CPS122/CPS520 - OBJECT-ORIENTED SOFTWARE DEVELOPMENT

Professor:	Russell C. Bjork russell.bjork@gordon.edu	Spring Semester, 2019-2020
Office:	KOSC 242 x4377	MWF 1-2 pm KOS 124
Hours:	MWF 2:10-3:10 pm; Th 1:30-4:30 pm and by appointment	Lab: Tu. 1:15-4:15 pm (beginning 1/21) KOS 118
Course Site:	Blackboard site + http://www.cs.gordon.edu/courses/cps122	

PREREQUISITE: CPS121

CATALOG DESCRIPTION:

Introduces object-oriented analysis and design using a statically-typed programming language, encapsulation, inheritance, polymorphism, use cases, UML modeling, and testing methods Continued development of design and programming skills using UML and Java through weekly laboratories and projects.

COURSE OBJECTIVES:

In general, this course has two major foci: to introduce the object-oriented paradigm, and to set programming in the broader context of software development, which includes requirements analysis, design, verification and validation, and maintenance as well. In particular, upon completion of this course, you should be able to:

1. Explain fundamental object-oriented concepts, such as "object", "class", "method", "encapsulation", "inheritance", "polymorphism", and "event-driven".
2. Write simple programs using most of the capabilities of the Java language.
3. Analyze a moderately complex problem and design a solution to it, using UML notation.
4. Implement and test such a design, using Java.
5. Carry out the process of designing, implementing, and testing a piece of software as a member of a small team.

TEXTS: Carol Britton and Jill Doake, *A Student Guide to Object-Oriented Development* (Oxford: Elsevier, 2005)

Peter Pasquale. *Java Backpack Reference Guide*. (Boston: Addison-Wesley, 2005)

COURSE TECHNIQUES AND PROCEDURES

Since this course is primarily concerned with the development of certain skills and habits, regular practice with evaluation will be the heart of the course. For each unit of material, you will be asked to read a portion of the text book, and to do the short answer questions at the end of the chapter.

Class sessions will include a discussion and amplification of the material in the text and the presentation of further examples and supplementary material. You should not expect to grasp

everything presented in the text when you first read it; however, you should note areas that are unclear to you and be prepared to raise questions about them in class.

As is true with any skill, the only way you can really learn software development is by doing it. For this reason, you will have opportunities to practice what you are learning by doing homework problems, programming projects and weekly laboratories. Additional practice will come from an ongoing design and programming project which you will work on as part of a small team of students, with various portions of the project being due throughout the term. You should look on this as being your key learning experience in the course.

COURSE REQUIREMENTS AND EVALUATION:

1. You will be expected to read most of the Britton/Doake text, as assigned in the schedule below. (Reading assignments should be completed BEFORE the class hour in which the topic is discussed, as specified in the schedule below.) However, our classroom discussion will not rigidly follow the order of material in the text, nor will it be confined to material covered there.

2. Each chapter in Britton/Doake ends with a set of "Quick check questions" that are designed to be done when you read the chapter. You should answer them as part of you reading. On days for which there is a reading assignment in Britton/Doake, we will discuss the corresponding quick check questions, and they will be graded on a "done/not done" basis. I don't expect you to get the "right answer"; all I'm looking for is a good-faith effort prior to class. At the end of the semester, 5% of your final grade will be based on your faithfulness in having answered these questions when due - whether or not your answer is correct. (If the reading of a textbook chapter is broken up into multiple assignments, the schedule will indicate which quick check questions go with each portion of the book assigned. **If there is no explicit statement in the schedule, you should do <u>all</u> the quick check questions**.)

3. Some exercises done in class will also be included with the above.

4. Most chapters in Britton/Doake include a set of Exercises at the end. Though these will not be formally assigned, we will do a number of them in class sessions. For this reason, you should be sure to bring your book with you to class when we are discussing topics in it.

5. The DePasquale book is basically a reference, rather than a traditional text book. There are no formal reading assignments in it.

6. Weekly laboratories will focus on gaining practical experience with the material covered in the book and/or in lecture. Lab assignments will be posted on Blackboard ahead of time, and **<u>must</u>** be read over carefully **<u>before</u>** coming to lab. In some cases, you will be explicitly directed to study certain material in preparation for the lab. For most laboratories, there will be a writeup to turn in. There may also be a quiz given at the start of the lab hour (based on your reading of the lab assignment and any assigned pre-lab preparation) and/or a quiz based on the work done in lab given at the end of lab. Each lab with a formal writeup/quizz(es) will account for 2% of the course grade (20% total for ten such labs). See schedule below for the tentative lab emphases.

7. In the first half of the semester, you will do two individual programming projects designed to improve your programming skills and help you gain familiarity with using Java. These projects

must be done in accordance with the handout "Guidelines for Computer Science Projects", which will be distributed with the first project. You are expected to read these carefully and comply with them exactly. Each project will be worth 10% of the final course grade.

8. Throughout much of the semester, you will work on an ongoing design and programming project as part of a team of two or three students. Portions of this project will be due at different times throughout the term, and some lab time will be devoted to working on this project, as shown in the course schedule below. Each portion will be graded individually when it is turned in. In total, this project will be worth 20% of the final course grade.

9. A mid-term examination (worth 15% of the final course grade) and a final examination (worth 20%) will be given as shown in the course schedule. Each exam will assume familiarity with material in the text, covered in lecture, and/or used in exercises or projects. Exams will be open book (course text only), open notes.

10. Your final grade will be computed on the basis of a weighted sum of the items listed above.

Summary:	Quick check questions and Exercise Set	5%
	Labs	20%
	2 Individual Projects	20%
	Team Project	20%
	Exams	35%
		100%

The following are minimum guaranteed grades for the percentages indicated:

	93% - 100%: A	90% - 92.9%: A-
87% - 89.9%: B+	83% - 86.9%: B	80% - 82.9%: B-
77% - 79.9%: C+	73% - 76.9%: C	70% - 72.9%: C-
67% - 69.9%: D+	63% - 66.9%: D	60% - 62.9%: D-

ACADEMIC DISHONESTY

From the Gordon College Student Handbook: "Academic dishonesty is regarded as a major violation of both the academic and spiritual principles of this community and may result in a failing grade or suspension. Academic dishonesty includes plagiarism, cheating (whether in or out of the classroom) and abuse or misuse of library materials when such abuse or misuse can be related to course requirements." For the purposes of this course, abuse or misuse of Gordon computer systems or networks related to course requirements will also be viewed as academic dishonesty.

Academic dishonesty will not be tolerated. You know better. Just don't!

POLICY STATEMENT ON EXTENSIONS AND INCOMPLETES:

1. Extensions of the due dates for homework or projects MAY be given in the event of extenuating circumstances (such as illness, personal emergency) IF you submit a brief written request to the professor as soon as possible after the circumstances arise.

2. A grade of Incomplete MAY be given without penalty IF you are unable to complete the course work by the last day of the term due to major illness or other similar emergency. You must apply for this using the form provided by the registrar. Such a request will only be granted if you are substantially up-to-date with your course work and were making good progress in the course up to the time that the difficulty arose. Of course, you must complete all work for the course by the midpoint of the next semester in accordance with College policy.

ATTENDANCE POLICY:

Regular class attendance is expected of all students, and class attendance will be recorded. Absences from class will be classified as "excused" or "unexcused". An excused absence is one where the student misses class for a compelling reason (such as sickness, a field trip for another course, or an athletic competition, but not something like alarm clock issues) and has requested an excused absence. A student may request an excused absence up to three times in the semester by simply notifying the professor via email of the reason for the absence - prior to missing the class if possible. If it is necessary to miss more than three classes, the student must provide written documentation (such as a health center or doctor's note, or a notification from an athletic coach) for additional absences - otherwise they will be considered unexcused. A student who anticipates the need to miss multiple classes due to athletic competitions or other student activities must furnish written documentation, should review the college's attendance policy in the catalog, and must then discuss alternatives to class attendance with the professor at the start of the semester.

If a student has an excused absence from a class where there was a quiz, at the professor's option either the student may make up the quiz or the quiz will not be counted in calculating the final grade. Normally homework or other written work due at a class where the student has an excused absence must be turned in prior to the class, but the professor may choose to allow work to be turned in late without penalty in the case of an unanticipated absence.

At the end of the semester, the student's final average will be reduced 1% for each unexcused absence after the first. A student who has more than 12 unexcused absences will fail the course automatically.

A student who is habitually late will have late arrival for class counted as a half absence for that class, and a student who sleeps through most or all of a given class session will be counted as absent for that class.

STUDENTS WITH DISABILITIES:

Our academic community is committed to providing access to a Gordon education for students with disabilities. A student with a disability who intends to request academic accommodations should follow this procedure:

1. Meet with a staff person from the Academic Success Center (ASC) and provide them with current documentation of the disability.
2. Obtain a Faculty Notification Form from the Academic Success Center, listing appropriate accommodations
3. Submit this form to professors and discuss those accommodations with them, ideally within the first two weeks of classes.

Some accommodations need more time to arrange so communicating early in the semester is important. For more information consult the Academic Success Center webpage: http://www.gordon.edu/academicaccessibility or email asc@gordon.edu

TENTATIVE COURSE SCHEDULE

Date	**Topic(s)**	**Reading**	**Written Work Due**
W 1/15	Course Introduction; Introduction to Object-Orientation		
F 1/17	From Python to Java		
M 1/20	*(Martin Luther King Holiday - no class)*		
T 1/21	Lab 1 - Introduction to Objects		
W 1/22	From Python to Java (continued)		**Start Individual Project 1**
F 1/24	Introduction to Software Development; The Software Lifecycle	Readings posted on Blackboard	**Exercises E-1, 2, and 3 in posted reading**
M 1/27	(continued)		**Individual Project 1 Reading Quiz**
T 1/28	Lab 2 - Completing Classes		
W 1/29	Requirements Elicitation, Specification, and Validation	Britton/Doake ch. 2	**ch 2 Quick Check Questions a-e only**
F 1/31	Use Cases and Initial Functional Tests	Portions of Britton/ Doake ch. 3: pp. 39-55 only	**ch 3 Quick Check Questions**
M 2/3	(continued)		**Individual Project 1 Milestone Due**
T 2/4	Lab 3 - Functional Testing/ Interactive Debugging		
W 2/5	Identifying Objects and Classes	Britton/Doake ch. 4	**ch 4 Quick check questions**
F 2/7	Defining a Class		

M 2/10	Encapsulation, Inheritance, and Polymorphism		
T 2/11	Lab 4 - Creating Classes		
W 2/12	Polymorphism, etc, continued)		**Individual Project 1 Due**
F 2/14	(continued)		
M 2/17	Representing Associations in Java; Collections; Arrays		**Individual Project 2 Reading Quiz**
T 2/18	Lab 5 -Inheritance and Polymorphism		
W 2/19	Associations (continued)		**Start Team Project**
F 2/21	Class Diagrams in UML	Britton/Doake ch. 5	**ch 5 Quick Check Questions**
M 2/24	(continued)		**Individual Project 2 Milestone Due**
T 2/25	Lab 6 - Java Collection		
W 2/26	Class Diagrams (continued)		**Team Project Preliminary Milestone Due**
F 2/28	(continued)		
M 3/2	Review and Catch Up		
T 3/3	Lab 7 - Work Session for Individual Project 2 (no writeup or quiz)		**Individual Project 2 Due at the end of lab**
W 3/4	**MIDTERM EXAM (through Class Diagrams)**		
F 3/6-F 3/13	*(Quad finals and Spring break - no class)*		
M 3/16	Test First Development; Unit Testing with JUnit; Project Class Structure	Portions of Britton/ Doake ch. 6 pp. 147-154	**ch 6 Quick Check questions a-b only**
T 3/17	Lab 8 -Implementing a UML Design part 1		
W 3/18	Identifying Responsibilities; CRC Cards		
F 3/20	Modeling Dynamic Behaviors of Systems; Interaction Diagrams in UML	Portions of Britton/ Doake ch. 6: pp. 155-171	**ch 6 Quick Check questions c-j only; Team Project Milestone 1-1 Due**
M 3/23	(continued)		

T 3/24	Lab 9 - Implementing a UML Design part II		
W 3/25	State and Activity Diagrams in UML	read Britton/Doake ch. 7; skim ch. 8	**ch 7 all Quick Check Questions; ch 8 Quick Check Questions a,b,g only; Team Project Milestone 1-2 Due**
F 3/27	Detailed Class Design and Implementation	Britton/Doake ch. 10	**ch 10 Quick Check Questions a-f only**
M 3/30	Graphical User Interfaces and Event-Driven Programming		
T 3/31	Lab 10 - Implementing a UML Design part III; remaining time used for work Session for Team Project Milestone 1-3		
W 4/1	Graphical UIs (continued)		**Team Project Milestone 1-3 Due**
F 4/3	Architectural Design; Components; Component and Deployment Diagrams in UML; the MVC and Client/Server Patterns	Portions of Britton/ Doake ch. 9 pp. 221-231	**ch 9 Quick Check Questions d-h only**
M 4/6	Design Patterns	Portions of Britton/ Doake ch. 9 pp. 242-245	**ch 9 Quick Check Questions l-n only;**
T 4/7	Lab 11 - Work Session for Team Project Milestone 1-4 (no writeup or quiz)		
W 4/8	Design Patterns (continued)		**Team Project Milestone 1-4 Due; (Individual) Quiz on material furnished by the professor for 1-4**
F 4/10	*Good Friday - no class*		
M 4/13	*Easter Monday - no class*		
T 4/14	Lab 12 - Graphical User Interfaces		

W 4/15	User-Interface Design	Portions of Britton/ Doake ch 9 pp 231-235	**ch 9 Quick Check Question i only; Team Project Milestone 2-1 Due**
F 4/17	Quality Assurance; Preconditions, Postconditions, and Invariants; Testing Strategies; Inspection; Correctness Proofs		
M 4/20	(continued)		
T 4/21	Lab 13 - Work Session for Team Project Milestone 2-2 (no writeup or quiz)		
W 4/22	Quality Assurance (continued)		**Team Project Milestone 2-2 Due**
F 4/24	Cohesion and Coupling	http://en.wikipedia.org/wiki/Cohesion_(computer_science); same site: Coupling_(computer_science)	
M 4/27	Reuse, API's		
T 4/28	Lab 14 - Work Session for Team Project Milestone 3-1 (no writeup or quiz)		
W 4/29	Exceptions		**Team Project Milestone 3-1 Due**
F 5/1	Input-Output		
M 5/4	(continued)		
T 5/5	*In the rest of the world, today is a Tuesday - but - by administrative decree, at Gordon, today is a Thursday, so no lab!*		
W 5/6	Review and Catch up		**Team Project Milestone 3-2 Due; (Individual) Project Quiz**
R 5/7			**Team Project Assessments emailed to the professor**

M 5/11 - 2:30-4:30 - FINAL EXAM
(Cumulative, with particular emphasis on material since the Mid-Term)

Relationship to *Massachusetts Digital Literacy and Computer Science Curriculum Framework*

Standard	How taught and assessed in course
3-5.CS.a-d, 6-8.CS.a-d, 3-5.CT.a-e, 6-8.CT.a-e Consistent with the natural progression of these standards, and the maturity of our students, some standards are so widely known that we teach them by exception. By assessing students in the class, we are able to teach them individually when formative assessments reveal a need	Some are taught and assessed as part of related 9-12 standards; others are taught individually as needed based on assessment of classroom and lab interaction and assessments of related 9-12 standards.
9-12.CT.a.1 Discuss and give an example of generalizing and decomposing aspects of a problem to solve it more effectively.	Students learn the Unified Modeling Language (UML) in lecture and use it extensively in working on a team project over the course of the second half of the semester. The project is developed as a set of graded milestones.
9-12.CT.b1. Recognize that the design of an algorithm is distinct from its expression in a programming language.	(ditto)
9-12.CT.b.2 Represent algorithms using a structured language.	(ditto)
9-12.CT.c.2 Create an appropriate multidimensional data structure that can be filtered, sorted, and searched.	In Project 1, students develop a editor for images represented as a 2-dimensional array that can be operated on by operations on individual pixels applied to all pixels; structural operations such as flips, shifts, rotate and changing size, and filtering operations such as blur, sharpen, and edge detection.
9-12.CT.d.3 Select the appropriate data structures to represent information for a given problem.	In lab 8 students are required to choose appropriate structures to represent the collections needed for a series of labs (8-10)
9-12.CT.d.11 Engage in systematic testing and debugging methods to ensure program correctness.	Students learn to use the JUnit facility in a lab which is assessed by a writeup and quiz, and are required to use it for subsequent labs and the team project above.
9-12.CT.d.12 Demonstrate how to document a program.	See team project above. Also, students are required to make appropriate use of javadoc-style comment for all labs and projects in the course.