CSCE 155H – Honors: Computer Science I
RAIK 183H – Honors: Computer Problem Solving Essentials

Fall 2016

"If you really want to understand something, the best way is to try and explain it to someone else. That forces you to sort it out in your own mind... that's really the essence of programming. By the time you've sorted out a complicated idea into little steps that even a stupid machine can deal with, you've certainly learned something about it yourself." –Douglas Adams, *Dirk Gently's Holistic Detective Agency*

"In my experience, you assert control over a computer—show it who's the boss—by making it do something unique. That means programming it… If you devote a couple of hours to programming a new machine, you'll feel better about it ever afterwards" –Michael Crichton, *Electronic Life*

"There are only two kinds of languages: the ones people complain about and the ones nobody uses." –Bjarne Stroustrup (creator of C++)

"I came into this class able to code by constantly using references and needing to double check my code. I left as a coding machine. The rate at which I did the last homework was like that of whole other person." –Previous student via course evaluation

Course Info

Prerequisites	Appropriate score on CSE Placement Exam; Math 103 or equivalent
Instructor	Chris Bourke cbourke@cse.unl.edu Avery Hall 363
Office Hours	See Blackboard
Textbook	See Blackboard
Labs & Teaching Assistants	See Blackboard

Course Description

This course provides an introduction to problem solving with computers. Topics include problem solving methods, software development principles, computer programming, and computing in society. This course is recommended for students majoring in Computer Science and Computer Engineering.

This is an Honors section; only students who are members in good standing of the UNL Honors Program or CSE Honors Program or those who have been invited may enroll in this course. As a separate Honors section, topic coverage will be the same but at a quicker pace and greater depth. Additional advanced topics may be covered as time permits. Expectations on student performance will also be higher than a normal course.

Course Objectives

The official specification for this course lists the following general course objectives.

1. Mastery of the fundamentals of programming in a high-level language, including data types and rudimentary data structures, control flow, repetition, selection, input/output, and procedures and functions.
2. Familiarity with problem solving methods, including problem analysis, requirements and specifications, design, decomposition and step-wise refinement, and algorithm development (including recursion).
3. Familiarity with software development principles and practices, including data and operation abstraction, encapsulation, modularity, reuse, prototyping, iterative development, exception handling, documentation, coding conventions, and testing.
4. Exposure to computing topics, including algorithms for searching and other problems, graphical user interfaces, event-driven programming, and database access.
5. Exposure to the history of computing.

Course Topics

1. Introduction to Computing
2. Introduction to Java and C
3. Conditionals
4. Loops
5. Functions & Modular Programming
    a) Pointers & References
    b) Simple Data Types & Numeric Representations
    c) Error Handling
2. Arrays
    a) Dynamic Memory & Memory Management
3. Strings
4. File Processing
5. Encapsulation
    a) Structures
    b) Objects & Constructors
6. Recursion
7. Searching & Sorting
8. Graphical User Interface & Event-Driven Programming
9. Introduction to Databases & Database Connectivity

Relationship of Course to ACE

This course will satisfy Learning Outcome 3: Use computational and formal reasoning (including reasoning based on principles of logic) to solve problems, draw inferences, and determine reasonableness.

- **Learning Opportunities**:

The lectures, together with homework and programming assignments and the weekly structured laboratory sessions, teach students methods for developing and implementing algorithms to solve problems. That is, the course not only teaches students about how to design algorithmic solutions, but also teaches students about how to engineer designs into working software. The engineering process of designing and implementing a program involves significant debugging, testing, and refining code. These activities teach and reinforce reasoning and inferencing: a student must develop tests to reasonably indicate program correctness and must draw inferences when diagnosing why a program does not compile, crashes, or generates incorrect output. Also, an algorithm is fundamentally a logical sequence of steps that, given a set of inputs, generates a set of outputs.

The course includes approximately:

- 45 hours of lectures each designed to explore concepts and paradigms that are central to the field of computer science.
- 15 hours of structured laboratory sessions, each designed to train students to apply what they learn in the lectures to actual implementations and analyses of algorithms and software.
- Several homework and programming assignments designed to help students learn about methods for designing algorithmic solutions and the practices of implementing solutions as correct software.

- **Outcome Assessment**:
  A variety of student work is used to assess achievement of the outcomes, including exams, homework and programming assignments, and structured laboratory work. Exams require students to demonstrate their knowledge in a written format. The programming assignments are inherently practical demonstrations of problem solving and algorithm development, with reasoning and inferencing to produce programs that compile, run, and compute the correct output. The laboratory work supplements the lectures and to provide supervised hands-on experiences of problem solving, algorithm development, and the realization of a computer solution. The students submit their results from solving the lab problems and performing the specified tasks. Laboratory pre-tests, worksheets, and post-tests are graded. The student must pass pre-tests prior to beginning the lab, complete worksheets with results, and take post-tests prior to the end of the lab. In summary, all of the following provide opportunities for students to demonstrate their skills related to the learning objective:

  - There are midterm exams and a comprehensive final exam.
  - There are several programming assignments.
  - There are weekly structured laboratory assignments.

Accommodations for Students with Disabilities

It is the policy of the University of Nebraska-Lincoln to provide flexible and individualized accommodations to students with documented disabilities that may affect their ability to fully

participate in course activities or to meet course requirements. To receive accommodation services, students must be registered with the Services for Students with Disabilities (SSD) office, 132 Canfield Administration, 472-3787 voice or TTY.

Grading

Grading will be based on homework, quizzes, labs and exams with the following contributions

Homework	75%
Labs	15%
Midterm	5%
Final	5%

Scale

Letter grades will be awarded based on the following scale. This scale may be adjusted upwards if the instructor deems it necessary based on the final grades only. No scale will be made for individual assignments.

A+	>= 97	B+	>= 87	C+	>= 77	D+	>= 67	F	<60
A	>= 93	B	>=83	C	>= 73	D	>= 63		
A-	>= 90	B-	>= 80	C-	>= 70	D-	>= 60		

Homework

There will be several homework assignments constituting the bulk of your grade. Assignments are due at the beginning of class. Code and other relevant files must be submitted using CSE's webhandin. If there are written portions of the homework, they should be typed. You should typeset code snippets using a monotype font (Courier for example) for readability. Figures may be hand drawn, but you are encouraged to use some sort of software to render them.

Note: the final homework may be due as late as Friday of Dead Week. As per the 15th week policy, this serves as notice.

Labs

There will be weekly labs that give you hands-on exercises for topics recently covered in lecture. The purpose of lab is not only to give you further working experience with lecture topics, but also to provide you with additional information and details not necessarily covered in lecture. Each lab will have some programming requirements and a supplemental worksheet.

Labs are setup as a "peer programming" experience. In each lab, you will be randomly paired with a partner. One of you will be the "driver" and one of you will be the "navigator". At the same terminal, the navigator will be responsible for reading the instructions and telling the driver what to do next. The driver will be in charge of the keyboard and workstation. *Both* driver and navigator are responsible for suggesting fixes and solutions together. Neither the navigator nor the driver is "in charge," it is an equal partnership. Beyond your immediate pairing, you are encouraged to help and interact and with other pairs in the lab.

Unless otherwise stated, you are required to finish the lab by the end of your regular lab meeting time. A lab instructor must sign off on your lab worksheet and you must turn it in to receive credit. Labs that have not been completed by then will be given a zero.

Exams

There will be a midterm exam and a comprehensive final exam. Details will be announced closer to the exam dates.

Grading Policy

If you have questions about grading or believe that points were deducted unfairly, you must first address the issue with the individual who graded it to see if it can be resolved. Such questions should be made within a reasonable amount of time after the graded assignment has been returned. No further consideration will be given to any assignment a week after it has been graded and returned (regardless of whether you fail to pick it up when handed back). It is important to emphasize that the goal of grading is consistency. A grade on any given assignment, even if it is low for the entire class, should not matter that much. Rather, students who do comparable work should receive comparable grades (see the subsection on the scale used for this course).

Late Work Policy

In general, there will be no make-up exams. Exceptions may be made in certain circumstances such as health or emergency, but you must make every effort to get prior permission. Documentation may also be required.

Homework assignments have a strict *in-class* (at the beginning) due date. Any written portions should be handed in hardcopy in class while softcopies should be handed in using webhandin as specified in individual homework assignments. The webhandin program that you will use enforces a *strict* handin time based on the CSE server's clock. Programs that are even a few seconds past the due date/time will be considered late.

Furthermore, many assignments will have requirements (file naming conventions, package requirements, command line input requirements, etc.) that will facilitate grading through an automated script. This script has been made available to you through the webgrader interface. You are expected to utilize this webgrader interface to ensure that your program is running as required and to fix any issues prior to the final due date (you may handin and run the script as many times as you like up to the due date). Note, however, that the script should not substitute for developing your own test cases and should not be used as the primary resource to debug your program; instead it is intended as a last-check mechanism.

It is understandable that unforeseen events may interfere with your ability to submit all homework assignments on time. As such, this course allows the following late work policy: you may hand in any *one* assignment up to one (academic) week late. Any submissions after a week will not be considered and will be given an automatic zero. Any late submissions after using your one “free pass” will not be considered.

If you work with a partner on a late assignment, both of your late passes will be used. If two people work together on an assignment and one of them has already used their late pass, the other may not use their late pass for both of them.

In addition, failure to adhere to the requirements of an assignment in such a manner that makes it impossible to grade your program via the webgrader means that a disproportionate amount of time would be spent evaluating your assignment. For this reason, we will not grade any assignment that does not compile and run through the webgrader. You will be expected to use your late pass to fix the issue(s) before we will consider it for grading. Failure to address the issue or submitting assignments with such problems will result in an automatic zero.

Dead Week Policy

In conformance with UNL's 15th Week Policy (see Registration and Records main webpage, http://www.unl.edu/regrec/), be aware that the final homework may be due during the final week of classes. Further, there will be a regularly scheduled lab during the final week of classes. Finally, all assignments, homework, labs or otherwise, will have a strict final due date during the final week of classes. This supersedes any unused late or screw-up passes that you may have (that is, such passes cannot be used to extend the due date of any assignment past the last week of classes).

Academic Integrity

All homework assignments, programs, quizzes, and exams must be your own work unless otherwise stated. No collaboration with fellow students, past or current, is allowed unless otherwise permitted on specific assignments or problems. The Computer Science & Engineering department has an Academic Integrity Policy. All students enrolled in any computer science course are bound by this policy. You are expected to read, understand, and follow this policy. Violations will be dealt with on a case by case basis and may result in a failing assignment or a failing grade for the course itself. The most recent version of the Academic Integrity Policy can be found at http://cse.unl.edu/academic-integrity

Honors Course

As an Honors course, topics are covered in a "greater depth" than a regular course. All CSCE 155 courses cover the same topics but have a different focus and use a different language (155A uses Java, 155E uses C). The depth component of this course will be to cover all the topics but will include coverage in both Java and C (and will highlight the differences and idiosyncrasies of both languages).

Many of the programming assignments will include questions about both languages: some exercises will be required to be done in one language or the other or may give you a choice as to which. Exams may also have questions that require knowledge of both languages. All students are responsible for material on both languages.

Weekly lab assignments will also have two versions: one in C and one in Java. Both versions will cover the same concepts and have the same exercises. However, each student is required to complete only one version (the choice is left to the student). However, you are *highly encouraged* to complete *both* versions to gain a better knowledge of both languages.

Communication

The best way to communicate with your instructor is through Piazza. This is a free online forum where you can pose questions anonymously or privately. The instructor as well as the TAs and your fellow students can answer and engage in discussions.

In addition, the instructor and teaching assistants will communicate with you either directly or through the CSE or Blackboard email system. You are responsible for ensuring that the email associated with your Blackboard account is up-to-date and that you are regularly checking it.

There is an anonymous suggestion box that you may use to voice your concerns about any problems in the course if you do not wish to be identified.

Finally, I will hold regular office hours (to be announced) and will make myself available by appointment; please email me to set one up.

Help

Your success in this course is ultimately your responsibility. That said, there several outlets for you to seek help and assistance.

1. Your Instructor – Attend lecture regularly and engage in class discussions, ask questions in class, visit me during my office hours or setup a meeting time to see me, email me!
2. Your TAs – TAs hold regular weekly office hours, visit with them and ask questions, ask for examples, etc.
3. Student Resource Center – Though they may not be your direct TA, all Graduate Teaching Assistants (and some advanced undergraduates) hold regular office hours in the Student Resource Center (http://cse.unl.edu/src/) open Monday thru Friday 9AM to 7PM and should be staffed most hours. Ask for help from anyone in the SRC.
4. Course materials – start on assignments early, attend lectures and labs, work extra problems from the book, read all required (and optional!) materials.
5. Your colleagues – Chances are, if you are having problems, your classmates are having them too. Discussion and dialog among students is encouraged (within the parameters set by CSE's academic integrity policy, this course's policy, and policies set for individual assignments).