Software Engineering – Writing Intensive

COMP 3100

Who am I?

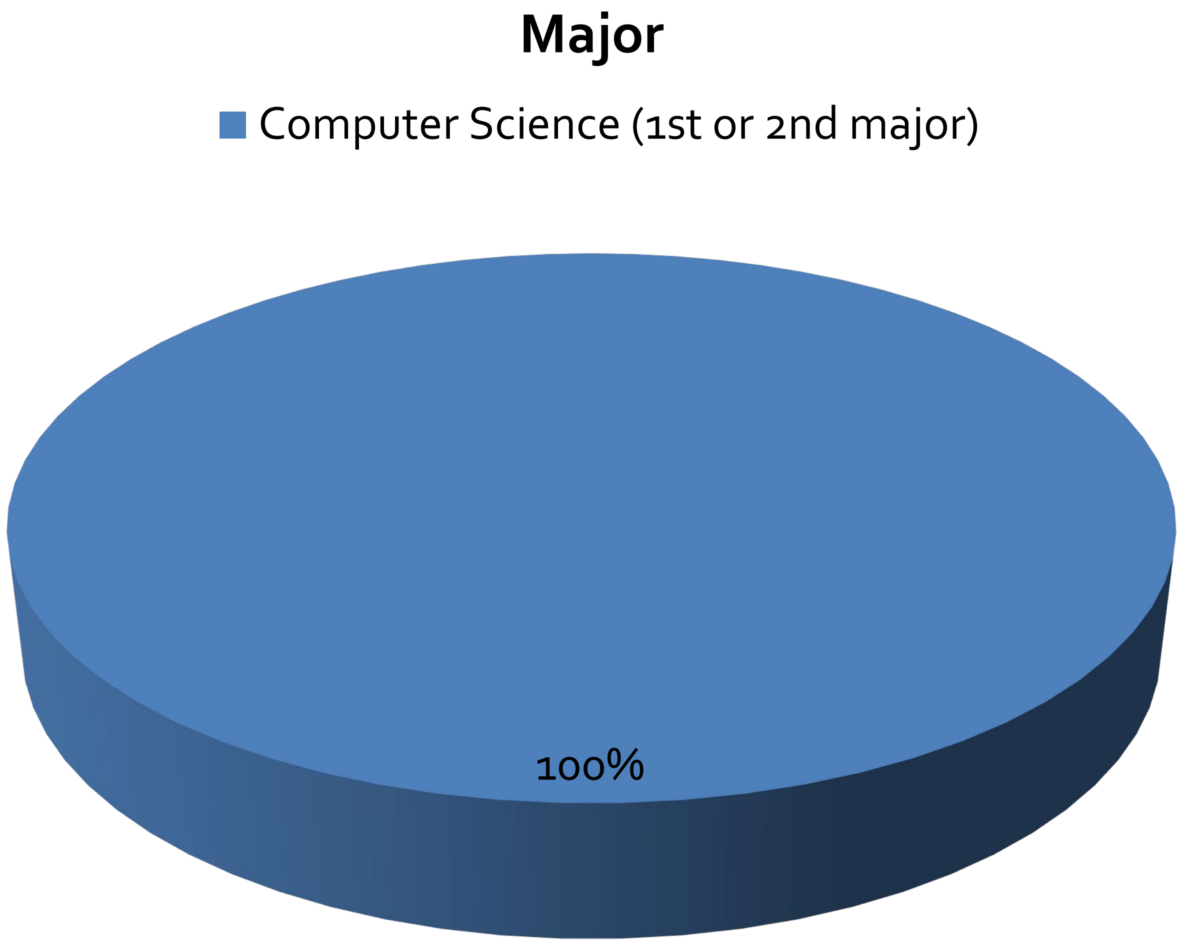
- Dr. Barry Wittman
- Not Dr. Barry Whitman
- Education:
  - PhD and MS in Computer Science, Purdue University
  - BS in Computer Science, Morehouse College
- Hobbies:
  - Reading, writing
  - Enjoying ethnic cuisine
  - DJing
  - Lockpicking
  - Stand-up comedy

How can you reach me?

E-mail:	wittman1@otterbein.edu
Office:	C123 (Art & Communication Building)
Phone:	(614) 823-2944
Office hours:	**MWF** 9:00 – 10:15 a.m.
	MWF 1:45 – 2:45 p.m. (in C142)
	W 4:00 – 5:00 p.m.,
	TR 10:00 – 11:30 a.m.,
	TR 2:00 – 4:00 p.m.,
	and by appointment

- **Website:**
  http://faculty.otterbein.edu/wittman1/

Who are you?



Why are we here?

- What's the purpose of this class?
- What do you want to get out of it?
- Do you want to be here?

Course Overview

Textbook

- David Bernstein and Christopher Fox
- ***Introduction to Software Engineering***
- Preview edition provided free to students on Brightspace
- Get it now!
- Please do not post it on the Internet

You have to read the book

- You are expected to read the material before class
- If you're not prepared, you may be asked to leave
    - You will forfeit the opportunity to take quizzes
    - Much more importantly, you will forfeit the education you have paid so much money to get

Course focuses

- Java expertise
- Preparation for Computer Science Practicum
- Requirements analysis
- Agile software development
- Software design
- Testing and quality assurance
- Project planning and management
- Version control
- Working as a team

Software Engineering

In previous classes…

- You probably only cared if your code **worked**
- Software engineering has a different focus
  - Is your code readable by others?
  - Is your code easy to update and expand?
  - How can a large number of people collaborate on one code base?
  - How do you know if your code works?

Java

- The code you do in this class will be Java
- Why?
  - It runs on most platforms
  - It's good for large-scale applications
  - I know it really well
  - It works with a solid unit-testing framework called JUnit
- If you need some Java refreshers, check out my book: https://attacking-problems.github.io/

More information

- For more information, visit the webpage:
  `http://faculty.otterbein.edu/wittman1/comp3100`
- The webpage will contain:
  - The most current schedule
  - Notes available for download
  - Reminders about projects and exams
  - Syllabus
  - Detailed policies and guidelines

Projects

One giant project

- 55% of your grade is one giant project
- You will work on teams of four to five students
- Each team gets to pick its project
  - Some kind of (board) game is a good idea
  - Lots of areas for additional features
  - Lots of functionality to test

Phases

Phase	Description	Weight	Due
Software Requirements Specification (Draft)	What is your program supposed to do? Customers often supply a specification, but you'll have to do it yourselves.	**5%**	09/13/2024
Software Requirements Specification (Final)		**5%**	09/20/2024
Design Document (Draft)	How will your program do what it's supposed to do? Include a prototype of some features.	**5%**	10/11/2024
Design Document (Final)		**10%**	10/21/2024
Basic Functionality and Unit Tests	A baseline of functionality with a suite of unit tests.	**15%**	11/08/2024
Final Program and Manual	Final, polished program, fully documented, with manual and system tests.	**15%**	12/06/2024

Turning in projects

- All projects will be committed to private repositories on GitHub (https://github.com/) before the deadline
- Do **not** put projects in your public directories
- Late projects will not be accepted

Written Reflections

Written reflections

- 5% of your grade will be four written assignments
- After each phase of the project, you will be required to submit a written reflection on the process and how your team is doing
- Before the text of each assignment, you will fill out a table rating each member of your team (including yourself) in a number of areas
  - This information will be used as part of a formula to weight the grade that each student gets on each project

Quizzes

Pop Quizzes

- 5% of your grade will be pop quizzes
- These quizzes will be based on material covered in the previous one or two lectures
- They will be graded leniently
- They are useful for these reasons:
    1. Informing me of your understanding
    2. Feedback to you about your understanding
    3. Easy points for you
    4. Attendance

Exams

Exams

- There will be two equally weighted in-class exams totaling 20% of your final grade

Exam 1:	09/23/2024
Exam 2:	10/30/2024

- The final exam will be worth another 10% of your grade

Final:	2:45 – 4:45 p.m. 12/13/2024

Course Schedule

Tentative schedule

Week	Starting	Topics	Chapters	Notes
1	08/26/24	Introduction and Git	1, notes	
2	09/02/24	Software Requirements	5	Labor Day
3	09/09/24	Software Processes	2	Draft Requirements Due
4	09/16/24	Scrum	3	Final Requirements Due
5	09/23/24	Software Quality Assurance	4	Exam 1
6	09/30/24	User Interaction Design and Software Engineering Design	6, 7	
7	10/07/24	Construction Techniques	8	Draft Design Document Due
8	10/14/24	Quality Assurance in Construction	9	October Break
9	10/21/24	System Testing	10	Final Design Document Due Monday
10	10/28/24	Deployment	11	Exam 2
11	11/04/24	Task Identification and Effort Estimation	12	Baseline Functionality and Tests Due
12	11/11/24	Financial and Economic Planning	13	
13	11/18/24	Scheduling	14	
14	11/25/24	Execution and Control	15	Thanksgiving
15	12/02/24	Review	All	Final Project and Manual Due

Policies

Grading breakdown

55%	• Project
5%	• Attendance on work days
5%	• Written reflections
5%	• Quizzes
20%	• Two equally weighted midterm exams
10%	• Final exam

Grading scale

A	93-100	B-	80-82	D+	67-69
A-	90-92	C+	77-79	D	60-66
B+	87-89	C	73-76	F	0-59
B	83-86	C-	70-72		

Attendance

- You are expected to attend class
- You are expected to have read the material we are going to cover **before** class
- Missed quizzes cannot be made up
- Exams must be made up **before** the scheduled time, for excused absences
- Attendance will be recorded and graded for work days (which are all Fridays except for Thanksgiving week)

R-E-S-P-E-C-T

- I hate having a slide like this
- I ask for respect for your classmates and for me
- You are smart enough to figure out what that means
- A few specific points:
  - Silence communication devices
  - **Don't use the computers in class unless specifically told to**
  - No food or drink in the lab

Computer usage

- We will be doing a lot of work on the computers together
- However, students are always tempted to surf the Internet, etc.
- Research shows that it is nearly impossible to do two things at the same time (e.g. use Reddit and listen to a lecture)
- For your own good, I will enforce this by taking **1% of your final grade** every time I catch you using your computer for anything other than course exercises

Generative AI

- You may **not** use generative AI tools like ChatGPT, Claude, Copilot, or Gemini to design any software or write any documents or reflections in this course
- But, because this course is a bridge to the real world, such tools may be consulted as a resource when trying to finish a difficult section of code
  - However, you **must** note that you did so in the comments for that code

Academic dishonesty

- Don't cheat
- **First offense:**
  - I will try to give you a zero for the assignment, then try to lower your final letter grade for the course by one full grade
- **Second offense:**
  - I will try to fail you for the course and try to kick you out of Otterbein
- Refer to the syllabus for the school's policy
- Ask me if you have questions or concerns
- **You are not allowed to look at another student's code, except for group members in group projects (and after the project is turned in)**
- **Don't use AI tools like ChatGPT except as described in the previous slide**

Disability Services

- The University has a continuing commitment to providing access and reasonable accommodations for students with disabilities, including mental health diagnoses and chronic or temporary medical conditions. Students who may need accommodations or would like referrals to explore a potential diagnosis are urged to contact Disability Services (DS) as soon as possible. DS will facilitate accommodations and assist the instructor in minimizing barriers to provide an accessible educational experience. Please contact DS at DisabilityServices@otterbein.edu. More info can also be found here. Your instructor is happy to discuss accommodations privately with you as well.

Software Engineering

Terms as used by the book

- **Software**:
  - Instructions executed by a processor **or**
  - Human-readable statements in a programming language
- **Program**: General term for a piece of software that can run on its own
- **Library**: Group of related sub-programs for accomplishing a specific collection of tasks

Software product

- A **software product** is one or more programs, sub-programs, or libraries and the data and supporting materials and services that allows a client to solve problems
- **Bespoke software products** are developed for a specific customer
  - Code that makes the brakes work, written for a particular car company
  - A website developed for a particular customer
- **Generic software products** are developed for a general market
  - Microsoft Office, for example

Software engineering

- **Engineering** is applying theories and tools to the specification, design, creation, verification and validation, deployment, operation, and maintenance of products
  - That's what the rest of the department does
- **Software engineering** is applying engineering discipline to building and maintaining software products
- Systematic and disciplined:
  - Development
  - Operation
  - Maintenance
- High quality software products delivered on time at minimum cost

Managerial software engineering concerns

- Managerial concerns are about organization and control
  - Project cost
  - Time estimation
  - Scheduling and tracking
  - Team management
  - Risk management
  - Quality

Technical software engineering concerns

- Technical concerns are about what product, how to build it, and building it
  - Software requirements
  - Design
  - Programming languages and environments
  - Coding standards
  - Defect prevention, detection, and removal
  - Version control
  - Documentation
  - Maintenance

All this is really hard

- You've hardly had to think about these things before except in the vaguest way
- The programs you've written have been small
- The specifications have been provided by your instructor
- You haven't had to worry about interacting with more than one other teammate

Questions when building software products

- What exactly should it do? What if people disagree?
- How does this product fit into the rest of the stuff the company does?
- How will users interact with the product?
- What parts should the product have?
- What languages should it be written in?
- What standards should we use to write it?
- How do we know if the program does what it's supposed to?
- How much time and money will it take to make it?
- What kind of documentation will it need?
- How will it change in the future?
- How far along are we in the process of making it?

More problems for large products

- The requirements themselves are huge
- The designs are large and complicated
- The code is long
- Testing gets harder because there's more to go wrong
- More people are on the project
  - Tracking progress gets harder
  - Communication gets harder
  - More managers are needed

Upcoming

Next time…

- Finish introduction to software engineering
- Using git and GitHub

Reminders

- Read Chapter 1: Introduction
- Learn about git:
  - https://www.youtube.com/watch?v=USjZcfj8yxE
  - https://www.youtube.com/watch?v=HVsySz-h9r4
- Form your teams!
- Think about what kind of project you want to work on
- Fridays will usually be work days