CSI 201 – Introduction to Computer Science – Spring 2014
http://www.cs.albany.edu/~sdc/CSI201 (Public Web site)

Syllabus

Instructor:  S. Chaiken
Office Hours:  Mon: 12:00PM-1:00, 2:45-3:45
             Wed: 12:00PM-2:15
             LI 96H, 442-4282
             sdc@cs.albany.edu
             other: appointment or drop-in if I’m not busy

Prerequisites: Inclination to use elementary mathematics and the detailed, precise thinking styles
taught in mathematical sciences. No previous experience with computers is required, just a willingness
to learn from working hard. BUT: Statistically, success in this course tends to increase with
increasing levels and grades in college level mathematics study.

This course is designed primarily to introduce the Computer Science discipline, which begins
with computer programming, to its majors and minors. It is an option for some requirements in
certain other majors or minors. Unlike introductory courses that are broad, shallow or conceptual
introductions to various aspects of a field, this course achieves its purposes by concentrating on
one central skill: computer programming. In this course you will learn to create programs in the
Java language. Your programs will direct the computer to use somewhat low-level capabilities to
perform various tasks. Many will be cool tasks to draw graphics and process digital images. You
can then show off your unique creations to friends and family on Facebook or other web sites, or
in person.

You will learn to control the basic internal calculation and data storage capabilities of today’s
computers. As you learn more about Java, and how to use it to write more sophisticated programs,
we will take up some of the areas and issues of computer science, such as how to use computation
for interesting applications, and issues like computational efficiency.

You will also be required to employ the most basic information technology skills for programmers, learning them as necessary from course staff and classmates. (Blackboard, common to other UA courses, is used. But, besides Blackboard, this course requires studying materials located by urls on the public Web, writing and editing programs and other text files, compiling and running programs, web up and downloading, managing folders/directories of your files, making archives, transferring archives, files, and folders/directories between computers, keeping backups and version histories, etc.).

Our curriculum makes the basics more interesting and easier to imagine by applying them to
make drawings and to manipulate digital images which you might get from your digital camera
or download from the Web. The basics are programming the computer, by writing, testing and
debugging codes you write in Java to store and sequentially process data.

This is a hands-on, cumulative skill practice and development kind of course—not a course of
just learning facts. The critical thing you must do to succeed to actively participate (not just listen,
read and study). The course plan gives incentives for regular, frequent and sometimes challenging
engagement with the subject’s intellectual work, but also makes it more fun and easier to maintain
the necessary attention.

Required Materials:
1. **Attendance** and “iClickers:” The iClickers will be used for in-lecture preparatory reading quizzes, warm-up exercises, thinking exercises, skill and concept quizzes, and polls. Which questions count for correctness and which just count for participation will be announced. You MUST get, register, bring and use your iClicker in the Tuesday January 27 lecture to AVOID losing points which contribute to your grade. However, everyone will be excused from the clicker components of up to 3 missed lectures. But some material presented in the lectures will not be presented or practiced anywhere else. Make sure you have a trusted friend to lend you notes about what the professor said or emphasized, and the content of class discussion if you are going to miss a lecture. Materials shown or distributed will be posted on the course web site shortly after each lecture.

If you don’t have an iClicker from last semester, **BUY ONE NOW** from the UAlbany or other college bookstores, or used from another student. And everyone get a supply of fresh batteries! Version 2 is sold now but the cheaper Version 1 iClicker is OK to buy used or keep using.

Don’t pay for and try to use the “iClicker” service and apps for Android and other smartphones. That technology is (for now) unreliable compared to the dedicated iClicker devices because those apps use WiFi and the world wide Internet to connect to the prof’s. iClicker base software. The service is expensive and is not supported yet by UA’s WiFi network.

To get iClicker grade points, you must register your iClicker with your name, UA NetId and the code printed on your iClicker’s back by using the web site [http://www.iclicker.com/registration](http://www.iclicker.com/registration). Doing this before the next lecture, when iClicker work begins to count, will ensure you getting all the grade credit. However, if you registered your iClicker for a UAlbany course this past Fall you need not register it again. (iClicker registrations expire each June.)

2. **Package:** MyProgrammingLab Web account packaged with the custom printed UAlbany Introduction to Computer Science, comprised of selected chapters from two regular textbooks. This is for sale at the UA bookstore and at Mary Janes, with ISBN 1256994987.

DO NOT REGISTER AND USE MyProgrammingLab until I provide updated instructions from the publisher!!

The chapters are selected from the two textbooks:


Texts are not allowed in exams, so sharing is OK, but you are expected to get a MyProgrammingLab subscription.

(Do not use, or spend any additional money for the CD packaged with the Guzdial and Ericson book: All the software in up-to-date versions is downloadable free from the Web.)

Reading and “warm-up” assignments will be assigned for which **clicker QUIZZES and warm-up exercises** will be given in the very next lecture. I expect every student will at least have access to the textbooks.
The main reading assignments will be short, and focused; and most will be related to MyProgrammingLab assignments followed by iClicker quizzes. The first main reading is Chapter 1 of the Custom Edition (Chapter 1 of Gaddis).

Some readings might be longer and lead to assignments to submit questions for the professor to cover in class. But that won’t be at the beginning.

The lectures will show how to run DrJava on UA “Information Commons” Windows PCs, and tips for installing it and the textbook’s supporting Java programs.

The first lab will begin in the first full week of classes, Monday, January 27.

The first homework includes you trying to get you to run Dr. Java in a UA library or personal computer, write Java programs, save them in .java files and, after labs, submit them to Blackboard. It’s critical to know right away (make a note on paper!) exactly where your files are on every computer you use! Then, resuming saving and resuming work and finally uploading it to Blackboard for grading and credit won’t be distracting when the assignments get longer and harder!


4. Project assignments are software specifications: Every word on project assignments, and in readings they rely on, is required reading. The professor and TAs will explain as needed, but will ask you about these readings when addressing your difficulties.

5. On-line learning and practice with MyProgrammingLab is required and your scores in them are counted in your grade. The MyProgrammingLab questions will be selected to be best done while you are reading the Gaddis chapters of the textbook The Gaddis textbook will be available on-line with MyProgrammingLab.

6. Readings posted or linked on the Web may be assigned. Course lecture notes and Web enrichment references are provided during the course.

1 Learning Goals

The chief learning goal is to demonstrate fluency and problem solving skills in computer programming at levels expected during the first semester of university computer science major study.

Briefly: Given a sufficiently detailed description, or her/his own conception, of what a computer can do and a strategy for doing it, the student shall write a program that makes the computer do it by implementing the given strategy.

Fluency consists of competence and confidence together. Your confidence to handle novel programming quirks on exams, new assignments and your own inventions is achieved by working out, repeatedly, within your own consciousness all the logic and details that go into homework solutions, practice questions and examples. The midterm and final exams will have coding questions: To actually write code to solve new problems based on the practice the projects required, but not the same! Memorization or thoughtless copying from the note sheet will not get a high grade on these exam questions.
This goal can only be achieved by repeated practice of programming in some specific language, and for this course, and its followup ICSI310, Data Structures, that language is Java. At Albany, the third major course, ICSI333, Programming at the Hardware/Software Interface, teaches more about how computers work and more intricate programming techniques using C and Assembly Language. A goal of University level computer science is to be able to quickly learn to use new programming languages after having mastered one or two. That happens after a year or two. For now, we concentrate on Java.

The **purpose** of this course at UAlbany is to enable you to either (a) work with or (b) become one of those experts who invent or construct software and maybe scientific ideas about computing. Scientific ideas, the content of computer and other mathematical sciences, are ideas (about the things of the science) that are precisely formulated, often with formulas and logic, that fit observations repeatedly, frequently measurements, better and better as the science develops.

This purpose is different from the most other UA computer science courses with no prerequisites because the latter teach broad and/or shallow introductions or overviews, the roles and issues of computing in commerce, culture and society, or how to use software you might need, choose or buy, rather than write software yourself.

If your are a CS major, ICSI201 is an important start, but the next course, Data Structures is **harder and even more important!** So should aim for an “A” in ICSI201 now. (Ask any professional programmer.) If you get a B, that is fine. If not, consider switching majors. If you get C or below in ICSI201, CS is probably not the right major for you.

### 2 For UA Students Not Majoring in Computer Science

Your major or minor might specify ICSI201 as just one option within a list of other courses, but you don’t want (or need) to commit time and mental energy now (or ever) to grapple with detail work like what programmers do for a living. In that case, you should consider the other options in your major’s list. If ICSI201 is required, discuss its purpose and possible alternatives with your major or minor advisor or mentor.

Computer science minors might want to take ICSI201 later, after electives with less intensive programming assignments, but they must take ICSI201 early enough to prepare for a required Data Structures course, which is harder than ICSI201.

If you are undecided, give it the UAlbany Old Purple college try! Even if you've never programmed, but want to try something new, or perhaps like brain teasers, puzzles, practicing musical instrument playing or singing for performance or fun, building things and/or taking things apart, drawing or painting your own art work or making your own clothing designs, puns or other language jokes, etc., it’s likely you will find programming great fun and very satisfying. Especially if you go beyond what’s assigned and be creative.

### 3 Other Meetings

**IMPORTANT:** You must have and know your UAlbany NetId and password for lab and homework. If you do not have these, finish your UA registration as soon as possible and get needed further help in LC-27, ITS Helpdesk.

The lab periods will be used for learning exercises conducted by the assistants where questions
may be quickly answered and discussed. Some labs may feature self-paced tutorials. Accomplishments in every lab will be assessed with the assistant checking your work or collecting work on paper PLUS the in-lab and followup homework assignment material submitted electronically to Blackboard after the lab period is finished.

You must register for some lab session. If the session is overfull, the professor will try to accommodate you entering a different session. If you cannot come to your registered lab period in any single week, or you need more time to complete its requirements, you can go to a different one that week so long as there is physical space available (which there is likely to be.)

The schedule for lab sessions and staff office hours will be maintained in the course web site. The prof. and TA’s also make appointments; their email addresses and rooms are posted too.

Email and the Discussion Board

Posting, replying to and reading questions and answers using the course discussion boards on Blackboard is one excellent resource, as long as you don’t post code that can be copied mindlessly. If in doubt, email it to the professor or a TA and ask.

We will also answer “reasonable” questions by email; which means definite questions that have definite and short answers; and which are at the academic level of the course and on course topics. We may edit and post on the Web or course Blackboard discussion board questions and answers we think would be helpful to others (but I will email you a personal reply too.) We might ignore any others. We will not write or copy long explanations that repeat material we presented or are available for you to read.

Also, if you spent say one half hour or more on one thing you are stuck with in programming, send an explanation and the code (the entire Java file please) to the prof. for a tip!

Grading

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Midterm</td>
<td>Mar. 13, 2014 (Thu)</td>
<td>In class</td>
</tr>
<tr>
<td>Final</td>
<td>May 15, 2014 (Thu)</td>
<td>8:00AM-10:00</td>
</tr>
<tr>
<td>iClicker</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MyProgLab</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Labs</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Homework Projects (7 to 9)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Awarding of the letter grades will be based on cutoffs applied to the score as calculated above. The maximum cutoffs are listed below:

<table>
<thead>
<tr>
<th>Grade</th>
<th>Cutoff</th>
</tr>
</thead>
<tbody>
<tr>
<td>A,A-</td>
<td>≥ 85%</td>
</tr>
<tr>
<td>B+,B-</td>
<td>≥ 70%</td>
</tr>
<tr>
<td>C+,C-</td>
<td>≥ 60%</td>
</tr>
<tr>
<td>D+,D-</td>
<td>≥ 40%</td>
</tr>
<tr>
<td>E</td>
<td>&lt; 40%</td>
</tr>
</tbody>
</table>

However, the professor reserves the right to scale particular assignment or exam score components to compensate for variations of their difficulty; and also award a higher grade for substantial
improvement. Estimated grades based on the midterm exam score will be announced with the midterm results, Mar. 28 or sooner.

Details regarding the exams coverage will be announced later. Sample questions will be available.

**Homework Programming Projects:** There will be five to seven programming project assignments. Some will be continuations of previous ones.

**Submissions that are not testable because they were not uploaded properly, don’t compile or are not structured as specified are given ZERO points initially.** What this means is taught in labs and written into project assignments.

To get more than zero, you must work with a TA to make your work testable and learn how to meet the technical specifications better in future work. A satisfactory submission will then earn the score for the programming problem solution, with 15/100 (for the first 2-3) or 20/100 points deducted (as an incentive to get it right the first time!)

When you follow the practice of incremental development, or at least compile and test your code **that one last time** before you turn it in, you will always find and correct any syntax errors. It is expected that you will start the projects early enough so that you can get any help you may need to get the syntax right.

Although the programming work can be done on your own computers, and this is encouraged, problems (hardware, software, network access) with your system will not be accepted as excuses for late or missing programming project or lab completion work. Information Commons computers are provided in the UA Libraries as the “last resort” for students with computer or network problems.

**However,** delays will be granted in case of weather problems or problems with the Blackboard or Information Commons UAlbany infrastructure.

**Cheating versus Learning**

Project assignments and all examinations in this course are individual exercises. The work that you do must be yours—not that of other students, friends, tutors, Web volunteers or posters, etc.

While it may seem like a time-saver for learning and doing assignments to copy something, this strategy will backfire on the exams. It will backfire because we design the tests for you to demonstrate specific thinking and problem solving skills that the assigned practice develops into your own, individual mind. It will again backfire when discover yourself unable to do subsequent assignments that rely on the skills you missed developing because you copied the work of someone else who actually has them. Finally, it will backfire when a job interviewer asks you to solve and explain a simple programming problem on the spot. Your “friend” who lent you his or her excellent CSI201 solutions will get the job, not you!

It will probably backfire even if it feels like you understood the solution at the time you copied or even worked it out collectively! This happens in your mind for the same reason that your body cannot learn a skill like ice skating, music playing or football by just watching. Making you mind struggle with programming puzzles **actually builds** physical connections among the neurons in your brain. Doing that repeatedly makes you do it faster and more confidently each time because it **strengthens** the newly made connections.

You are encouraged to form study groups, discuss assignments and techniques in general terms, especially on the Blackboard discussion boards, etc., but the assignment work you turn in must
be your own work. In particular, two or more people may not create an assignment together and submit it for credit.

The particular rule is that you are encouraged to explain to each other anything for the course in English (or other natural, human language like Chinese or French) and in diagrams, geometric sketches, etc.

BUT, you must not communicate by typing or dictating, nor copy (from people or sources other than the textbook and this course’s notes) by pasting or manually typing, actual Java code that constitutes parts of solutions to assignment problems. Nor may you communicate text-editing instructions that may be followed without thinking or understanding.

Briefly: NEVER COPY CODE YOU ARE ASKED TO WRITE YOURSELF!!!

BUT WHAT IF we or a TA/tutor WORK OUT SOLUTIONS TOGETHER, or somebody just shows me their answer or a part of it? Realistically, this happens! Remembering that the goal is learning (to demonstrate solving varied problems on exams or job interviews), here is the thing to do: Put (or throw) away your notes with literal answer fragments and reproduce the ideas your friends or helpers showed you. But what if you can’t yet do that? Talk to somebody who really understands the subject, who asks you specific content questions to see where your understanding reaches and ends, and leads you to understand more.

This takes time, so another tip: Start projects early! Programming tasks USUALLY (even for pros) lead people to unexpected mistakes or misunderstandings, and take longer than most people think when they first start the task!

You might find or invent fragments or examples of Java code that may be helpful for others to see, but which will not cause classmates to fail to learn because enables them to skip thinking for themselves. If there is any doubt, you must pass them through the professor (by email) to review before posting them on discussions or otherwise communicating them.

For example, if you show a classmate or tutor code with bugs, and he or she notices that a variable is not initialized properly, he or she should tell you something like “the variable you seemed to use for counting something might not be initialized properly because the count is wrong”, but NOT “Just change the 1 in your line ‘int count = 1;’ to a 0 and you should be fine.”

If you get a tutor, make sure the tutor will really help you learn by guiding you to solve the problems yourself (like the tutor asking questions, pointing out details you overlooked in reading assignment instructions, telling you what to read, finding other explanations on the web or books, etc.), and is not so lazy (or incompetent!) that he or she just does your homework for you, perhaps wrong!

iClicker “fraud” such as using another student’s iClicker to help a friend skip class is also considered cheating. Our 3 missed lecture policy is intended for you keep your course grade up even if you choose to miss class occasionally.

Version Histories: After a few weeks, we will teach and begin to require that you use a simple form of a practice followed by every serious software developer today: Keeping a digital history of your programming work files as your work progresses. All the history files and folders/directories (in a compressed “.zip” archive) must be included in your submitted project work.

• You will not lose good, previous work when you “mess up” your program.
• It encourages and demonstrates the important practice of incremental development, in which you only make small (tiny!) extensions, corrections or improvements to already working code. The older code is useful to test the newer. The principle is to always test and perfect each little challenge before moving on to the next. Compilable versions that solve more and more of the assigned problem are kept available at all times, so a “programming mess” never gets far out of control.

• It documents that you have really done the programming project by yourself. It also enables to course staff (prof. and TAs) to give you better assistance by being able to tell where you went wrong and where to pick up from.

**Academic Integrity:**

1. A zero score, for all students involved for the affected assignment or lecture (zero averaged in, not a missed lecture) where an incident occurs.

2. A written report will be sent to the Vice-Provost for Undergraduate Studies, according to University policy intended to detect cheating in a multiplicity of classes.

3. For repeated cheating or cheating on an exam by impersonation, communication or copying from somebody else’s paper, may result in a failing grade and/or a referral of the matter to the University Judicial System for all students involved.

We may do automated code comparisons between submissions of current students together with submissions from prior course offerings to detect copying.

Violations of Computer and Network Usage policy on laboratory or other University systems are regarded as academic integrity violations, like cheating. If you want to try some kind of computer stress testing experiments, talk to the professor and you may get permission to do it responsibly on a special, isolated research system!

**Passwords**

You are also warned against disclosing your password to anybody either willfully or by accident. Another person knowing your password can really get you into trouble. Similarly, if your password is compromised or you change the permissions of your files so that others can read them, they might copy your work and you might get caught when we compare submissions.

**Network S: Drive Quota**

It’s only 100MB at present, but it is backed up several times a week!

You can easily observe what’s left by logging in to a Library computer; you might, as you get into using the command line, prefer to use the `quota -v` command to monitor your network drive disk quota. It is wise to take steps to monitor and reduce disk space if necessary because using it up may make your account becoming unusable when you need it most. If this happens, corrective actions might take several days. Lateness of programming projects or lab exercises will not be excused, and ITS generally doesn’t take “urgent” system administration actions for student problems due to this kind of thing.
Make-up Exams:
Make-up exams will be given only for valid (and verifiable) excuses (e.g. a major medical situation, accident, etc.). In such a case, it is your responsibility to contact the instructor ahead of time if at all possible.

Incompletes
A grade of I will only be given for genuine extenuating circumstances that are beyond your control after the midterm point. Both of the following conditions must be met:

1. Your work must be in good standing as of October 18, 2013 (about two weeks before the drop date), defined as follows: You must have an average score of at least 70% on the programming assignments and at least 70% on the everything else due up to that point; and further, your midterm grade must also be equivalent to at least a C. Therefore, if you miss the midterm or have performed poorly on exams, assignments, etc., you are not eligible for an I grade. Your choices are to work smartly and hard to catch up, get a poor or failing grade, or drop the course by the drop date of Wed. Nov 6, 2013.

2. Written documentation must, upon request, be supplied about the extenuating circumstance either by you or by the University administration. The Vice-Provost for Undergraduate Studies and her assistants are there to assist you and will send emails to your professors that request appropriate accomodations.

Under no circumstances will the condition for completing an I grade be repeating the entire course without a new registration.

Disabilities, etc.
Accomodations are made for clients of the Disability Resource Center (DRC) upon adequate prior notice and according to that center’s policies. Many Albany students had much greater educational success after strategies to work around a wide range of mental and/or physical personal variations were recommended by the experts at UA’s DRC. Of course privacy prevents their identities from being disclosed.

Students with genuine continuing hardship situations, or any disability related problems with Lab or homework computer usage should confer with the professor by Sept. 18.