CPSC 315 Programming Studio: Spring 2016

Syllabus

NEWS: *1/20/16, 10:19AM (Wed)*

- [01/20/2016] Class room is HRBB 204 (not 124).
- [01/20/2016] slide01.pdf slightly updated (10am, 1/20).
- [01/20/2016] Team assignment: Read-only board. Also see the eCampus groups (for email, etc.) ecampus.tamu.edu.
- [01/20/2016] Project 1 announced
- ----
- [01/20/2016] Small-Group code review for education, by Phillip Guo (Communications of the ACM)
- [01/20/2016] Teamwork Brown Bag presentation by Robert Lingard at Calstate, Northridge
- ----
- [01/18/2016] Fill out the Programming Proficiency Survey by Tuesday 1/19 5pm
- [01/18/2016] Course web page goes online.

Read-Only Bulletin Board.: 1/14/16, 10:33AM (Thu)

Page last modified: 1/20/16, 10:19AM Wednesday.

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

I. General Information

Instructor:

Dr. Yoonsuck Choe
Email: choe at tamu.edu
Office: HRBB 322B
Phone: 979-845-5466
Hours: MWF 2:40pm-3:30pm

Teaching assistants and peer teachers:

Teaching Assistant (labs): Randall Reams
Email: randall.c.reams at gmail.com
Office hours: Tue/Thu 9:30am-10:30am
Office: ETB 2021

Grader TBA

Grader TBA

Peer teacher: Cody Taylor
Office: RDMC 111J
Hours: MW 11:30am-12:30am Email: cody.taylor49 at tamu.edu

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, HRBB 204

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 2/3 of the allocated 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: taught by the TA

Section 207 (Honors): TR 12:45pm-2 pm RDMC 111J
Section 509: TR 11:10am-12:00pm RDMC 111J

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 CSE 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 (see below). The remaining 16% of the grade will be based on attendance (6%: attendance sheets will be circulated) and the two online quizzes (5% each).

The projects are scored by the team, however, different individual contribution can lead to differential grades given the same team score.

- i = individual score
- t = team score
- c = your contribution (X %, e.g., 25%). Sum of c for all members in the team should equal 100.
- d = contribution divisor (Y %, e.g., 25% for a team of 4, 33.3% for a team of 3, etc.)
- Formula: i = min(sqrt(c/d) * t, 110).
- Examples:
    1. Team got 90 and your contribution was 25% for a 4-person team.

        i = min (sqrt(25/25) * 90, 110) = min(1*90, 110) = 90.

    2. Team got 85 and your contribution was 35% for a 4-person team.

        i = min (sqrt(35/25) * 85, 110) = min(100.57, 110) = 100.57.

    3. Team got 95 and your contribution was 20% for a 3-person team.

i = min (sqrt(20/33.333) * 95, 110) = min(73.587, 110) = 73.587

The grading scale expected to be used is: >90 = A; >80 = B; >70 = C; >60 = D; all else F.

Honors credit:

Those who are taking this for the honors credit (section 207), the following will be different from the regular section.

1. Smaller team size (default team size of 3, as opposed to 4 in the regular section).
2. Additional project requirements (to be announced in the project assignment document).

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://aggiehonor.tamu.edu/

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 mandatory 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 0.9998^{m}. 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.
- 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 Disability Services, currently located in the Disability Services building at the Student Services at White Creek complex on west campus or call 979-845-1637. For additional information, visit http://disability.tamu.edu.

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 MWF	Lab Tuesday/Thursday	Notices	Deadlines	Slides
1	1/18	**No Class: Martin Luther King day**	Tue: IDE, programming proficiency survey.			
1	1/20	Introduction; Project 1: Intro to Databases, Entity-relationship model, relational DB [Chapters 1, 9.1, 9.2];	Thu: IDE, GIT, Team assignment			slide01 slide02 slide03
1	1/22	Project 1: SQL Schema; Project 1 announcement				slide03 slide04
2	1/25	Project 1: SQL Schema;	Debugger use; Project 1 Design, DB engine [Chapter 23]			slide03 slide04
2	1/27	Project 1: SQL queries, Database implementation; API Design	Project 1, DB engine		Project 1 Design Documents Due	slide05 slide06 slide07
2	1/29	Software Design Principles; Testing and Test-Driven Development (TDD) [Chapter 5, Chapter 22] (**SELF-STUDY**: Naming, Style, Commenting [Chapters 11.1, 11.2, 31])				slide08 slide09 slide10 naming style commenting 0909
3	2/1	Software Design Principles; Testing and Test-Driven Development (TDD) [Chapter 5, Chapter 22]	Project 1: Parsing		Project 1 DB Engine code due	slide08 slide09 slide10
3	2/3	Debugging, Software development approaches; Agile Development [Chapter 5.1, 5.2, 5.3, 8.1, 8.2, 8.3]	Project 1: Parsing, DB Engine Code Review/Debug			slide11 slide12 slide13
3	2/5	Agile Development; Collaborative Code Development; Project 1 intermediate review				slide13 slide17
		Agile Development; Collaborative Code	Project 1:		Project 1:	slide13

4	2/8	Development; Project 1 intermediate review	Integrating parser and DB engine		Parser code due	slide17
4	2/10	Design patterns; Code portability [Chapter 21, 24]	Project 1: Integrating parser and DB engine			slide18 slide19
4	2/12	Code portability; Code performance; Code Tuning [Chapter 25, 26]				slide20 slide21
5	2/15	Code portability; Code performance; Code Tuning [Chapter 25, 26]	Project 1: DB application coding		Project 1 Parser+DB engine integrated code due	slide20 slide21
5	2/17	Project 2: Introduction to AI, Search	Project 1: DB application coding			slide14 slide15
5	2/19	Project 2: Game Search				slide15
6	2/22	Project 2: Game Search ; Project 2 Annoucement	Project 1 status check; Project 2 design	Project 2 announced	Project 1 final version due	slide15
6	2/24	Network protocols and socket programming [General reading: Chapter 6.1-6.4]	Project 2: game mechanics			web_link;
6	2/26	Project 1 presentation				web_link;
7	2/29	Advanced AI: Intro to machine learning	Project 2: game mechanics / socket programming		Project 2 design documents due	
7	3/2	Advanced AI: Neuroevolution	Project 2: socket programming			slide16
7	3/4	Advanced AI: Neuroevolution				slide16
8	3/7	Project 3: Android introduction	Project 2: AI engine		Project 2 Game mechanics and server code due	kwon-android01-choe
8	3/9	Project 3: Android programming	Project 2: AI engine			kwon-android02-choe kwon-android03-choe
		Project 3: XML; SOLID				slide22

8	3/11	principles				slide23
9	3/14	**Spring Break**				
9	3/16	**Spring Break**				
9	3/18	**Spring Break**				
10	3/21	**No class**	Project 2: client GUI			
10	3/23	**No class**	Project 2 status check		Project 2 AI engine due	
10	3/25	**No class**				
11	3/28	Project 3 announcement	Android SDK installation and testing, emulator test run	Project 3 announced		
11	3/30	**No class**	Android SDK installation and testing, emulator test run		Project 2 final version (including GUI client) due	
11	4/1	**No class**				
12	4/4	Project 2 code review and live demo	Android SDK user interface		Project 3 Design documents due	
12	4/6	**No class**	Android SDK user interface			
12	4/8	**No class**				
13	4/11	**No class**	Android SDK: graphics		Project 3 user interface code due	
13	4/13	**No class**	Android SDK: graphics			
13	4/15	**No class**				
14	4/18	Project 3 code review	Project 3 status check		Project 3 core algorithm implementation due	
14	4/20	**No class**	Project 3 status check			
14	4/22	**No class**				
15	4/25	**No class**	Project 3 status check			
			Project 3 status			

15	4/27	Final project presentation	check	
15	4/29	Final project presentation		
16	5/2	Final project presentation	Project 3 status check	Project 3 final version due
16	5/4	Final project presentation (Redefined Friday, class meets on **Tuesday 5/3**)	Project 3 status check	

IV. Credits

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