Computer Science Department
cs.salemstate.edu

SYLLABUS **Fall, 2021**

CSC 115 Software Design and Programming II **4 cr. DII**
Prerequisite(s): CSC110 or ITE210

Instructor: Beifang Yi
email: byi@salemstate.edu
Office: MH 211A/Online/Zoom
Hours: TR (9:25-10:40am)
WF (12:15-1:30pm)
Phone: (978) 542-7246/Zoom
Web Site: http://cs6.salemstate.edu/~byi/

Section	Time	Room	Final Exam
L21	W/F 1:40-2:55	ZOOM Meeting ID: **952 3021 3714** Passcode: **163037** https://salemstate.zoom.us/j/95230213714 SSU Course Online at: https://elearning.salemstate.edu	**Dec 21, Tuesday** **2:00pm—4:00pm** **Zoom/Online** **(Zoom/Canvas)**
01	W/F 3:05-4:20		
Office Hours: T/R: 9:25—10:40am W/F: 12:15—1:30pm			

Catalog description:

This course extends the treatment of object-oriented methodologies, languages and tools begun in CSC110. The emphasis is on the analysis of complex problems, particularly those involving multiple design alternatives, and the use of class libraries. Fundamental strategies for algorithm design are presented and discussed. Specific topics include inheritance, polymorphism, recursion, stream and file I/O, exceptions, and graphical interface programming. Style, documentation, solution robustness, and conformance with specifications are emphasized throughout. Three lecture hours and three hours of scheduled laboratory per week, plus extensive programming work outside of class

Prerequisite: CSC110 or 201J.

Goals:

The purpose of this course is to enhance and extend students' understanding of tools and techniques for object-oriented software development. Upon completion of the course, a student should be able to do the following:

CG01: analyze a problem statement for completeness and clarity;
CG02: use the methodology of object-oriented design to develop class diagrams (data descriptions and methods) for a problem solution;
CG03: demonstrate understanding of and apply fundamental strategies for algorithm design;
CG04: convert this solution into source code in the designated high-level programming language in accordance with a well-defined set of style rules;
CG05: debug and test the program;
CG06: provide clear documentation for the result.

Objectives:

By the end of the course students will have:
CO01: gained a deeper understanding of object-oriented design methodology;
CO02: learned to recognize situations in which multiple design alternatives are possible;

CO03: applied fundamental algorithm design strategies;
CO04: learned to recognize and apply design patterns;
CO05: learned and utilized techniques for validation and verification of programs;
CO06: gained experience in judging the effectiveness and cost of a software design;
CO07: gained experience in choosing among competing design alternatives;
CO08: gained experience in the use of the UML modeling language;
CO09: extended their knowledge of an object-oriented programming language, including graphical user interfaces, event-driven programs, file-based input/output, and the use of libraries;
CO10: produced full documentation for multiple completed projects, including formal class diagrams;
CO11: participated in one or more group projects.

Course Topics:

A detailed topics list and a general course bibliography can be found on the Computer Science Department website at https://cs.salemstate.edu/courses/course-information. Select CSC 115 to access a PDF document.

Text:

(Required) Java How to Program: Early Objects, 11th Edition, by Deitel & Deitel. Prentice-Hall, 2017 (ISBN: 978-0-13-474335-6).

Required Material:

(Required) Thumb (flash) drive, 16 GB minimum or online storage (for saving your projects and coursework) in addition to your personal computer/laptop.

Additional references:

- Course teaching materials: http://cs6.salemstate.edu/~byi/CSC201J_202J_Deitels/index.html.
  - This website is password-protected and ask the instructor of the password (or log into Canvas for it).
- Course online system (Canvas): https://elearning.salemstate.edu/.
  - Access to this site via the username and password given/assigned by SSU.

Software:

(Required) Java SE 8 (or above. Version 8 or 11 is preferred). Download the JDK (i.e., Java Development Kit) at https://www.oracle.com/java/technologies/javase-downloads.html. You may use your preferred IDE for working on Java programming assignments.

Cell phones:

Turn the ringer off, or, better yet, *turn the phone off*.

Lecture/Lab Attendance:

Class policy is that of the Registrar's office - see the University catalog for details. Lecture will start promptly at the scheduled time, so please make a serious effort to not be late. While class attendance does not *directly* affect your final grade, some of the material covered in class is not found (in the same form) in the text, so class attendance and notes are very important. Note that you are at all times responsible for materials and assignments discussed in class. We will use SSU's online course management system, Canvas (https://elearning.salemstate.edu/) to post all types of assignments (including different formats/types of tests), grades, and announcements regarding the course topics and progress. You will need to visit Canvas (with your SSU Navigator use-name and password) for the course activities. Canvas uses your ***SSU-stored email*** box for the communication between you and the instructor and thus you ***must use this email*** address. Each student is responsible for completing all course requirements and for keeping up with all that goes on in the course (whether or not the student is present).

Class attendance is **strongly recommended**. **Lab exercises will be tested and graded in the lab hours and no late submissions are accepted**. **Some tests will be held in the lab hours.**

Some of the class/lab hours will be used to review important course topics, to discuss and investigate Java implementation details that time may not permit to be fully presented in the texts (for design and implementation drills, for programming exercises) to assist with design and implementation problems that arise in project exercises, to discuss and review important topics/questions regarding the tests, and to check/examine/grade the exercises and homework.

Student-Instructor Communication:

If you have any questions regarding course material, and *in particular if you are having problems with a programming assignment*, the most effective way to get assistance is to *discuss with the instructor* (*either in the class, in the lab, or outside the classroom*).

Please **note**: Canvas is used for submission of the assignments/labs/projects and for posting grades. If you ask questions through Canvas-grading-submission features, they usually do ***not*** go to the instructor directly. Please ask questions in the class/lab or send emails (you may send email via Canvas) to the instructor!

Lab Exercises:

Programming practices and exercises regarding Java programming topics covered in the lecture are initialed and in the lab hours and lab exercises are designed not only for help on the understanding of the course topics but also for the preparation of programming projects. Lab exercises must be completed**, tested and graded** (by the instructor) **during the lab hours**. Lab submissions to Canvas only (by the deadline) do ***not*** guarantee lab credits—you need to have them tested by the instructor in the lab hours by the deadline.

Lab will be used to discuss and investigate Java implementation details that time may not permit to be fully explored during the scheduled lecture period, for design and implementation drills, to assist with design and debugging problems that arise in longer lab / project exercises, and to check/examine/grade the exercises and homework.

Please note that lab exercises are different assignments from the programming projects and the credits earned from the programming projects do not count for your lab assignment grades.

Some tests will be held in the lab hours.

The following table provides the lab activities (subject to minor changes as we go through the coursework).

Week	Dates	Contents (textbook chapters and others)
1	9/2—9/3	Programming Basics (Ch 1, 2, 3, 4)
2	9/6—9/10	Design & Implementation Basics (Ch 5, 6, 7)
3	9/13—9/17	Design & Implementation Basics (Ch. 8) OOP: Inheritance (Ch 9)
4	9/20—9/24	OOP: Inheritance (Ch 9)
5	9/27—10/1	OOP: Polymorphism & Interface (Ch 10)
6	10/4—10/8	OOP: Polymorphism & Interface (Ch 10)
7	10/11—10/15	Exceptions & Exception Handling (Ch 11 & others)
8	10/18—10/22	Java GUI (Ch 2, 3, …, 11)
9	10/25—10/29	Java GUI (Ch 2, 3, …, 11) Strings, Characters (Ch14)
10	11/1—11/5	Files, I/O Streams (Ch15)
11	11/8—11/12	Generic Collections (Ch16) Lambdas and Streams (Ch17)
12	11/15—11/19	Recursion (Ch18) Searching, Sorting (Ch19)
13	11/22—11/26	Searching, Sorting
14	11/29—12/3	Generic classes/Methods: A Deeper Look (Ch20) Custom Generic Data Structures (Ch21)
15	12/6—12/10	Custom Generic Data Structures (Ch21) Applets & JUnit

Lab Grading Procedures:

1. Submit your solution (Java source code or a compressed/zipped folder/directory which contains all the source code programs) for the lab assignment to Canvas by the deadline;
2. Show/test your solution in the lab hours;
3. Have the instructor test your code in the lab hours;
4. Answer the questions provided by the instructor correctly during the testing.

Final Grade:

Final grade will be determined using the following grading weights:

written assignments	8%
lab exercises	22%
programming projects	30%
mini-tests/quizzes	20%
final examination	20%

Attendance is not used to calculate the final grade: however, note that you are at all times responsible for all types of assignments and materials presented in class.

The following table shows how the course work is assessed against the course objectives:

	Tests	Assignments (including programming projects)	Lab Exercises
CO01	✓	✓	✓
CO02	✓	✓	✓
CO03	✓	✓	
CO04	✓	✓	✓
CO05	✓	✓	✓
CO06	✓	✓	✓
CO07		✓	✓
CO08	✓	✓	✓
CO09		✓	
CO10		✓	
CO11		✓	

Tests (mini-tests/quizzes and final examination):

There will be mini-tests/quizzes administered throughout the semester and one *comprehensive* final examination. Please refer to Final Grade above for the grading weights of these tests.

The mini-tests will be like closed-book written quizzes (with questions in the form of multiple-choice, true-false, fill-in-blank, and short-answer questions) and programming problems. The tests must be completed during the lab hours (or other designated time periods). These programming mini-tests will be much like lab exercises and/or programming projects on a smaller scale.

The final examination will be a comprehensive examination, i.e., covering all the topics given in the class with emphasis on assignments (reading/writing) and programming skills/algorithms.

Missed Tests:

Tests (mini-tests/quizzes/final-exam) may not be made up except for *documented/emergency* situations. If a test must be made up, arrangements must be made with the instructor to take the test before it is discussed in class (usually within a week of the test being administered).

Due Dates/Time:

- Late submission of assessed assignments or projects will result in penalties defined in the following:
  - **25 percentage** points will be deducted for being late, up to 24 hours.
  - **50 percentage** points will be deducted for being late, more than 24 hours and up to 48 hours.
  - **75 percentage** points will be deducted for being late, more than 48 and up to 72 hours.
  - **100 percentage** points will be deducted for being late more than 72 hours (i.e., a grade of zero will be given).
  - A grade of **zero** will be given for any assignments including projects which are submitted after the final examination time. That is: **no assignments (including semester project) will be accepted after the final examination**.
- (Refer to course syllabus for more details on urgent/exceptional situations for extension on labs.)

Study Groups:

While I strongly encourage study groups, I require that each student hand in his/her answers in her/his own words - if two answers are highly similar to each other, neither will receive credit.

When working on your programming projects, you may discuss with others the project topics, the algorithms and methodologies related to the project; but when you work on writing the code, this coding work should be 100% of your own work. **If two answers/written code segments come out exactly the same or highly similar, neither will receive credit and/or further actions will be taken** (such as reporting to the department and/or university).

Academic Integrity:

Academic Integrity Policy and Regulations can be found in the University Catalog and on the University's website (http://catalog.salemstate.edu/content.php?catoid=13&navoid=1295#Academic_Integrity). The formal regulations are extensive and detailed - familiarize yourself with them if you have not previously done so. A concise summary of and direct quote from the regulations: "Materials (written or otherwise) submitted to fulfill academic requirements must represent a student's own efforts". *Submission of other's work as one's own without proper attribution is in direct violation of the University's Policy* and will be dealt with according to the University's formal Procedures.

All students are expected to be familiar with the academic regulations, including those regarding Academic Integrity, for Salem State University as published in the college catalog. In addition, each student is responsible for completing all course requirements and for keeping up with all that goes on in the course (whether or not the student is present).

Salem State University is committed to providing equal access to the educational experience for all students in compliance with Section 504 of The Rehabilitation Act and The Americans with Disabilities Act and to providing all reasonable academic accommodations, aids and adjustments. Any student who has a documented disability requiring an accommodation, aid or adjustment should speak with the instructor immediately. Students with Disabilities who have not previously done so should provide documentation to and schedule an appointment with Disability Services and obtain appropriate services.

Students must comply with the university's Covid-19 health and safety protocols for the 2020–2021 and 2021–2022 academic years. These protocols include wearing masks in class and on campus in public spaces, practicing physical distancing where possible, including in class, engaging in a daily symptom check, notifying counseling and health services at 978.542.6413 if they have any symptoms associated with COVID-19, and not coming to campus or to an in-person class if they have any of the symptoms related to COVID-19 until cleared by the student life wellness area. Students who have documented disabilities that may prevent them from complying with these policies are required to contact the disability services office.

In the event of a university declared critical emergency, Salem State University reserves the right to alter this course plan. Students should refer to Salem State for further information and updates. The course attendance policy stays in effect until there is a university declared critical emergency. In the event of an emergency, please refer to the alternative educational plans for this course located at Canvas (https://elearning.salemstate.edu/) where the course contents such as announcements, assignments are posted. Students should review the plans and gather all required materials before an emergency is declared.

Please remember that if, for any reason, you decide to drop this course, you **MUST** do so officially through the Registrar's office. The last day to withdraw from a course this semester is **November 19th**.

Note: This syllabus represents the intended structure of the course for the semester. If changes are necessary, students will be notified in writing and via all regular class communication mechanisms (class discussion, emails, and/or the course link at Canvas https://elearning.salemstate.edu).