COMP 110/L: Intro. to Algorithms and Programming

Fall 2020

Instructor: Mahdi Ebrahimi (mahdi.ebrahimi@csun.edu)

Office Hours: M\W 2:00 PM - 3:00 PM (by appointment – online via Zoom)

Class Details:

Section	Class Title	Days & Times	Room	Class Dates
COMP 110-20 (17150)	INTRO ALGRTH/PROG (Lecture)	MoWe 9:30AM - 10:45AM	Online via Zoom	Aug 24, 2020- Dec 8, 2020
COMP 110L-20 (17152)	ALGRTH/PROG LAB (Laboratory)	MoWe 11:00AM - 12:15PM	Online via Zoom	Aug 24, 2020- Dec 8, 2020
COMP 110L-21 (17153)	ALGRTH/PROG LAB (Laboratory)	MoWe 12:30PM - 1:45PM	Online via Zoom	Aug 24, 2020- Dec 8, 2020

Final Exam:

Section	Class Title	Exam Date	Exam Time	Exam Room
COMP 110-20 (17150)	INTRO ALGRTH/PROG (Lecture)	12/09/2020, Wednesday	08:00AM - 10:00AM	Online via Zoom and Canvas

Zoom Lecture Policy:

In the event the course is held online, students will be required to appear on **webcam** during lectures and exams. For students who feel this is a violation of their privacy, they can take steps to create a neutral background, either physically or electronically. For any student who still objects, the instructor may be able to grant exceptions if the student provides the instructor with a reasonable justification for why the policy is not acceptable. Requests for exceptions will be considered on a per student basis.

Communication:

Please use email rather than telephone voice mail for messages. Please keep emails short and focused, and use a clear subject line beginning with “COMP 110 Question”. Many technical questions are better handled in person during lecture and lab rather than email, since the class as a whole might benefit from the discussion.

You may email me (mahdi.ebrahimi@csun.edu) at any time; I will generally respond within 24 hours (during the academic days). Always include your name, course, and CSUN e-mail address in your messages to me; an email address like meqwik@love.com leaves me clueless about who you are**!**

Course Description (from the catalog):

Introduction to algorithms, their representation, design, structuring, analysis, and optimization. Implementation of algorithms as structured programs in a high-level language.

Goals:
This course teaches basic skills in analyzing problems and solving them by finding or creating an appropriate algorithm (a recipe for solving a problem). Once an algorithm has been identified, it is translated into an executable application using the Java programming language. This last task is often called "programming," although this step is only one among many important steps in the software development lifecycle. The principles and skills related to problem solving and algorithms are general and are not specific to any programming language.

The CSUN CS Department has made a policy decision to use the Java language for lower division courses such as COMP 110, so Java language details will be presented as an important part of both lecture and lab, although the policy may change in the future. For now, you must demonstrate an appropriate level of both problem solving and Java programming skill to successfully complete the course. The lecture component (COMP 110) focuses on concepts and practical examples. The lab (COMP 110/L) focuses on developing working and correct applications using Java and problem-solving techniques covered in lecture.

This course does not assume any previous experience in computer programming and material begins at an introductory level. However, coverage is fast-paced and moves on to more advanced topics quickly. This is not a survey course for non-majors, it is a real programming course designed for students concentrating in Computer Science and related majors who need to quickly develop
real programming skills.

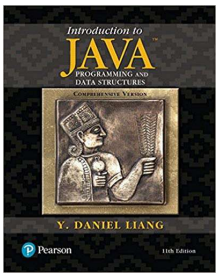
Course Objectives: A successful student will be able to:
1. Demonstrate the knowledge of a computer and operating system. The editing and compilation process.
2. Translate human-readable algorithms represented by pseudo code, flowchart or flow block diagram to Java.
3. Write and test Java programs using the 4 fundamentals of programming: sequence, choice, loop, and methods.
4. Construct programs that require several methods and good knowledge of passing parameters.
5. Demonstrate the knowledge of basic steps of software development: problem statement, program development, testing and documentation.
6. Solve problems with one- and two-dimensional arrays.
7. Use basic sorting and searching methods.
8. Apply the class String.
9. Read and write text files. Demonstrate practical using of Exception Handling.
10. Recognize the role of Object-Oriented Programming in software development.
11. Run the examples and exercises studied in the course.
12. Understand the ideas of Polymorphism and Inheritance.

Prerequisites: *Grade of "C" or better in MATH 102, 103, 104, 105, 150A or 255A, or a passing score on the Math Placement Test (MPT) that satisfies prerequisites for MATH 150A or 255A.*
***Corequisite**: COMP 110L.* Introduction to algorithms, their representation, design, structuring, analysis and optimization. Implementation of algorithms as structured programs in a high-level language. Lab: 3 hours per week. (Available for General Education, Lifelong Learning if required by student's major.)

Course Material:
Course material is available on Canvas (https://canvas.csun.edu)
Grades will be posted on canvas (https://canvas.csun.edu)

Any questions about a Lab/Exam grade should be addressed within two days of posting. After two days, all grades are final.

Textbook:



No textbook is required. If you'd like a textbook for further study, a good supplemental textbook is:
"Introduction to Java Programming and Data Structures", Comprehensive Version, 11th Edition - ISBN-13: 978-0-13-467094-2
Any edition from the past several years will suffice.

Software Tools:

This class requires you to have direct and continual access to a computer. In short, it is a de facto requirement to own your own laptop computer to be successful in your chosen field.

You are free to do your work on your personal laptop. Lab machines use the Linux OS, but the software applications required for class work are available for all common OSes. You must install the Java Development Kit (JDK) and an editor to write programs for the class. You can use whatever editor you prefer. If you don't have a preference, then I suggest JGrasp. It is important that you install the JDK first, then the JGrasp editor. Make sure you install the JDK, not the JRE (Java Runtime Environment).

For the JDK, go to:

http://www.oracle.com/technetwork/java/javase/downloads/index.html
Download the Java SE (Standard Edition) JDK, the latest version, SE 8.
Install JDK first, before proceeding to download and install JGrasp.

For JGrasp, go to:

http://www.jgrasp.org
Click on the Downloads link and download the installation for your platform (Windows, MacOS, or Linux/Unix).

This software is available in JD1104 and in the open computer science labs.

Grading:

You will receive a **single combined grade** for the lecture and lab. Your grade is based on the following components:

Lab Assignments	25%
Lab Midterm Exam #1	10%
Lab Midterm Exam #2	15%
Handwritten Midterm Exam	15%
Lab Final Exam	15%
Handwritten Final Exam	20%

Lab assignments will be frequent, typically with two due per week. The exact number of lab assignments has not been set, as this will depend somewhat on how the class progresses. These are low-stakes assignments which are intended to introduce an entirely new concept for the first time. Lab assignments are submitted through Canvas (https://canvas.csun.edu/). In the event that there is a problem with

Canvas, you may email your assignment to me (mahdi.ebrahimi@csun.edu), though this should be considered a last resort.

Grading Policy: Exam questions will relate to the contents of both the textbooks **and** *material discussed in class*. To do well, you should attend class regularly, participate in discussions, do all assignments, and take notes. If you miss a class, please arrange with someone to get notes and go over the important points with them. Total category grade is calculated based on the sum of all grades for the category.

Plus/minus grading is used, according to the scale below. The left column shows the minimal score necessary to receive the grade in the right column. The highest letter grade possible given the score is chosen; e.g., if you receive an 88.2, you'd receive a 'B+' for the course, which corresponds to being >= 86.5.

If your score is >=...	...you will receive...
92.5	A
89.5	A-
86.5	B+
82.5	B
79.5	B-
76.5	C+
72.5	C
69.5	C-
66.5	D+
62.5	D
59.5	D-
0	F

- **NOTE: Failure to take the Final Exam will result in a grade of "WU" which is equivalent to a grade of "F"**
- An important part of this course is the notation, terminology, concepts, and definitions; therefore, I do not answer questions during examinations.
- In fairness to all, I don't give make-up for any missed projects, quizzes, or exams.
- An incomplete (I) grade is given for genuine medical and other certified emergencies only; it is never given to catch up with missed assignments. Furthermore, to receive an Incomplete grade, you must have successfully completed at least two-thirds of the semester with a passing grade.

Plagiarism and Academic Honesty:
Plagiarism in any assignment or cheating in the examinations will result in a grade of F in the entire course.
On an exam, you are expected to submit only your own work. On a programming project, it is permissible to discuss solution approaches in a general sense with other students. But when submitting a program for a grade, the program must represent your own work. It cannot be a copy of another student's program, even if you worked in a group with that student. **Penalties for academic dishonesty on a single exam or programming project may result in a grade of "F" for the entire course.** A report will also be made

to the Office of the Vice President for Student Affairs. Students who repeatedly violate this policy across multiple courses may be suspended or even expelled.
If you have any doubts about what is considered dishonest, please ask the instructors for guidance before taking such a serious risk. In general, **full disclosure is the best policy** on any submission. In other words, if a friend helped you to complete a project, **state this fact in writing at the beginning of the submission.** Such submission may not earn full points.

Makeup or Extra Credit:
No makeup assignments are given to compensate for poor performance in regularly assigned work. The pressure of work, academic workload from other classes, and schedule extracurricular activities during the semester (e.g., getting married, travel) are unacceptable excuses for missing classes or not submitting the assignments by the due dates. You get no credit by telling me that you already know the stuff; the only way to earn points is by completing the coursework on time. If you know the course material, then please drop this class and enroll in a class that is useful.

Late Policy / Exam Scheduling:

Late assignments will be accepted without penalty if prior arrangements have been made or there is some sort of legitimate emergency (at my discretion). If you must be absent from an exam, contact me ASAP to see if alternative accommodations can be made. Note that all exams have been scheduled ahead of time (see the Class Schedule and List of Topics).
If an assignment is otherwise submitted late, it will be penalized according to the following scale:

If your assignment is late by <= this many days...	...it will be deducted by...
1	10%
2	30%
3	60%
4+	100%

To be clear, assignments which are submitted four or more days beyond the deadline will not receive credit.

Submitting Work:
1. Most assignments will be submitted electronically via Canvas
2. Please do not email submissions.

Class Schedule and List of Topics (subject to change)

Exactly which topics are covered and when is subject to change.

Wk	Dates	Topics	Notes (exam dates subject to change)
1	08/24- 08/26	Compiling and running Java code	
2	08/31 - 09/02	Reading user input, arithmetic operations	
3	09/07 - 09/09	Methods and tests	**09/07 Labor Day Holiday** **No class**
4	09/14 - 09/16	Introduction to objects, instance variables, visibility modifiers	**09/16 Lab Midterm Exam #1**
5	09/21 - 09/23	Overloading, basic conditionals, random numbers, floating point	
6	09/28 - 09/30	Complex conditionals, `switch`, more random numbers	
7	10/05 - 10/07	Command-line arguments, arrays, introduction to loops	
8	10/12 - 10/14	More loops and arrays	
9	10/19 - 10/21	Complex loops	
10	10/26 - 10/28	Array of objects - Loops and array practice	
11	11/02 – 11/04	`String`, multidimensional array, Review	
12	11/09 - 11/11		**11/09 Lab Midterm Exam #2,** **11/11 Handwritten Midterm Exam** (online via Canvas in class/lab time)
13	11/16 - 11/18	Inheritance, `super`, polymorphism	
14	11/23 - 11/25	Interfaces, exceptions	
15	11/30 - 12/02	File I/O, `finally` Exam Review	**12/02 Lab Final Exam**
16	12/07 - 12/09	Final Exam Week	**12/07 - Last Day of Class** **Handwritten Final Exam:** **Wed 12/09, 09:30 AM – 11:30 AM** **Online via Zoom and Canvas**