CPSC 315 Programming Studio: Fall 2012

Syllabus

NEWS: *8/27/12, 08:47AM (Mon)*

- [08/27/12] Programming proficieny survey (in class): this will help determine the teams
- [08/26/12] Course web page goes online.
- ----------
- [LINKS] •News archive•Grades•Codes•Lecture notes

Read-Only Bulletin Board.: 8/26/12, 09:22PM (Sun)

Page last modified: 8/27/12, 09:21AM Monday.

General Information Resources Weekly Schedule Credits Lecture Notes Example Code Read-Only Board

I. General Information

Instructor:

Dr. Yoonsuck Choe
Email: choe(a)tamu.edu
Office: HRBB 322B
Phone: 979-845-5466
Office hours: M 10:30am-11:30am, W/F 4-5pm

TA and Peer Teacher:

Chris Pu (TA)
Email: shipu(a)cse.tamu.edu
Office: HRBB 503
Office hours: TBA

Amanda Cofsky
Email: ancofsky(a)gmail.com
Office: HRBB 219
Office hours: F9-11am

Nick Melnyk
Email: melnynic1(a)tamu.edu
Office: RDMB 111C
Office hours: TR1-2pm, F11:30am-12:30pm.

Prerequisite/Restrictions:

This class is intended for students who have completed CPSC 314 - Programming Languages, and

are concurrently taking CPSC 313 - Intro to Computer Systems. It is meant to be somewhat of a "capstone" course for the lower-level computer science courses, before taking courses in the upper-level tracks.

Lectures:

MWF 1:50pm–2:40pm, ARCC 105

The course is listed as a 2-hour per week lecture, and 2-hour per week lab, however it has been intentionally scheduled for 3 hours per week of lecture (along with the lab). We will meet a minimum of 28 lecture periods over the course of the semester. The idea is to "front-load" these lectures in the earlier part of the semester, to cover material that might be useful when working on the programming projects, and spend less lecture time during the project periods themselves. Also, some days when the instructor travels might be used as some of the "missed" days. The specific list of days we will meet will be provided on the course web page.

There is a final exam time reserved for this class. Although the plan is to wrap up the course before this time, students should leave the final exam time available until instructed otherwise, since it might be used for project presentations or something similar. However, there will not be a final exam in the course.

Labs:

Section 501: MW 03:00 pm-03:50 pm RMDC 111C
Section 502: MW 11:30 am-12:20 pm RMDC 111C
Section 503: MW 04:10 pm-05:00 pm RMDC 111C

Goals:

This course is intended as an intensive programming experience that integrates core concepts in Computer Science and familiarizes students with a variety of programming/development tools and techniques. Students will primarily work in small teams on month-long projects emphasizing different specializations within computer science. The course focuses on honing good programming techniques to ease code integration, reuse, and clarity.

The primary goal for this class is to have students emerge with strong programming skills, able to address both individual and team programming challenges competently. The class is meant to allow students to improve their programming skills through significant practice.

Objectives:

The expected accomplishments of the students are as follows:

1. Become a confident software developer experienced in the full software development cycle.
2. Become a capable and effective member in a small software development team.
3. Become an effective communicator within the context of software projects.

Outcomes:

The students who take this course should be able to demonstrate the following upon the completion of this course.

1. Knowledge of programming and debugging tools.
2. Knowledge of various programming paradigms.
3. Ability to design and refine large software systems based on rough system requirements.
4. Ability to implement and test software system design.
5. Ability to work as a member of a software project development team.
6. Knowledge of various software development paradigms.
7. Ability to manage software development projects.
8. Ability to write technical documentation regarding software systems.
9. Ability to communicate the overall design and details of software systems.
10. Introductory-level knowledge in database systems, artificial intelligence, and software engineering.

Textbook:

We will be using the following textbook:

- Code Complete, 2nd edition, by Steve McConnell, Microsoft Press, 2004.
  Book web page

Other books that may be drawn from, and that might be useful references include both the first edition of Code Complete, as well as:

- The Practice of Programming, by Brian W. Kernighan and Rob Pike, Addison Wesley, 1999.
- Code Craft, by Pete Goodliffe, No Starch, 2007. (Note: this book is available to read online for free through TAMU).

Computer Accounts:

1. Computer accounts: if you do not have a unix account, ask for one on the CS web page.

Topics to be covered:

Among the topics to be covered in lecture periods are:

- Style considerations in writing code
- Design of software sytems and APIs
- Coding beyond the single component
- Basic collaborative software coding practices
- Design for portability, performance, testability
- Specification and documentation
- Basic software tools and their use
- Object oriented design
- Design patterns
- Testing

- Subject-specific topics related to the team projects

Though many topics will overlap, this course is not intended to be as in-depth or comprehensive as a standard software engineering course, which focuses more on project management - students may take the software engineering class after taking this class.

Note: You should expect to spend a significant amount of time (>10 hours/week) outside of class time on programming projects. This may require meeting with team members outside of the class/lab periods.

See the Weekly Schedule section for more details.

Grading:

There will be three major projects in the course, each counting for 28% of the overall grade. Specific grading practices for each project will be announced when that project is given out, but the grade may include factors such as evaluation of code clarity, teamwork, etc. Peer evaluation may be used as a significant contributing factor to these grades. The remaining 16% of the grade will be an individual grade based on individual exercises, quizzes, participation in the course survey, and an evaluation of class participation (which might include participation in code reviews). Individual assignments will be small programming assignments to be completed on an individual basis.

The 16% of the grade will start off as being based totally on instructor judgement of class participation and effort. As the course progresses, any quizzes given out, individual assignments given out, or other specific graded material will note the portion of this individual grade which that quiz/assignment/etc. affects. The remainder of the individual grade will be based on the subjective class participation and effort grade. For example, if there are 8 quizzes at 1% each, one individual assignment at 4%, and participating in the course evaluation is 2%, then the remaining 2% is based on the subjective evaluation.

The grading scale expected to be used is ? 90% > B ? 80% > C ? 70% > D ? 60% > F. In addition to this, the instructor reserves the right to provide a relative or absolute curve to the final class grade (note that such a curve has not always been applied, and should not be assumed). Also, the instructor may raise the grades of any students near a borderline based on a subjective evaluation of class participation and effort.

Academic Integrity:

AGGIE HONOR CODE: An Aggie does not lie, cheat, or steal or tolerate those who do.

Upon accepting admission to Texas A&M University, a student immediately assumes a commitment to uphold the Honor Code, to accept responsibility for learning, and to follow the philosophy and rules of the Honor System. Students will be required to state their commitment on examinations, research papers, and other academic work. Ignorance of the rules does not exclude any member of the TAMU community from the requirements or the processes of the Honor System.

For additional information please visit: http://www.tamu.edu/aggiehonor/

For this class, certain aspects of the honor code need to be clarified.

1. There may be times in this course where you or your team make use of external code/software/libraries. Whenever this is done, you must make sure that, in addition to following any restrictions on that code itself, you clearly document what the source of the external code was, and how it was used.
2. There may be cases in this course where you or your team seeks outside assistance related to one of the projects. Any assistance received from people other than members of your team, the professor, teaching assistant, or peer teacher needs to be clearly documented.
3. You will be working in team environments in this course, and your work as a team will be used to determine grades. As such, it is your responsibility, when asked, to:
    - accurately describe the work that you have done on a team project. Claiming credit for work that you have not done or that others did instead is a violation of the code.
    - accurately describe (to the best of your knowledge) the performance of other team members. "Covering" for another team member (claiming they did more work than you know they did) or "spiking" them (claiming they did less work than you know they did) are examples of honor code violations.
    - prevent (as best you can) or report (known) violations of the honor code by your other team members. You share responsibility when a project is turned in; if you are aware of a teammate having violated the code in his/her work on the project, and do not report it, you are claiming credit for that violation yourself.

If there are any questions or concerns about whether an action is appropriate, you should check with the professor or teaching assistant first. If in doubt, assume that it is not appropriate.

Course Policy:

- Attendance: Attendance is expected in the course, and may be recorded in both lectures and labs. 16% of the course grade will be based on individual evaluation of assignments and class participation, and repeated absences may negatively affect the grade. In addition, students might miss quizzes, which will not be made up without prior approval. Students with absences should notify the instructor ahead of time about any planned missed classes or labs. Unapproved absences may result in a lower course grade.
- Late Assignments: Each project will have a specified date and time at which it is due, and dates and times for which various intermediate parts of the project are due. Projects that are turned in late will have a penalty applied to the overall project grade, which will affect the grade given on that project for all team members (if individual reports are late, those will affect only the grade for that team member). The total number of minutes, m, that assignments within a project are late will be added up. The final grade on the project will be multiplied by 0.9998m. For example, if the project is 1 hour late, you lose a bit over 1%. If it is one day late, you lose about 25%. After 3 days, you're down to 42% of your grade lost.
- Quizzes: The instructor may give out small quizzes in class to ensure that students are continuing to follow course material. Any quizzes will be short and simple, related to recent course discussions or reading assignments. Quizzes will affect only the 16% "individual" grade portion on the class. Makeup quizzes will not be offered without prior approval.
- Course Evaluation: An online course evaluation will be used for the class. Participation in this evaluation will affect the 16% portion of the course grade (the specific amount will be

announced toward the end of the course).
- Communication: A class web page (listed at the top of this syllabus) will be maintained throughout the semester. Students are responsible for checking both the web page and email regularly for class updates.
- Code Documentation: A key part of this class is understanding the importance of clear code construction and documentation. So, when assignments are graded, a significant portion of the grade may be based on an evaluation of how well the code is written, and how easy it is to follow. Just producing code that "works" is not sufficient; it will be your responsibility to produce code that the grader can follow.

Students with Disabilities:

The Americans with Disabilities Act (ADA) is a federal anti-discrimination statute that provides comprehensive civil rights protection for persons with disabilities. Among other things, this legislation requires that all students with disabilities be guaranteed a learning environment that provides for reasonable accommodation of their disabilities. If you believe you have a disability requiring an accommodation, please contact the Department of Student Life, Services for Students with Disabilities, in Cain Hall or call 845-1637.

II. Resources

1. TBA

III. Weekly Schedule and Class Notes

- **Lecture notes**: all notes will be uploaded in this directory.
- It is **your responsibility** to download, print, and bring the notes to the class. Notes will be available 24 hours before each class.
- See the **TAMU Calendar** for breaks, etc.
- More detail will be available as we go along.

Week	Date	Lecture	Lab	Notices	Deadlines	Notes
1	8/27	Introduction; Project 1: Intro to Databases [Chapters 1, 9.1, 9.2]	No lab today			slide01 slide02
1	8/29	Project 1: Entity-relationship model, relational DB, SQL Schema	IDE, SVN, Team assignment	Project 1 announced		slide03 slide04
1	8/31	Naming, Style, Commenting [Chapters 11.1, 11.2, 31]	--			extra01 extra02 extra03

2	9/3	Project 1: SQL queries, Database implementation	Debugger use; Project 1 Design, DB engine [Chapter 23]		Project 1 Design Documents Due	slide05 slide06
2	9/5	API Design, Software Design Principles [Chapter 5]	Project 1, DB engine			slide07 slide08
2	9/7	Testing and Test-Driven Development (TDD) [Chapter 22]	--			slide09 slide22
3	9/10	Debugging, Software development approaches [Chapter 5.1, 5.2, 5.3, 8.1, 8.2, 8.3]	Project 1: Parsing		Project 1 DB Engine code due	slide10 slide11
3	9/12	Agile Development, Collaborative Code Development	Project 1: Parsing, DB Engine Code Review/Debug			slide12 slide17
3	9/14	Project 1 intermediate review	--			
4	9/17	Design patterns [Chapter 21]	Project 1: Integrating parser and DB engine		Project 1: Parser code due	slide18
4	9/19	Code portability, Code performance [Chapter 24, 25, 26]	Project 1: Integrating parser and DB engine			slide19
4	9/21	Code tuning [Chapters 25, 26]	--			slide19 slide20
5	9/24	Project 2: Introduction to AI	Project 1: DB application coding		Project 1 Parser+DB engine integrated code due	slide13
5	9/26	Project 2: Search	Project 1: DB application coding			slide14
5	9/28	Project 2: Game search	--			slide14
6	10/1	Project 2 Announcement [General reading: Chapter 6.1-6.4]	Project 2 design	Project 2 announced	Project 1 final version due	
6	10/3	Project 2: Network protocols and socket programming	Project 2: game mechanics			slide15
6	10/5	Project 1 presentation (presentation by top team in	--			

		each section)				
7	10/8	Advanced AI: Neuroevolution	Project 2: AI game mechanics		Project 2 design documents due	
7	10/10	Advanced AI: Intro to machine learning	Project 2: AI engine			
7	10/12	SOLID principles	--			extra05
8	10/15	**No class**	Project 2: AI engine		Project 2 Game mechanics code due	
8	10/17	**No class**	Project 2: AI engine			
8	10/19	Project 3: Android introduction: Lecture notes by Dr. Jaerock Kwon	--			
9	10/22	Project 3: Android app fundamentals: Lecture notes by Dr. Jaerock Kwon	Project 2: Socket programming		Project 2 AI engine due	
9	10/24	Project 2 intermediate review	Project 2: Socket programming			
9	10/26	Project 3: XML	--			slide21
10	10/29	Project 3 announcement	Project 2 status check	Project 3 announced	Project 2 final version due	
10	10/31	Brain-storming session for Project 3 app ideas	Android SDK installation and testing, emulator test run			
10	11/2	Project 2 presentation (live competition)	--			
11	11/5	**No class**	Android SDK user interface		Project 3 Design documents due	
11	11/7	**No class**	Android SDK: graphics			
11	11/9	**No class**	--			
12	11/12	**No class**	Project 3 status check		Project 3 GUI code due	
12	11/14	Project 3 intermediate review	Project 3 status check			
12	11/16	**No class**	--			

13	11/19	**No class**	Project 3 status check	Project 3 core algorithm implementation
13	11/21	**No class**	--	
13	11/23	**No class (Thanksgiving)**	--	
14	11/26	**No class**	Project 3 status check	
14	11/28	Final project presentation (all teams)	--	
14	11/30	Final project presentation (all teams)	--	
15	12/3	Final project presentation (all teams)	--	Project 3 final version due

IV. Credits

Most of the course content and lecture slides were originally developed by Prof. John Keyser. Thanks to Long Mai and Allen Hurst at Improving Enterprises for valuable feedback.
