EECS 3311 3.00: Software Design

Sections A & E, Fall 2020

Course Syllabus

1 Course Policies

To ensure a smooth, fair, and effective online delivery of this course:

1. **Plagiarism**: When you submit your **labs** or **project**, you claim that it is **solely** your work. It is considered as **an violation of academic integrity** if you copy or share **any** parts of your work (e.g., code, diagrams) during **any** stages of your development. The instructor and TAs **will** examine **all** submitted code, and suspicious submissions will be reported *immediately* to Lassonde as *a breach of academic integrity*. **We do not tolerate academic dishonesty**, so please be fully responsible for your learning.

2. **Online Submission/Assessment**: **Stringent deadlines** are imposed on all **weekly study quizzes** (to be completed online on eClass/Moodle), as well as **labs** and the **project** (to be submitted via the `submit` command *electronically* to the EECS server). An **exam** is scheduled online (via eClass) with **stringent timing requirements** (start time, duration, and end time to be announced by the registrar office).

   All announced deadlines are in the Eastern Time Zone (Toronto time). Students on a different time zone must figure out the corresponding local time.

   Students are responsible for **taking proactive steps and/or seeking assistance** well in advance to ensure that their technical setup (e.g., reasonably stable internet connection, a working computer which does not freeze sporadically) allows them to complete and submit each online assessment item (quiz, lab, project, exam) in time.

   Rationales for this policy are to: **urge** students with technical issues to take steps or seek assistance to fix/improve them (otherwise, how can they benefit from the online setting in the first place?); and **discourage** students trying to take an unfair advantage (e.g., a student ignorant of the submission deadline or starting late may claim technical failure to have an extension, a student who has already seen the exam questions may claim network/computer failure in order to gain extra time or a deferred exam).

   When it comes to assessments, your instructor's priorities are fairness and academic integrity.

3. **No Team Work**: All **labs** and the **project** are to be developed and completed **individually** (i.e., **team work is forbidden**). This is meant for avoiding students having difficulties finding a suitable teammate, as well as potential disputes between teammates (e.g., non-responsiveness, overdue progress, last-minute notice of withdrawal): the online nature of this course would only exacerbate these problems.

4. **Late Enrolment**: Students who are on the waiting list should assume an eventual successful enrolment into the course and are responsible for: **1)** contacting the instructor (`jackie@eecs.yorku.ca`) **within Week 1** for course information (e.g., lecture materials, lab assignments access and deadlines); and **2)** studying lecture videos, attending Q&A sessions, taking quizzes, and submitting lab assignments in time.

   No deadline extension of labs or deferred quizzes will be accommodated.

2 Academic Integrity

Labs/Project

- All labs and the project are to be completed ***individually***: no group work is allowed.
- TAs will perform thorough checks on **all** lab and project submissions: convincingly suspicious submissions will be reported to the Lassonde Student Service for a ***formal investigation*** immediately.
- To protect yourself from ending up a submission that is suspiciously similar to someone else's, you want to ***avoid***:
  - Discussing code-level details about labs/project with anyone.
  - Discussing concrete steps about your solution or someone's solution.
  - Sharing any part(s) of your code (e.g., file transfer via email, SMS, screen sharing via Zoom) at any stage of your development.
  - Giving or receiving instructions about what exactly you should type for a fragment of code.

    (e.g., it is *acceptable* to ask about how to write a loop in general, but *unacceptable* to ask about how to write a loop specifically for solving a problem related to the assignment).
- The best ways to help your fellow students are clarifying instructions and showing them how to use breakpoints/debugger.

Weekly Quizzes

- All quizzes are to be completed ***individually***: no group work is allowed.
- It is considered ***a breach of academic honesty*** if:
  - You collaborate with someone on completing a quiz.
  - After you have attempted the quiz and before that quiz is closed, share your quiz questions with someone.

Reporting Cases

Enforcing the policy of academic honesty not only maintains the ***standard*** of the course, but also ensures ***fairness*** among all students in the class. If you have sufficient reasons to believe that cases of violation are present, let the instructor know and confidentiality will be maintained.

3 Instructor

- Chen-Wei (Jackie) Wang (http://www.eecs.yorku.ca/~jackie/)
- Contact: jackie@eecs.yorku.ca
- Virtual Office: https://yorku.zoom.us/my/jackie.loves.oxford
- Office Hours: 4:00pm – 5:30pm, Mondays, Tuesdays, & Wednesdays; or by Appointments.

4 Prerequisites

- **General Prerequisites**: A cumulative grade point average (GPA) of 4.50 or better over all previously completed Major EECS courses. The GPA computation excludes all EECS courses that have a second digit 5, or are Co-Op/PEP courses.
- LE/EECS 2011 3.00
- LE/EECS 2031 3.00
- SC/MATH 1090 3.00

5 Available Help

- Course forum on eClass: https://eclass.yorku.ca
- Instructor's office hours (4.5 hours weekly)
- Scheduled lab sessions (3 hours weekly)
- Scheduled Q&A sessions (3 hours weekly)

6 Course Description

A study of design methods and their use in the correct construction, implementation, and maintenance of software systems. Topics include design, implementation, testing, documentation needs and standards, support tools. Students design and implement components of a software system.

This course focuses on design techniques for both small and large software systems. Techniques for the design of components (e.g., modules, classes, procedures, and executables) as well as complex architectures will be considered. Principles for software design and rules for helping to ensure software quality will be discussed. The techniques will be applied in a set of small assignments, and a large-scale project, where students will design, implement, and maintain a non-trivial software system.

7 Course Learning Outcomes (CLOs)

Upon completion of the course, students are expected to be able to:

CLO1 Describe software specifications via Design by Contract, including the use of preconditions, postconditions, class invariants, as well as loop variants and invariants.

CLO2 Implement specifications with designs that are correct, efficient, and maintainable.

CLO3 Develop systematic approaches to organizing, writing, testing, and debugging software.

CLO4 Develop insight into the process of moving from an ambiguous problem statement to a well-designed solution.

CLO5 Design software using appropriate abstractions, modularity, information hiding, and design patterns.

CLO6 Develop facility in the use of an IDE for editing, organizing, writing, debugging, documenting designs, and the ability to deploy the software in an executable form.

CLO7 Write precise and concise software documentation that also describes the design decisions and why they were made.

8 Main Textbook

1. **Title**: Object-Oriented Software Construction
   Author: Bertrand Meyer
   Publisher: Prentice Hall, 1997
   Edition: Second Edition

   – This is a classic text on software design principles.

9 Reference Textbooks

1. **Title**: Touch of Class: Learning to Program Well with Objects and Contracts
   Author: Bertrand Meyer
   Publisher: Springer, 2013
   Edition: Second Edition

   – This book describes computational thinking with the Eiffel language.

   – Use this text to learn about design by contract, polymorphism, static typing, dynamic binding, genericity, multiple inheritance, and lambda expressions (agents). These are all topics needed for this course.

   – Visit the book page here with a complete course with slides, videos and exercises.

   – This book is available with online access via Steacie Library (you need your library account for this) here.

2. **Title**: Design Patterns: Elements of Reusable Object-Oriented Software
   Author: Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides
   Publisher: Addison Wesley, 1994.
   Edition: First Edition

   – This is a classic text on a catalogue of extensible and maintainable patterns of code structures.

3. BON (Business Object Notation)

   – The BON method for analysis and design of object-oriented software is a means of extending the higher-level concepts of the Eiffel programming language into the realm of analysis and design aided by a graphical notation akin to but different from UML.

   – BON is described in depth in the book Seamless Object-Oriented Software Architecture by Kim Waldén and Jean-Marc Nerson. Prentice Hall, 1994. The book is out of print but is available as a pdf here.

10 Grading Scheme

		Subtotal
12 Study Quizzes	24%	52%
Exam (Comprehensive)	28%	
Lab0 (Software)	1%	18%
Lab1 (Software)	5%	
Lab2 (Software)	3%	
Lab2 (Design Diagram)	2%	
Lab3 (Software)	5%	
Lab3 (Design Document)	2%	
Project (Software Phase 1)	2%	30%
Project (Software Phase 2)	18%	
Project (Design Document)	10%	

Starting from Week 2, there will be 12 weekly study quizzes (to complement your study of video lectures). Each quiz is worth 2%.

11 Mapping Raw Marks to Letter Grades

According to the Common Grading Scheme for Undergraduate Faculties approved by Senate:

Letter Grade	Grade Point	Interpretation
A+	9	Exceptional
A	8	Excellent
B+	7	Very Good
B	6	Good
C+	5	Competent
C	4	Fairly Competent
D+	3	Passing
D	2	Marginally Passing
E	1	Marginally Failing
F	0	Failing

- For each grading unit you are assigned a **raw mark score** (not necessarily out of 100).
- The raw mark score is **not** a grade as it is merely used to ***rank*** you in the class.
- Also, you will be provided with a ***mapping*** from your raw mark score to a ***letter grade***.
- Each grading unit's mapping represents your instructor's judgement with respect to the ***qualitative descriptions*** of letter grades in the above table.

  e.g., A raw mark score of 76 (out of 100) might be a C, not a B+, after the mapping is applied.

  e.g., A raw mark score of 45 (out of 100) might be a D, not an E, depending on the mapping.
- Weighted sums of these cutoffs will be used as the final cutoffs of the final course grade. See an example here.

12 Scheduled Q&A and Lab Sessions

	Monday	Tuesday	Wednesday	Thursday	Friday
8:30					
9:00					
9:30					
10:00			EECS3311 Lab		
10:30					
11:00					
11:30					
12:00					
12:30					
13:00					
13:30					
14:00					
14:30		EECS3311 Q&A	EECS3311 Lab		
15:00					
15:30					
16:00	Office Hours	Office Hours	Office Hours		
16:30					
17:00					
17:30	EECS3311 Q&A				
18:00					
18:30					
19:00					

– Between Section A and Section E:

- Two of the schedule lecture time slots

  17:30 – 19:00 on Mondays and 14:30 – 16:00 on Tuesdays

  are used to hold (optional, Zoom) Q&A sessions to answer your questions related to the lecture videos.

  Zoom: `https://yorku.zoom.us/my/jackie.loves.oxford`

- Two of the schedule lab time slots

  10:00 – 11:30 and 14:30 – 16:00 on Wednesdays

  are used to answer your questions.

- You are welcome to attend any of the scheduled lab or Q&A sessions.

13 Learning Schedule

Figure 1 illustrates the schedule of required work items. In Week i ($1 < i < 12$):

- On Monday and Tuesday, two Q & A Sessions (live on Zoom, optional) are held for the ***old*** lecture series W_{i-1} (from last week).
- On Wednesday, a ***new*** lecture series W_i is released.
- At ***5:00pm (Eastern Toronto Time)*** on Friday, $Quiz_{i-1}$ (for the ***old*** lecture series W_{i-1}) is closed and $Quiz_i$ (for the ***new*** lecture series W_i) is opened.

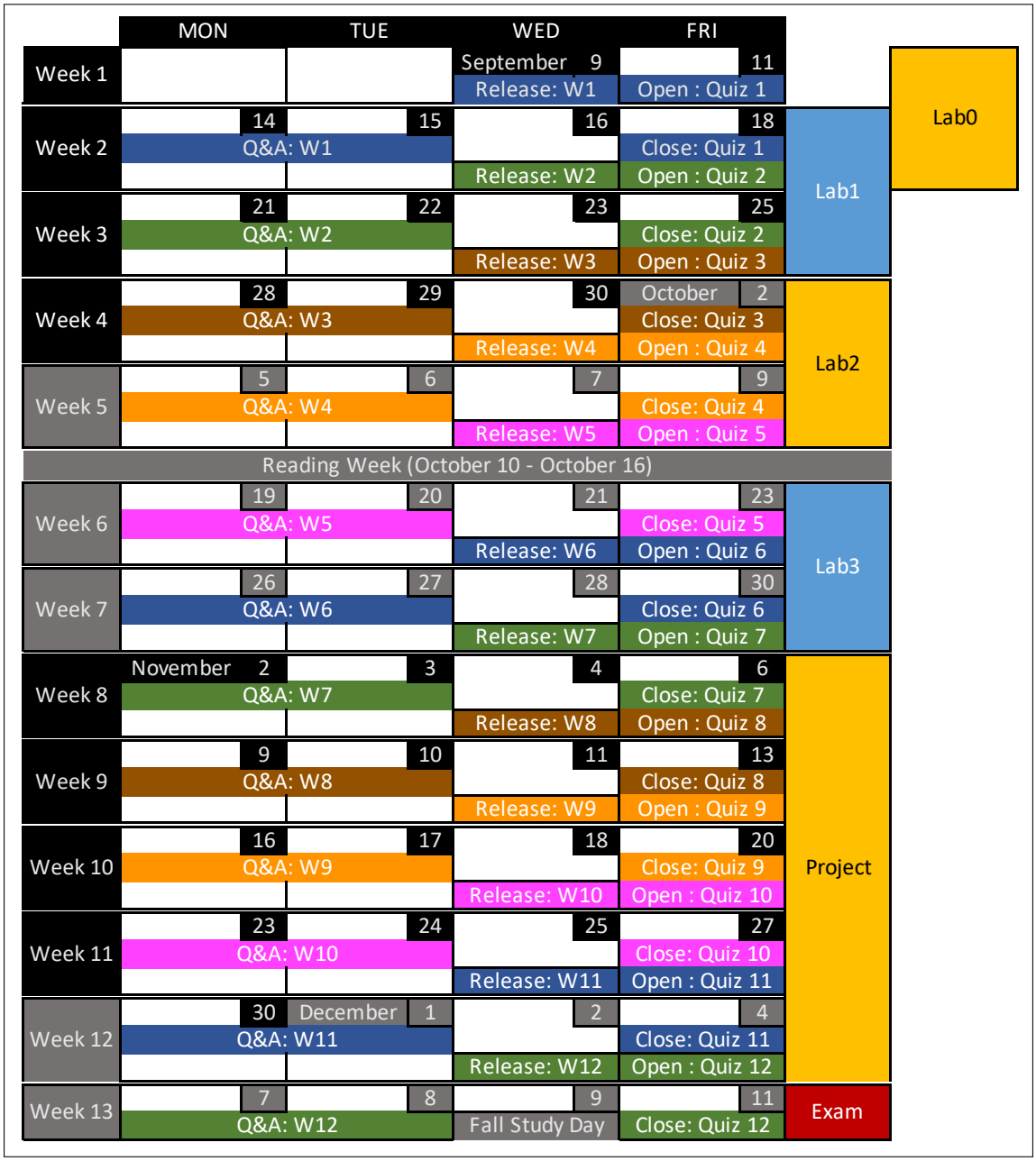

Figure 1: Completing Expected Work Items

14 Weekly Study Quizzes

Q. What is the format of quizzes?

A.

- Each quiz is open-booked (you may consult with the lecture materials or your own notes).
- Each quiz will be opened for its submission for a week.
- Each quiz will consist of between 10 and 20 questions with a variety of forms (e.g., multiple choice, matching answers, true/false).
- There is only **one single attempt** allowed for the quiz.
- As soon as you start the attempt, you will be given **a fix amount of time** to complete (the exact length may vary from quiz to quiz, but in principle between 15 and 30 minutes).
- This means that during the ***one week period*** where the quiz is enabled, you can find any comfortable time slot, such that:
  - You are *mentally* ready: You have finished all video lectures, taken notes, thought about the subjects yourself, and clarified any misunderstandings.
  - You are *physically* ready: You are responsible for making sure that you will be free from distractions ***once the timer of the quiz starts***, and that your internet connection and your computer's conditions should allow you to complete the quiz ***before the timer stops*** (e.g., network failure, your computer freezing).

Q. Can I work on quizzes with my classmates?

A. No. Collaborating with anyone to complete your quiz is considered as a breach of academic honesty, so is discussing with your classmates/friends the quiz questions.

Q. How can I get prepared for quizzes and get the most out of them for my learning?

A. You are encouraged to study the lecture and tutorial materials in groups and, if at all possible, come up with questions so you can speak to Jackie as early as possible.

The facts that each quiz is open-booked, and that you have one-week to complete it, suggest that it is really meant to help you do self-assessment and have a good basis to have discussion with Jackie promptly.

15 Expected Workload

Week	Should be Working
1	• **Complete** at least the ***first 6 to 7 hours*** of Lab0. • Study videos for lecture series W1.
2	• **Complete** the ***last 3 to 4 hours*** of Lab0. • Work on Lab1. • Attend Q&A sessions for lecture series W1. • Study videos for lecture series W2. • **Complete** Quiz1. **Not yet finished with W1? You're already behind!** **Only started with W1? You're very behind!!**
3	• **Complete** Lab1. • Attend Q&A sessions for lecture series W2. • Study videos for lecture series W3. • **Complete** Quiz2. **Not yet finished with W2? You're already behind!** **Only started with W2? You're very behind!!**
4	• Work on Lab2. • Attend Q&A sessions for lecture series W3. • Study videos for lecture series W4. • **Complete** Quiz3. **Not yet finished with W3? You're already behind!** **Only started with W3? You're very behind!!**
5	• **Complete** Lab2. • Attend Q&A sessions for lecture series W4. • Study videos for lecture series W5. • **Complete** Quiz4. **Not yet finished with W4? You're already behind!** **Only started with W4? You're very behind!!**
Reading Week (**Complete**: ETF Tutorial)	

Week	Should be Working
6	• Work on Lab3. • Attend Q&A sessions for lecture series W5. • Study videos for lecture series W6. • **Complete** Quiz5. **Not yet finished with W5? You're already behind!** **Only started with W5? You're very behind!!**
7	• **Complete** Lab3. • Attend Q&A sessions for lecture series W6. • Study videos for lecture series W7. • **Complete** Quiz6. **Not yet finished with W6? You're already behind!** **Only started with W6? You're very behind!!**
8	• Work on Project • Attend Q&A sessions for lecture series W7. • Study videos for lecture series W8. • **Complete** Quiz7. **Not yet finished with W7? You're already behind!** **Only started with W7? You're very behind!!**
9	• Work on Project • Attend Q&A sessions for lecture series W8. • Study videos for lecture series W9. • **Complete** Quiz8. **Not yet finished with W8? You're already behind!** **Only started with W8? You're very behind!!**
10	• Work on Project • Attend Q&A sessions for lecture series W9. • Study videos for lecture series W10. • **Complete** Quiz9. **Not yet finished with W9? You're already behind!** **Only started with W9? You're very behind!!**

Week	Should be Working
11	• Work on Project • Attend Q&A sessions for lecture series W10. • Study videos for lecture series W11. • **Complete** Quiz10. **Not yet finished with W10? You're already behind!** **Only started with W10? You're very behind!!**
12	• Work on Project • Attend Q&A sessions for lecture series W11. • Study videos for lecture series W12. • **Complete** Quiz11. **Not yet finished with W11? You're already behind!** **Only started with W11? You're very behind!!**
13	• **Complete** Project • Attend Q&A sessions for lecture series W12. • **Complete** Quiz12. **Not yet finished with W12? You're already behind!** **Only started with W12? You're very behind!!**

16 Tasks in Weekly Lab Sessions

– All labs and the project are to be completed **individually**: **no group work** is allowed.

Week	Scheduled Lab Date	Task
Week 1	Sep. 9	Lab0
Week 2	Sep. 16	Lab1
Week 3	Sep. 23	
Week 4	Sep. 30	Lab2
Week 5	Oct. 7	
Reading Week (Lab2, ETF Tutorial)		
Week 6	Oct. 21	Lab3
Week 7	Oct. 28	
Week 8	Nov. 4	Project Phase 1
Week 9	Nov. 11	
Week 10	Nov. 18	
Week 11	Nov. 25	Project Phase 2
Week 12	Dec. 2	

17 (Tentative) Lecture Topics

Lecture videos are being actively recorded, so the order of lecture topics below are subject to changes.

Week	Lectures	Topics
1	Lecture 1	• DbC (Design by Contract): Motivation & Terminology • Supporting DbC (Java vs. Eiffel): Pre-Condition, Post-Condition & Class Invariant • Runtime Assertion Checking of Contracts
2	Lecture 2	• Modularity, Modular Design, & Abstract Data Types (ADTs) • Reference vs. Shallow vs. Deep Copies of Objects • Writing Complete Postconditions
3	Lecture 3	• Use of Generics • Software Architecture Design Diagrams • Abstractions via Mathematical Models • Writing Complete Postconditions: Set-Typed Return Values • Case Study: Abstraction of a Birthday Book
4	Lecture 4	• Iterator Design Pattern: Supplier vs. Client • Use of Generics in the Iterator Pattern: Exercise
	Lecture 5	• Modelling Compositions via Expanded Types • `once` Queries, export status • Singleton Design Pattern
5	Lecture 6	• Poor Design Attempts with No Inheritance: Violating Cohesion & Single-Choice Principle • Inheritance: Code Reuse, Expectation, Intuition of Polymorphism & Dynamic Binding • Case Study: Testing of Postcondition • Inheritance: Multi-Level Inheritance Hierarchy, Ancestors, Descendants • Inheritance: Static Types, Dynamic Types, Polymorphism, Dynamic Binding • Inheritance: Casting
Reading Week		

Week	Lectures	Topics
6	Lecture 6	• Inheritance: Polymorphic Argument Values, Return Values, Collections • Inheritance vs. Generics
	Lecture 7	• ETF: Abstract Interface, Abstract State, Model Classes, Modularity • Regression Testing, Acceptance Testing (with an oracle) vs. Unit Testing
7	Lecture 8	• Design of Interactive Systems: Problem, Top-Down (Non-OO) Approach • State (Multi-Panel) Design Pattern: Architecture • State (Multi-Panel) Design Pattern: Supplier vs. Client • Template Design Pattern
8	Lecture 9	• Design of Recursive, Tree-Structured Systems: Problem, Two Poor Design Attempts • Multiple Inheritance • Composite Design Pattern: Architecture • Composite Design Pattern: Supplier vs. Client
9	Lecture 10	• Visitor Design Pattern: Problem, Cohesion, Open-Closed Principle • Visitor Design Pattern: Supplier vs. Client, Double Dispatch, Single Choice Principle
	Lecture 11	• Case Study: Information Hiding via Modularity & Abstraction • Strategy Design Pattern
10	Lecture 12	• Design of Client-Server Apps: Problem • First Design Attempt: Remote Procedure Calls • Observer Design Pattern: Architecture, Supplier vs. Client • Use of `agent` for functional programming • Event-Driven Design

Week	Lectures	Topics
11	Lecture 13	• Inheritance: Sub-Contracting • Sub-Contracting: Static Analysis vs. Runtime Checks
	Lecture 14	• Program Correctness w.r.t. Specification • Weakest Preconditions wp • Hoare Triple: $\{\ Q\ \}\ \mathtt{S}\ \{\ R\ \}\ \equiv\ Q \Rightarrow wp(\mathtt{S},\ R)$
12	Lecture 14	• Rules of wp calculation: assignments, conditionals, sequential composition • Contract of Loops: invariant vs. variant • Correctness of Loops: Proof Obligations & Proofs • Wrap Up of the Course