Introduction
The CSC116 credit-by-exam is a two-step process. The first step consists of writing an assigned program within a one week period. If the student scores a 70 or higher on the programming portion of the credit-by-exam, the student may then register to take the written exam. To receive credit for CSC116, the student should then pass the written exam with a 70 or higher. A post exam meeting with one of the CSC116 CBE Administrators is required to discuss the specifics of the exam results.

Starting the CBE Process
Email group-csc116-cbe-admin@ncsu.edu to state your readiness to start the CBE process. You MUST send the email with your NC State email account.

Program Notes
- The program must be completed by midnight, seven days after the program is assigned. The specific due date will be given to students via email.
- The program must be the student’s own work. Students may use a textbook and the Java 8 API to help them with the program. Submission of the program via email implies electronic signage of the following Honor Pledge:

  “I have neither given nor received unauthorized aid on this test or assignment.”

  Any violations of the Academic Integrity Policy will result in failure of the credit-by-exam and the violation will be reported to the Office of Student Conduct.
- The program should be emailed to group-csc116-cbe-admin@ncsu.edu as specified in the instruction email, by the stated deadline for grading.
- You must earn a 70 or higher on the program to take the written exam.

Exam Notes
- You must arrange to take your written exam within one week of receiving your program results (assuming that you scored a 70 or higher on your program).
- The exam must be completed in 3 hours.
- The exam is closed book, closed note.
- No electronic devices of any kind are needed or allowed. Cell phones and laptops should be closed and turned off.
- You may not use any exams from previous offerings of CSC116 to study with.
- Write answers in the spaces provided; do not attach extra pages.
- You may be asked to write code and complete programs and answer multiple choice and short answer questions. The code that you write is expected to compile and run; mistakes that would cause your code to not compile or run incorrectly will result in a loss of points.
- The exam must be the student’s own work. You will sign a version of the Honor Pledge when taking the exam. Any violations of the Academic Integrity Policy will result in failure of the credit-by-exam and the violation will be reported to the Office of Student Conduct.
- You must earn a 70 or higher on the exam to receive credit for CSC116.

Textbook

The textbook website (http://www.buildingjavaprograms.com/) provides valuable resources including lecture slides that may be used to study for the exam. Additionally, the Practice-It section provides an electronic environment for practicing coding with feedback on correct/incorrect code. Many of the problems at the end of the text book chapters are available in the Practice-It tool.
You will also want to refer to the following course packs and websites to supplement the course textbook:


You may use other textbooks to study for the exam.

Java

All coding questions for the exam must be answered using the Java programming language. We are currently using Java 8, and expect answers to conform to Java 8. Java 8 allows for the use of generics and provides the Scanner class for console and file-based text input.

Course Learning Objectives

1. apply classic problem-solving techniques to simple computational and information-management problems (without reference to any programming language), specifically
   - breaking large problems into smaller ones,
   - sequential analysis of solution steps,
   - logical analysis of alternative cases,
2. evaluate an arithmetic expression using order of operations, promotion from integer to floating-point types, and integer division,
3. use a programming language to write code that selects one of several alternatives based on more than one predicate,
4. use a programming language to write a loop whose exit depends on more than one predicate,
5. find and correct logical programming errors using debugging printout, pencil-and-paper tracing, and systematic search (to locate where an incorrect decision or value first appears),
6. implement an object-oriented design that has at least two interacting classes,
7. write and document programs that adhere to specific coding and documentation standards (e.g., javadoc for documentation; conventions regarding the naming of classes and methods, definition of constants, indentation, etc.),
8. use the Java system classes to do text-based input and output,
9. construct and use arrays with one and two dimensions

Topics Covered

- Procedural decomposition
- Arithmetic expressions
- Promotion of integers to floating-point types
- Integer division
- If statements
- Loops (e.g. for loops, while loops, do-while loops)
- Errors (e.g. compilation, syntax, and logic errors)
- Object-oriented design and implementation
- Javadoc
- Console-based input and output
- File-based input and output
- Arrays