptions, Logging</td>
<td>Running an effective meeting (IC)</td>
<td>Iteration 2 ends</td>
</tr>
<tr>
<td>Nov 11, 14</td>
<td>Security</td>
<td>Communicating Tech Debt and inaccurate estimates to business</td>
<td></td>
</tr>
<tr>
<td>Nov 18, 20</td>
<td>How to deal with Tech Debt</td>
<td>Gathering and negotiating user requirements (IC) User observation</td>
<td></td>
</tr>
<tr>
<td>Thanksgiving Break</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Dec 2, 4</td>
<td>Code smells</td>
<td>Usability</td>
<td>Second mini-assignment due</td>
</tr>
<tr>
<td>Dec 9, 11</td>
<td>DevOps Support</td>
<td>Looking for a job Interviewing (IC)</td>
<td>Iteration 3 ends</td>
</tr>
<tr>
<td>FINALS Week</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Mini-assignments** - should be 600 - 800 words
- Ask someone how they accomplish a task, perform an ethnographic observation, and make a report on differences
- Interview 3 managers - ask them what makes a good or bad developer, write a report on what surprised you
- Pick a company you would like to work for. Research their development practices. What overlaps with what we’ve talked about in class?
- Interview 4 hiring managers or recruiters - ask them their favorite interview questions and what they look for in a response. How did this differ with your expectations? How will this change your behavior?
- Interview 2 people with Death March experiences - compare and contrast their perceived reasons for the death march and how they think one could be avoided in the future

**Ignite presentations** - 20 slides, 15 second auto-advance on the slide, 5 minutes total
1. Design patterns
2. Character sets / Unicode
3. Mocks
4. Scrum
5. Kanban
6. Paper prototyping
7. Gantt charts
8. Time boxing
9. Mythical Man Month
10. Lean as it relates to software
11. Continuous Deployment
12. Usability testing
13. 508 Accessibility
14. Domain driven design
15. Debugging heuristics
16. Data Visualization
17. Time zones, months & Easter
18. Body language of different cultures
19. Negotiating in different cultures
20. Typeface
21. Non-functional requirements
22. Automated Build Tools
23. Load testing
24. How mobile is changing design paradigms
25. Intellectual property
26. Payment gateways
27. Big data
28. Viruses
29. Encryption
30. HTML5
31. Responsive design
32. Entrepreneurship
33. Emotional Intelligence
34. Big-O-Notation
35. ITIL
36. StackOverflow.com
37. Amazon Cloud, Azure, Google
38. Platform as a Service
39. Infrastructure as a Service
40. Breach planning
41. GitHub
42. Technology Podcasts
43. C# vs. Java
44. NoSQL databases
45. Stock options and dilution
46. OAuth
47. How to find bugs
48. Good passwords
49. Cucumber
50. FitNesse
51. How Facebook scales
52. Outsourcing
53. Practices for distributed teams
54. Patterns of Enterprise Integration
55. Database refactoring
56. Publishing your mobile app