CPSC-4175, Fall, 2017

Charles Carter

September 14, 2017

1 Grading

Evaluation Item	Weight	Metrics
Homework	25%	Homework is due at noon on Mondays for the chapters covered. For example, the homework on chapter 1 will be due at noon on Monday, August 21, 2017. Homework is assessed on a scale of 0 to 100. Late homework will receive a grade of 0, i.e., late homework will not be accepted. For exceptions to this rule, see me. I do not grade on correctness but on effort. Wrong answers receive full credit, *as long as the stsdent has made a reasonable attempt to answer the question.*
Quizzes	25%	In-class quizzes are given at the beginning of class, and will mostly cover in class labs, lectures, and discussions. Quizzes are graded on a scale of 0 to 100. Failure to complete a quiz due to absence or tardiness will result in a grade of 0. For exceptions to this rule, see me.
Graded Exercises	25%	We will complete a number of graded exercises, which will consist of engineering artifacts, source listings, research papers, and other hands-on activities. Exercises will be done in class, outside of class, or both as appropriate for the exercise. Exercises are graded on a scale of 0 to 100. Generally, exercises are due Fridays at noon for the week of the assignment. The first exercise is due at noon, Friday, August 18, 2017. There is no late policy on graded exercises, although failure to complete an exercise will result in a grade of 0. Generally, class discussions will depend on completion of exercises, so it's best to have these exercises completed prior to class.
Group Project	25%	Class projects are 16 week projects, and will be peer graded. These will consist of four iterations of four weeks each. Deliverables include (1) a final requirements specification, (2) application documentation, (3) test suites, and (4) source listings. See the project description for details.
Totals	100%	

2 How to turn in work

2.1 PER INDIVIDUAL

Every student will create a local git repository and a remote Github repository for this class. Name the remote repository 'CPSC-4175', I don't care what you name your local repository as I'll never see it, but it may be less confusing if you use the same name for both. As a subdirectory in your class repository, create a directory named **homework**. You will have other assignments during the term, and you will want to create other subdirectories from time to time.

Do your homework using Markdown. DO NOT email your homework to me. Push your work to the remote Github repository and make sure it's in the correct directory. Send me ONE email, subject like 'CPSC-4175 individual repo' and message like '<your name>, <repo URL>'. Example:

`Charles Carter, https://github.com/ccc31807/CPSC-4175`

2.2 PER TEAM

Every team will create ONE remote Github repository for their team project. Name the remote repository 'TEAM ZULU PROJECT' (using your assigned alphabetic identifier). As subdirectories in your class repository, create a directory structure like this:

```
/analysis
/design
/implementation
/tests
/userdocs
```

Push your work to the remote Github repository and make sure it's in the correct directory. Send me ONE email, subject like 'CPSC-4175 project repo' and message like '<team name>, <repo URL>'. Example: ă `Team Zulu, https://github.com/ccc31807/CPSC-4175`

3 Schedule

Week	Date	Chapter	Topic	Project
1	August 14	Chapter 1	Scope	Iteration 1 Analysis
2	August 21	Chapter 11	Analysis	Iteration 1 Design
3	August 28	Chapter 12	Requirements	Iteration 1 Implementation
4	September 4	Chapter 13	OOAD	Iteration 1 Testing
5	September 11			Iteration 2 Analysis
6	September 18	Chapter 14	Design	Iteration 2 Design
7	September 25	Chapter 15	Implementation	Iteration 2 Implementation
8	October 2	Chapter 6	Testing	Iteration 2 Testing
9	October 9	Chapter 3	Process	Iteration 3 Analysis
10	October 16	Chapter 2	Life Cycle	Iteration 3 Design
11	October 23	Chapter 9	Estimation	Iteration 3 Implmentation
12	October 30	Chapter 5	Tools	Iteration 3 Testing
13	November 6	Chapter 18	Technologies	Iteration 4 Analysis
14	November 13	Readings	Version Control	Iteration 4 Design
15	November 20	Readings	Security	Iteration 4 Implementation
16	November 27	None	Project Presentations	Iteration 4 Testing
17	December 4	None	Project Evauations	

4 Project

The group projects comprise the heart of this class. The purpose is to afford the opportunity for students to actually practice the concepts covered by readings and class discussions. Each project will consist of a substantial software project, described below. A project should be complex enough to contain several modules, but simple enough so that a substantial part of the project may be completed in 16 weeks.

We will be using two person groups. The primary reason for this is to give students experience in pair programming. Pair programming consists of a methodology where two people use one computer. Studies have shown that two people working in pairs get more accomplished than two people working as individuals. By the end of class, Thursday, August 17, you should have found your partner. I will assign partners to all

students who do not have a partner at the end of class. I will also discuss group alterations during the terms for those students who need t discuss problems working with their partner.

Criteria for grading includes (1) a final software requirements specification, (2) final design documentation, (3) implementation source listings including (a) the user interface, (b) the data store, and (c) the logic or business rules module, (4) automated testing suites, and (5) system documentation including (a) a technical reference, and (b) a user guide.

You may choose your own project. You can implement it using any technology you wish. Here is a list of sample projects for you to consider.

School system This may consist of a student module, an instructor module, a curriculum module, and provides for the assignment of instructors to classes, students to classes, course management, student management, an d instructor management. As example would be the CSU student information system.

Electronic learning system This may consist of a collection of courses. Each course may allow for tests, quizzes, exercises, research papers, discussions, etc. As example would be Cougarview.

An auction system. This my consist of a sellers module, a bidders module, a purchase module, and provide for different kinds of sales (i.e., open bidding, bidding with reserve, buy now, etc.). It would include user registrations and appropriate databases of auctions, transactions, etc. An example would be eBay.

Messaging system This would include user registrations, posting messages, sending messages, composing groups, and similar functionality. An example would be Twitter.

Encyclopedia This would include editor registrations, posting entries, editing entries, search functions, analytical functions, and similar functionality. An example would be Wikipedia.

Electronic voting system This would include voter registrations, candidate qualification, casting ballots with appropriate authentication, and calculation of results. An example would be the current voting system in Georgia.

Networking system This would include user registrations, posting employment data, school data, certification data, allow the posting of jobs, etc. An example would be LinkedIn.

Online store This would implement the display of merchandise, a shopping cart, a payment module, and inventory control mechanism, etc. An example would be Amazon.

Database GUI This would implement a graphical front end to a database. Modules would (1) create or drop databases, (2) create, drop, and alter tables, (3) insert, update, or delete data, and (4) run simple queries. An example would be TOAD.

Organization system This might include member registration, a member directory, an event calender, a newsletter, a photo album, a FAQ, or other functions suitable for organizations. Examples would include a social organization, a service organization, a band or orchestra, a church, an athletic team, etc.

Automated teller machine This might include customer authentication, checking account status, acceptance of deposits, and dispensing cash.

A strategy game At a minimum, this would consist of a user interface, a data source, and a logic engine implementing the game rules.