CSCI 221: Computer Programming II
Fall 2018

Professor:	Dr. Renée McCauley, mccauleyr@cofc.edu
Office & hours:	HWE 317, **TR** 11:15-12:15, **W & F**, 1:30-3:30, by appointment and any time my office door is open.
TA	Hassam Solano-Morel, HWE 311, MW 10am-12 noon, and by appointment. Can answer course-specific assignment questions and general Java programming questions. Luis Mejia, HWE 311 – can answer general Java programming questions. Hours TBA.

Course catalog description: This course further develops object-oriented programming introduced in CSCI 220. Topics include file input/output, inheritance and polymorphism, exceptions, error handling and algorithm analysis.
Course Pre-requisite: CSCI 220 & 220L – grade of C- or better.
Course Pre- or Co-requisite: Discrete structures I, Math 207

Required Text is an online, interactive textbook that you buy access to:

1. Sign in or create an account at learn.zybooks.com
2. Enter zyBook code: COFCCSCI221McCauleyFall2018
3. Subscribe. (A subscription is $58 and will last until late December 25, 2018.)

Personal access to the text is essential, as completion of the interactive components will contribute to your course grade.

Recommended Interactive Development Environment (IDE): BlueJ <BlueJ.org>
BlueJ was designed for use by students learning Java, and I will use it in all code demonstrations. I highly recommend it. Other environments are not as easy to use, and I'd prefer that you focus on Java, not figuring out or maybe fighting a more complex IDE. Note: You may use any environment you wish, but will have to be very careful to remove "package" headers and pull files out of "src" folders before submitting. (I will demo using BlueJ.)

Communication/participation:
We will use **piazza.com** for out-of-class discussions. Piazza caters to getting you help fast and efficiently from classmates, the TA, and the instructor. Rather than emailing questions to the teaching staff, I encourage you to post your questions on the class Piazza page, and anyone in the class can answer it. You can post anonymously to the class if you wish. *You should have already received an invitation to join the piazza class.*

> **NOTE:** when you join piazza, piazza will ask you if you wish to be a part of Piazza Careers. If you say yes, they will share/sell your profile information with companies seeking to recruit you (, but possibly seeking to sell stuff to you). This is an "opt in" service. I suggest that you **opt out**, because we don't really know what they do with your data. **Opt out by choosing the "I don't need any help getting the most fulfilling and rewarding career opportunities at this time."**

Graded Events:

- **Assignments** (25% of final grade)**:**
  - **zyBooks** participation and challenge exercises (5% of final grade), associated with each chapter covered.
  - **Labs/Inclass activities** (5% of final grade) are short programming or other exercises which may be done in class or may be done outside of class. In class labs must be completed in class on the day assigned, and will include paper-based and programming problems. These assignments will usually be handed in on paper, but could require OAKS upload.
  - **Programming assignments** (15% of final grade) are larger programming assignments, these solutions will be uploaded to OAKS.

- **Quizzes** (15% of final grade) – there will be a short quiz every Tuesday, except the first day of class, and during weeks where there is a test scheduled.

- **2 Tests** (35% of final grade) in-class, paper-based. See piazza for tentative dates.

- **Comprehensive final exam** (25% of final grade) primarily paper-based, but may include a programming component. Exam dates and times are determined by the college. See final exam schedule for dates/times.

- **Grading scale and grade computation:**
  - All components of the course (tests, exam, labs, assignments, etc) are vital to success and passing the course. Passing is considered a grade of C- which requires an overall class average of 70% or higher. To pass the course, you must earn a 70% on the Assignments portion of the course.
  - Grading Scale: A: 90-100; B: 80-89; C: 70-79; D: 65-69; F: < 65. Plus and minus grades may be given at the discretion of the instructor.

- **Programming assignments:**
  - Programming assignments will be posted on piazza and the calendar will show when assignments are made and when they are due.
  - Assignments are **due at the date/time specified when the assignment is made*, but you will be given 24 hours leniency to get it completed and uploaded***. No extensions / no exceptions. Please don't ask.
  - If a student submits all of her/his completed programming assignments by the due date/time, not using leniency, a student will receive 2 bonus points toward the overall assignment grade at the end of the semester.
  - Programming assignments will be submitted through OAKS or as specified on the assignment. Details of what each submission requires will be specified on the assignment. Assignments submitted other ways **WILL NOT BE GRADED.**
  - Programming assignments must be completed individually, unless a policy that allows collaboration is specified on the assignment write-up. If you violate the collaboration policy, you will be referred to the CofC Honor Board. You are encouraged to read the complete Honor Code and all related processes in the *Student Handbook* at http://studentaffairs.cofc.edu/honor-system/studenthandbook/index.php
  - Our course TA will be available to guide you with your assignments, as will I. If you choose to seek help from the tutors at the Student Learning Center, be aware that they should never give you code, or even touch your computer. You should never include in your solutions any technique not covered in class or that you don't understand fully.
  - **No extra assignments will be given to make up for work missed during the term!**

Course policies:

1. This is an assignment-intensive class. Expect to spend a minimum of 10 hours per week working outside of class on the work for this class. If you start an assignment the day before the assignment is due, expect to miss the deadline.
2. I want to answer your questions – post them to piazza or visit my office during my office hours.
3. In class participation is essential to success – come to class prepared and feel free to ask questions. Outside of class, post questions on piazza.
4. You are expected to attend every class meeting. You are expected to arrive on time and stay until the end of class.
5. Excused absences are accommodated. Documented sickness is excused – talk to the instructor if you miss due to sickness, even if you did not see a doctor.
6. You are responsible for material, information and deadlines, whether or not you attend and whether or not your absence is excused. **Please check piazza, follow-up with classmates and the instructor if you must miss a class.**
7. You are **NOT ALLOWED TO USE a computer in class**, **except on lab days**.

Learning Outcomes	
1.	Apply the software development process in problem solving. Specifically, focus on coding to a specification this includes contracts and typing (e.g., pre- and post-conditions), using pseudo-code to draft an algorithm before it becomes code, test driven development, range and boundary checking, and unit testing.
2.	Learn Java syntax. This includes constants and variables, assignment, arithmetic operations, relational operators, logical operators (including short–circuit), selection statements, repetition statements, and data types (i.e. reference, primitive) and primitive wrapper classes.
3.	Write programs that use IO operations. This includes reading/writing text from standard IO, reading/writing structured data (e.g. csv or imaging data) from a file.
4.	Write programs that store and manipulate data using one- and two-dimensional arrays. This includes iteration and common tasks such as finding max/min, summing, making copies (shallow and deep copy operations) and other aggregate operations.
5.	Design classes and apply them in program development. This includes object instantiation, methods (e.g. constructors, getters, setters, helpers), overloading and overriding (e.g toString, equals), static versus non–static methods/variables, access qualifiers (public, protected, private), and "this" reference.
6.	Write programs that incorporate object-oriented design principles. This includes inheritance (concrete classes, abstract classes and interfaces, is-a relationships), composition (has-a relationship) and class hierarchies. Identify scenarios in which inheritance or composition is appropriate
7.	Apply exceptions in program development. This includes defining exception classes using inheritance, 'throws' and 'try/catch' syntax, and understanding the difference between checked and un-check exceptions.
8.	Analyze recursive algorithms (such as factorial, fibonacci, list length, and binary search) and explain/trace using a stack data structure.
9.	Design, implement, and test a dynamic array-based collection (such as an ArrayList, OrderedList, PriorityQueue,). This includes implementing an interface, managing both size and capacity, and traversal via an iterator.
10.	Code a multi-class project/application that applies the object-oriented design principles covered in this class. (Multi-class is defined as a minimum of 4 classes.)

Disability Accommodation: Any student who feels he or she may need an accommodation based on the impact of a disability should contact me individually to discuss your specific needs. Also, please contact the College of Charleston, Center for Disability Services http://www.cofc.edu/~cds/ for additional help.

Inclement Weather Contingency Plan:
Should a class meeting be canceled due to inclement weather or any other emergency situation, please refer to the piazza class page for assignments or other information on what to do to replace that class meeting.