Salem STATE | UNIVERSITY

Computer Science Department
cs.salemstate.edu

SYLLABUS **Spring, 2021**

CSC 260 Data Structures and Algorithms **4 cr.**
Prerequisites: CSC105 and CSC115

Instructor: Beifang Yi	**Office**: MH 211A/Zoom	**Phone**: (978) 542-7246
email: byi@salemstate.edu	**Hours**: TWRF (10:00-10:50) TWR(1:30-2:00)	**Website**: http://cs6.salemstate.edu/~byi/

Section	Time	Room	Final Exam
01	W/F 10:50-12:05	ZOOM Meeting ID: **952 3021 3714** Passcode: **163037** https://salemstate.zoom.us/j/95230213714 One tap mobile: +19292056099,,95230213714#,,,,*163037# US (New York) +13017158592,,95230213714#,,,,*163037# US (Washington D.C)	**Thursday 5/13 8:00am-10:00am Online (Zoom/Canvas)**
L21	W/F 12:15-1:30		
Office Hours: TWRF: 10:00-10:50 TWR: 1:30-2:00			

Catalog description:
Basic data structures such as stacks, queues, linked lists, and trees are studied and applied to problems in data storage and manipulation. Applications include basic searching and sorting algorithms. Fundamental strategies for algorithm design are reviewed and extended. Design, analysis, implementation, and quality assurance techniques are discussed. Three lecture hours and three hours of scheduled laboratory per week, plus extensive programming work outside of class.
Prerequisites: CSC105 and CSC115.

Goals:
The purpose of this course is to develop students' knowledge and appreciation of organization and retrieval techniques and to familiarize students with the basic concepts of order-of-magnitude analysis. The goals of this course are:

CG01:	to develop an appreciation for the process of data abstraction and its usefulness in software development;
CG02:	to develop the skills and knowledge necessary to perform design and basic analysis of algorithms;
CG03:	to present a selection of the most common data structures and their standard implementations and uses;
CG04:	to present a selection of the most common algorithms for searching and sorting.

Objectives:
Upon successful completion of the course, student will have:

CO01:	applied data abstraction techniques;
CO02:	implemented several classic data structures "from scratch";
CO03:	demonstrated knowledge and use of ADTs available in one or more language libraries;
CO04:	recognized the factors required to perform algorithm design, analysis of algorithms and performed order-of-magnitude analysis;
CO05:	chosen, with justification, an appropriate structure to match the requirements of a given problem, implemented the structure if necessary, and used it in an appropriate way to solve the problem;

CO06: utilized standard techniques for program validation;
CO07: demonstrated the ability to use the UML modeling language;
CO08: produced documentation for at least one major completed project, including formal class diagrams and rigorous test set specification and results;
CO09: participated in at least one group project involving problem analysis and design specification and selection;
CO10: demonstrated recognition of the need for future professional development through research into future trends in the areas of analysis of algorithms and application development and profiling.

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 and select "CSC 260 Data Structures and Algorithms" to access a PDF document.

Text:

(Required) Data Structures and Algorithms in Java, 6nd Edition, by Michael Goodrich, Roberto Tamassia, and Michael Goldwasser, Wiley, 2014 (ISBN: 978-1-118-77133-4).

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 website: http://cs6.salemstate.edu/~byi/CSC260/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.
- *Java Software Structures: Designing and Using Data Structures*, 4th Edition, by Lewis and Chase. Pearson, 2014
- *Data Structures and Algorithms in Java*, 2nd Edition, by Robert Lafore. SAMS, 2003
- *Data Structures and the Java Collections Framework*, 3rd Edition, by William J. Collins. Wiley, 2011

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.

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:

Data Structures and Algorithm Analysis is one of the fundamental subjects in computer science and also one of the core requirements in software design and development. Learning how to develop software is very much a **hands-on, experiential process** - the only way to be sure that you understand the material is to apply it by designing and writing programs.

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!

Final Grade:

Final grade will be determined using the following grading weights:

assignments (reading, writing)	15%
lab exercises	15%
programming projects	25%
Mini-tests	20%
final examination	25%

Please note: This syllabus (lab sessions) only focuses on the lab activities section (that is, CSC260-L21) of its corresponding class lecture section (that is, CSC260-01). Please refer to the CSC260-01 syllabus for the specifications and requirements of the other coursework listed in the above table. Also **notice** that although the grading weight of the lab exercises is indicated as only 12% of the whole course grade, the lab coursework does occur, in some format and/or to varying degrees, in other portions of coursework, such as Mini-tests, the final examination, and programming projects.

Lab Exercises:

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

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

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

Week	Dates	Lab Activities
1	1/21—1/22	Java Primer (Ch 1)
2	1/25—1/29	OOD (Ch2)
3	2/1—2/5	Fundamental Data Structures (Ch3)
4	2/8—2/12	Fundamental Data Structures (Ch3)
5	2/15—2/19	Algorithm Analysis (Ch 4)
6	2/22—2/26	Recursion (Ch 5)
7	3/1—3/5	Stacks/Queues/Deques (Ch 6)
8	3/8—3/12	List & Iterator ADT (Ch 7)
9	3/15—3/19	**Spring Recess**
10	3/22—3/26	Trees (Ch 8)
11	3/29—4/2	Priority Queues (Ch 9)
12	4/5—4/9	Maps/Hash/Set (Ch 10)

13	4/12—4/16	Search Trees (Ch 11)
14	4/19—4/23	Sorting/Selection (Ch12)
15	4/26—4/30	Text Processing (Ch 13)
16	5/3—5/5	Graph (Ch 14)

Lab (Programming) Submission Requirements

1. You need to create a folder/directory for the lab with the folder name like "**LABnn_YourLastName**" as the folder/directory name.
    a. nn—2 digits: the lab number.
    b. YourLastName—your last name (no spaces).
    c. Examples: "LAB02_Yi" (the 2nd lab for Yi).
2. Copy Java source code files (i.e., *.java files) and necessary data files for your lab solution to the folder.
    a. You may have to do some necessary changes on the class files dependent on the IDE you are using such as commenting out "project xxxx" line in your code.
3. Your Java class naming requirements:
    a. Use meaningful class names which follow Java Code Convention.
    b. For each lab question, the main class (that which contains "main()" method) must have the name like "**LABnn_TestClass.java**".
4. Compress "LABnn_YourLastName" folder (and its contents) to a **SINGLE zipped** file (.rar, .zip) and submit this file to Canvas.
5. **Important**:
    a. You need to apply the above naming requirements at the ***beginning*** when you work on your lab—that is, do ***not*** first complete your lab and then change the Java class names!
    b. If you do ***not*** follow this naming convention, you may lose up to **100%** of lab credit points.
    c. Double-check your solution before submission:
        i. Extract that zipped/compressed file to a temporary folder.
        ii. Enter the folder (or its subfolders) and under command/terminal compile your Java files with command "**javac *.java**" or "**javac LABnn_TestClass.java**".
        iii. Test for correct results with command with command "**java LABnn_TestClass**".
        iv. If you can successfully compile and test those programs in the above steps, you may submit that zipped folder for your solution.

Lab Grading Procedures:

1. Submit your solution—a compressed/zipped folder/directory which contains all the source code programs for the 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;

Tests (mini-tests and final examination):

There will be several mini-tests 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 may not be made up except for *documented/emergency* situations. If a test must be made up, arrangements must be made with the instructor, usually within a week of the test being administered.

Due Dates/Time:

- Late submission of assessed coursework 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/labs/projects which are submitted after the final examination time. That is: **no assignments/labs/projects will be accepted after the final examination**.
- Should there be an emergency that prevents you from completing/submitting your assignments/projects on time, you will need to send *an email request* for the extension on the coursework submission. The instructor will reply to this request email with a specific number of days for the extension period or a new deadline for the assignment and you will need to keep *this email as a record of the extension approval*. Sending only a request email does ***not*** guarantee the extension approval.
- **Please double-check** your submissions (since your assignment submissions are usually graded after their deadlines): to guarantee your successful submissions, you would need to download your submissions and examine the downloaded materials; as for the projects, you would need to check the downloaded programs and then compile/run the code.

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 the Office for Students with Disabilities and obtain appropriate services.

In the event of a University declared critical emergency, Salem State University reserves the right to alter this course plan. Students should refer to salemstate.edu 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/). 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 **April 16th**.

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).