# CS 4488 Spring 25 Syllabus

Analysis, specification, design, implementation, and testing of a large software project. Formal approach and tools. Software lifecycle. Human computer interaction. Project and team management. Uses a different programming language. PREREQ including transitivity: (>=C-); CS3310 Databases, CS1337, CS2263, CS3305, CS 4412, Math 1187, 2240, and Math 1175. Pre or COREQ (>=C-): CS 3337.

**Initial Individual Assignments and Lectures (First 2 Weeks)**

1. Mon 13Jan25 8:00 in class lecture on course organization and possible team projects
2. Wed 15Jan25 8:00 in class lecture on Brooks, running coding teams, etc
3. **Wed 15Jan25** Your rank ordering of team projects due
4. **Thurs 16Jan25** Summary of “The Six Second CV Test” Due
5. Fri 17Jan25 8:00 in class lecture
6. Weekend: Team members and projects announcement (e.g., Sorting Hat).
7. **Wed 22Jan25** Team Charters due. See Syllabus below for details
8. **Wed22Jan25** Individual CVs due

See CS 4488 S25 Moodle page for all individual and team assignments

Meeting times: **8:00-8:50 MWF. Poky 304 BA. IF: 282 Tingey**. Textbook: Brook’s The Mythical Man-Month anniversary ed ISBN-13 978-0-201-83595-3.

**Instructor:** David V. Beard, PhD, Professor of Computer Science. 208.282.2684 Colonial Hall 232 [beard@isu.edu](mailto:beard@isu.edu) David Beard’s office hours: 11:00-11:30 TH. I am generally available any time, so send me some email if you want to meet at another time. If I cannot address your issues over email, I will setup a meeting.

The objective of this course is to give computer science students an opportunity to practice the skills acquired in their other CS major courses in the relative safety of the classroom before being dropped into a professional coding team. It is expected that students will be taking this course during their last semester. Elements will be drawn from the entire experience as a CS major. The course is experiential in nature, i.e., the student will be required to produce results for use by real individuals and will be evaluated individually and collectively, both on the process and the product. The course is based on the intention learning process; it is not a textbook or case course. The professor is a resource, not a lecturer. This course assumes significant knowledge of standard CS programming languages such as C++ and C#. You may be expected to extend existing software to add additional features or to learn a new programming language on your own. (Note the 4488 Moodle page has “bootcamps” for rapidly learning C++ and C# including links to solid video tutorials. There are also solid tutorials available from stackoverflow, Cplusplus.com, msdn.com, etc. C++ code must run within the visual studio community 2022 projects and Dev C++/11 with the g++ compiler. C# code must run as a windows form within visual studio community 2022.

Note: If the professor cannot run your code, it does not work.

## CS4488 Team Organization and Deliverables

You are an essential element in a team of professionals who must management your project to create a tool that solves your clients’ problem. (“If the user can’s use it, it doesn’t work. Grade==0”) I suggest you select a leader and get started. 4488 teams use Brook’s labor division of a team ‘movie director’ (i.e., ’technical lead’, ‘surgeon’, ‘architect/chef implementer’, ‘chief programmer’ etc.) and a team ‘movie producer’ (’scrum master’ (sort of), ‘project manager’, ‘project facilitator’, ‘team bean counter’, etc.). The director and producer must be different people who throughout the semester each submit periodic reports (detailed below) to the professor through moodle. Do not expect your professor to tell you what to do. You are responsible for maintaining a schedule and delivering a viable product. (Note that depending on various issues, some students may be assigned to an individual project, in which case they will be required to submit both the director and producer reports.)

Your team needs to have your team director submit your written team charter (described below), detailing how your team is organized and the rules under which you will operate: Who does what? Attendance rules? Who takes attendance at meetings? Task List? Etc.? How will you make decisions? Who is in charge? When is your first meeting with the client? How often will you be meeting with the client? Note that either the director or producer may be ultimately in charge, but someone must be, and that person must be identified in your team charter.

**You are empowered to manage and fire you team members and even team leaders**. This action should not be taken lightly. It does not fail the student in the class, but rather returns the student to the class environment. If you exercise this option, you should follow your published organizational procedures detailed in your charter. The student may protest the firing, with the professor having the final say.

Teams and students are expected to take full responsibility for their learning process. The class meets 8:00-8:50 MWF. Most of the class time will be devoted to group activities. The majority of these will not involve the professor. Attendance is mandatory at all class periods and team meetings. Teams will meet each MWF 8:00 in the course classroom or another room provided by the professor. The professor should be copied in on all email communication with the client. The professor will periodically meet with the team.

To complete the projects, students will be expected to participate in group/class activities and individual coding and other work outside the scheduled period. College class are supposed to ***average*** 9 hours outside the classroom of homework for a 3-hour class. The professor and your team members are expecting massive amounts of coding and other work outside the team meetings 8:00-9:00 MWF. 4488 students not putting in sufficient time can expect a grade of F.

Early in the course, the professor may lecture on basic subjects needed to get the teams moving. In addition, he may prepare special lectures on issues the team considers important. The student is expected to take full responsibility for the learning process. The course is intended to allow the student to integrate the contents of other courses and focus on the process of organizing and running a large-scale project. This will require a commitment of time that the student should not under-estimate. The students are responsible for developing the detailed schedule of activities.

It is the responsibility of each team to establish a **written organization structure document** to carry out the assigned project using professional standards. For each project, the team is responsible for the following minima:

1. Plan for your overall enterprise
2. Analysis of the user, the users’ tasks, the external design, and the internal design.
3. Feasibility analysis and proposal for each task
4. Effort estimates – including task decomposition and estimated durations
5. Project schedule in advance of project implementation
6. Hardware and software recommendations (if appropriate).
7. Progress reports.
8. Final reports including documentation, operating instructions, software, training materials, software repair procedures, procedure documents.
9. Attendance of each team member to teach meeting during the reporting period including arrival time if >8:00am. (On time is late.)

The purpose of the projects is to allow the students to demonstrate their ability to use what they have learned. Although students have worked in groups, they generally have not had to manage a team and the many elements of the product life cycle.

**External Code Libraries**

Do not use any external code library that is not part of the base language (e.g. #include <iostream> is ok in C++) without instructor permission. Your goal is to learn to write the code in such a library, not simply call black boxes someone else has written. Computer Scientists need to be able to write the black boxes others use.

**Individual and Team Grading**

Every 3 weeks or so (as indicated on the moodle page) each team **producer**, working with the team, will submit a short producer report listing the following in 3 separate sections:

1. Producer Report Section 1: Tasks the team was planning on accomplishing (including which **team member(s) responsible**, which team members worked on the task, **estimated number of hours,** and **start date**).
   1. The producer report section 1 of planned tasks from Producer Report N should match the last of planned tasks from section 3 of Producer Report (N-1).
2. Producer Report Section 2: What tasks were actually finished (not worked on but “stake through the heart”) **and by whom**, with **estimated staff hours, actual staff hours, start date, and completion date**. (For each task I will need a rank ordered list of those team members who worked on that task)
   1. The section on what tasks were accomplished needs to include a rank ordering of the relative effort of teach team member during that producer report time period.
   2. What team meetings (dates and times) occurred during the producer report period (time and duration) and who did and did not attend.
3. Producer Report Section 3: What tasks the team is planning on accomplishing next including by whom and estimated staff hours and most likely start dates.

If a planned task (section 1) is not completed, then in section 2, the producer needs to subdivide that task into completed and non-completed subtasks along with actual staff hours. Thus, **do not state that task #3 “was “worked on”** but rather “task #3.2.4 was completed, and task 3.2.5 was not completed.” You also need to indicate which tasks are dependent on the completion of what other tasks; a PERT/GANT chart might be useful here. Team members must be allowed a reasonable amount of time to review the report and suggest corrections before the report is submitted to the professor.

Every 3 weeks or so, each team **director** needs to upload a zip file to the moodle page containing a complete snapshot copy of all the code and other documents for the project. I need a zipped folder with all the software etc, not just a link to some software product somewhere in the Cloud. **Your director-uploaded-code snapshot needs to compile and run on the professor’s computer using the professor’s installed software**. (Dev C++ and visual studio 2022 C++ or C# windows forms apps.) If the professor cannot run the code, it does not work, so it is the team’s responsibility to ensure the code runs on the professor’s computer.

Each project deliverable (i.e., the bi-weekly uploads and final deliverable) including internal and external documentation, project management documents (e.g., PERT chart), source code (down to the class and module block comment), testing documents, relational database source and documentation, etc. must include attribution to the team individual or individuals responsible for its generation.

**Individual Assignments**

As indicated on the Moodle page, there are a number of period individual writing assignments including your resume. These represent a significant portion of your overall grade and poorly done or missing writing assignments can significantly reduce your overall course grade.

**Grade Breakdown**

1. Overall Group project grade: 40%
2. Individual portion of the overall group grade 40%

Determined by the professor using a variety of inputs including evaluation of the team code, producer reports, team meetings, other documents, etc.

1. Individual writing assignments as specified in Moodle: 20%

The grade on the group project is a subjective judgement on the part of the professor based on the relative project difficulty, team effort, and team results compared to other 4488 team projects over the years. The professor will take into account documented team time and effort exploring ideas and directions that did not work as long as the “post-mortem” of these efforts is detailed in the various bi-weekly producer reports.

**Team Coding Projects and Cheating**

Teams are expected to only turn in code they have written themselves. Someone other than a team member writing part or all of a submitted solution, or copying of any portion of submitted work, regardless of the number of editor changes, renaming, and/or retyping, is considered cheating.  If you want to use some software package, you will need explicate and written (email\_) permission of the instructor.

Each submitted class and method needs a block comment that contains the name of the individual team member who has primary responsibility for that item and certifies that he or she wrote the code himself or herself. If the team submits an item that has been copied in whole or in part from another source or written by a non-team member, that team member will be considered to have cheated. If the team member who has primary responsibility for an item cannot to describe how that code works, it will be considered clear evidence of cheating. If the team submits an item that is not attributed to an individual team member and has been copied in whole or in part from another source or written by a non-team member, then the entire team will be considered to have cheated. It is the responsibility of the entire team to conduct “due diligence” to assure themselves that the team has not submitted any code copied in whole or in part off the internet or written by a non-team member, regardless of the number of editor change. Referencing the location from which you have copied code off the internet is still cheating. Cheating will result in failing the course.  Note this standard is similar to what would be expected at a software company to make sure the company is not using code or tries to copyright code that has been written by someone else and not owned by the company.  ISU Policy 4000, Academic Integrity and Dishonesty Policy, can be found at  <http://www2.isu.edu/policy/4000/index.shtml>.

**Some other thoughts and questions:**

1. If you do all the work yourself, you cannot pass
2. If you free-ride you cannot pass
3. Team work is the path to success – if one end of the boat sinks, the entire boat sinks.
4. Since this course is participatory in nature, no provision is made for missed work.
5. All work is due no later than the beginning of the final class period (Friday dead week).
6. Are there ethical considerations you have missed while doing the project?
7. How would one calculate the payback on a project?
8. Explain how to calculate the break-even point of a project.
9. Explain the relationship between top-down design and bottom up testing.
10. Could your project be marketed overseas?
11. Explain structured walk-throughs in detail including participants and outcomes.
12. What is the time relationship between programming, documentation, design, and testing?
13. What are the components of a system proposal?
14. What is the work breakdown structure?
15. What is the difference between a requirement and a specification?
16. What is the relationship between the systems analyst and the systems tester?
17. Could your organization be viable in the ‘real world”?