PREREQUISITES: CS 142 Java I (C or better).

INSTRUCTOR:

<table>
<thead>
<tr>
<th>Name</th>
<th>Dave Straayer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Phone #</td>
<td>(253) 460 4390</td>
</tr>
<tr>
<td>Note:</td>
<td>I’m a little hard of hearing. If at all possible, use email instead of phone. If you must leave a phone message, leave your phone number slowly, clearly, and twice.</td>
</tr>
<tr>
<td>Email</td>
<td><a href="mailto:dstraayer@tacomacc.edu">dstraayer@tacomacc.edu</a></td>
</tr>
<tr>
<td>Web Page</td>
<td><a href="http://www.tacomacc.edu/home/dstraayer/">http://www.tacomacc.edu/home/dstraayer/</a></td>
</tr>
<tr>
<td>Office</td>
<td>15 – 235</td>
</tr>
<tr>
<td>Office Hours</td>
<td>10:30-11:20 Mo-Fr Or by appointment</td>
</tr>
</tbody>
</table>

CREDITS: 5 Credits

CLASS LOCATION: 16-206

CLASS Website: http://www.tacomacc.edu/home/dstraayer/published/CS143/

DAYS/TIME: Days: Mo – Fr; Time: 9:30-10:20

COURSE TEXT Starting Out with Java From Control Structures through Data Structures Ed. 2, Gaddis & Muganda, ISBN: 978-032-154-5862

The first edition will also work. It is essential to have a good Java text book for a class like this, but experienced programmers may be able to make do with any good Java reference.

CATALOG DESCRIPTION:

This is an intermediate programming course, using Java. It follows CS-142. Topics will include classes, interfaces, inheritance, polymorphism, exception handling, recursion, data structures, and an introduction to performance analysis and implementation trade-offs.

LECTURES, POWERPOINT, WORKSHEETS, AND ALL THAT JAZZ:

I use PowerPoint to plan and deliver my lectures. I find it a good lecture-planning tool, and it helps compensate for my not-so-great whiteboard handwriting. 😊 I will project my PowerPoint slides using the in-classroom computer and video projector. All the PowerPoint lecture slides will be printed and distributed the first full week of the class, and will also be available on my Canvas.

We will be programming in this class. I will frequently project examples of programming that I will be doing in “real time” during class. You will be able to follow along on your computer as I do this – I recommend it. The files I develop and use in class will be available through the class web site: http://www.tacomacc.edu/home/dstraayer/Published/CS142/. (I use a straight website, rather than Canvas for this, because my Eclipse workspace is automatically updated and available to students every time I save or run.)
Lectures will also be “captured” with the Tegrity system, which captures whatever is on the screen, plus my voice. You can get at the Tegrity-captured lectures through Angel.

If you expect to get value out of this class, you will have to participate. As I demonstrate code, I encourage students to interject questions and offer “what if” scenarios for us to explore together. I’ll make note of students who are active in class and award “extra credit points” to them.

**COLLEGE-WIDE STUDENT LEARNING OUTCOMES:**

The abbreviation following each objective refers to the College-Wide Learning Outcomes:

COM=Communication; CRT=Critical Thinking; IIT=Information and Information Technology;

RSP=Responsibility; LWC=Living and Working Cooperatively

**Program learning outcomes:**

1. Apply knowledge of mathematics and science to engineering related problems. (COK, CRT)
2. Design a system, component, program or process to meet desired needs. (COK, COM, CRT, IIT, LWC, RES)
3. Conduct scientific experiments, analyze and interpret the resulting data. (COK, CRT)
4. Communicate design ideas, solutions to engineering related problems or results of scientific experiments effectively, using both English and mathematical languages. (COK, COM, CRT, IIT)
5. Function effectively on a team to produce a cohesive and professional work product. (COK, COM, CRT, IIT, LWC, RES)

**COURSE LEARNING OBJECTIVES:**

Upon successful completion of this course the successful student will be able to:

1. Specify, design, and implement, test, and debug a Java program of moderate complexity. [2]
2. Be able to utilize Object-Oriented concepts effectively. [2]
3. Be able to implement programs as command-line programs, applets, or GUI programs. [2]
4. Understand and use common GUI controls. [2]
5. Be able to write a program that uses both vector and raster graphics. [1,2]
6. Be able to write programs using a commonly accepted style, including indentation, capitalization, and commenting. [2,4]
7. Be able to select and use appropriate data structures. [2]
8. Be able to understand the implications of algorithm analysis and use it to select appropriate algorithms. [1,2,3]
9. Be a part of a software development team that delivers a project beyond what an individual could accomplish in the available time. [5]

(These are all deeply involved in the IIT outcome, but I’d like to point out that few human activities so tax one’s CRT as programming. You will need plenty of RSP to succeed. The specification and documentation of programs requires considerably more COM skills than you may realize. Finally, you will learn how standards of object-oriented programming, coding style, and agreed-on interfaces help to enable team programming: LWC)

**TECHNOLOGY:**

1. You are going to need a modern computer for this. OK, you don’t really need to own your own computer, but trying to do a class like this just using campus lab computers seems pretty lame. The inexpensive “Netbook Computers” with an Intel Atom processor that can be had new for around $200 have proved just fine. Java is platform-independent. Several students have used Mac and Linux systems.
2. Java JDK whatever Sun is using as their current Java Development Kit release. See http://java.sun.com/javase/downloads/index.jsp. Don’t bother to install the NetBeans or Java EE editions.

3. We will be Eclipse, an open-source development environment. See http://www.eclipse.org. You don’t need the EE developer support; get the Eclipse IDE for Java Developers at http://www.eclipse.org/downloads/. Make sure Window Builder is installed. Since Eclipse is open-source, it is free. It has a nice editor and debugger in it. It is a bit complicated, and probably a lot more than we really need, but we’ll manage.

4. Canvas: we will be using this LMS (Learning Management System) for sharing documents like this Syllabus, assignments, worksheets, etc. Canvas also provides forums where you can consult with other students about assignments. Your gradebook is there, too. You will be able to access Tegrity-recorded lectures from Canvas. We also use it for electronic submission of some assignments. I suggest that configure your Canvas account so that all important announcements are also delivered to your cell phone as text messages. You access Canvas with any internet-connected browser via the TCC Portal. Please use only Canvas internal Email to communicate with me.

EVALUATION CRITERIA (GRADING):

1. Programming (55% of your grade): I will post programming assignments on the class web page. There will be approximately one per week.
   a. See the class website for the assignments: http://www.tacomacc.edu/home/dstraayer/published/CS143/
   b. Assignments have a due date and time (usually a Sunday at noon), but late assignments may be accepted with loss of the “on-time” part of the grade.
   c. DO NOT COPY OR LEND OUT YOUR CODE. If your code is copied by another student, both of you will receive a zero for the assignment in question. The evidence of the plagiarism-detection program describe below will be used to determine copying.

2. Tests/Quizzes (30% of your grade): We will have several tests. My current plan is to use stock test questions provided by the author of our book, and I’ll install them as Canvas quizzes. I may substitute real-time programming quizzes. See the schedule below.

3. Team Project (15% of your grade): We will have one major team project. Your grade will be partially determined by other team member’s evaluation of your contribution. I will assign you to teams randomly. Your team will be expected to produce an initial proposal by the sixth week, a refined plan by the eighth week, and a finished product during the final exam period. The team project should consist of a fully-functioning program that does something useful or fun, accompanied by user-level documentation. The deliverables should include:
   a. Proposal and plan presentations, in the form of PowerPoint presentations, or equivalent Open-Office files.
   b. The source code and associated support files (icons, images, etc.), delivered as an Eclipse package.
   c. A printable user document in PDF format, explaining how to install, configure, and use your program.
   d. An executable file, .jar or .exe that constitutes your program.
Team projects will be presented on the last two class meetings.

GRADING SCALE:

At the end of the quarter, your points will be added up, and grades will be assigned as follows:

<table>
<thead>
<tr>
<th>Grade</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>A: [94,100]%</td>
<td>A-: [88,94)%</td>
</tr>
<tr>
<td>C+: [75,79)%</td>
<td>C: [72,75)%</td>
</tr>
</tbody>
</table>
**ATTENDANCE:**

Your attendance is closely linked to your performance. If you have to miss a class or two, you should be responsible enough to use the PowerPoint notes I publish, the book, your “study buddies” in class, and (as a last resort), my office hours to catch up what you missed. If you can’t get to class on time, or must leave early, please observe the following rules: don’t do it often, and do your darnest to not disturb the class as you are coming or going.

**INCOMPLETES:**

“Incomplete” grades will only be available to students who were otherwise succeeding (C or better, 75% of the coursework completed), but were unable to complete the course due to documented extenuating circumstances. A signed “incomplete contract” will be necessary, and the course grade will revert to an “E” (or other grade as agreed in the contract) if work is not completed by the contract date.

**ACADEMIC DISHONESTY:**

This gets complicated in a programming course. Programming is frequently a team activity, and I expect you to consult with other students on your weekly programming assignments. In fact, you will get extra credit for helping other students on the forum. But I expect your turn-ins to be your own.

Another complication that developed recently is that some students have figured out how to get assignments done (with the aid Google, they say), without ever gaining the ability to program independently. Being able to program is the most important outcome of this class. If I suspect that you don’t understand the code that you’ve submitted, I reserve the right to require you to be interviewed by me and expected to explain your code. If you can’t make a satisfactory explanation of the code you submitted, your submission may be down-graded and/or treated as an incident of academic dishonesty and reported appropriately.

I use JPlag, a system developed at Karlsruhe University in Germany that finds similarities among multiple sets of source code files to detect software plagiarism. JPlag does not merely compare bytes of text, but is aware of programming language syntax and program structure and hence is robust against many kinds of attempts to disguise similarities between plagiarized files. JPlag is very good at detecting program copying. It is also very good at not falsely accusing students; I have not had a single “false positive” accusation from it in several years use. All the grades I post for programming assignments are tentative until I perform a JPlag run. A part of the JPlag report is a histogram of code-overlap. Outliers on the “high overlap” side are clear and unmistakable. Make sure yours is not one of them!

Overlap of 90% or greater is clear evidence of copying. Both submissions will get 0 points for the assignment, unless you can make a very good case that there was not copying, with appropriate evidence. No grades are final until the JPlag run has been made, and I will postpone runs until straggling assignments get in (late turn-ins seem to have a higher problem…). Do not share your code with other students!

“As stated in the TCC catalog, ‘Students are expected to be honest and forthright in their academic endeavors. Cheating, plagiarism, fabrication or other forms of academic dishonesty corrupt the learning process and threaten the educational environment for all students.”

The complete Administrative Process for Academic Dishonesty is available on the TCC website at: www.tacomacc.edu/stuonline/policies/start.shtm

I will not condone cheating or other forms of academic dishonesty. Cheating, and classroom behaviors reasonably associated with cheating (including test scores grossly inconsistent with programming assignment grades), are a violation of the student disciplinary code and are punishable through the disciplinary procedures of the college. If I find you cheating or suspect that you have been cheating, your assignment/exam will be voided. Instructors have an obligation to report incidents of cheating, or suspected cheating, in accordance with the procedures outlined in the Student Rights and Responsibilities Code.
STUDENT ACCESS/ACCOMODATION

Students with Special Needs: Students are responsible for all requirements of the class, but the way they meet these requirements may vary. If you need specific auxiliary aids or services due to a disability, please contact the Access Services office in Building 18 (253-566-5328). They will require you to present formal, written documentation of your disability from an appropriate professional. When this step has been completed, arrangements will be made for you to receive reasonable auxiliary aids or services. The disability accommodation documentation prepared by Access Services must be given to me before the accommodation is needed so that appropriate arrangements can be made.

EXTRA HELP:

You are encouraged to make an appointment with the instructor or meet during office hours for any further clarification. But your best source of extra help is likely to be online forums in which you can ask questions of other students and experts. There is vibrant online community of people learning Java and helping others learn it.

CLASSROOM POLICIES:

My hearing is not too good (I wear hearing aids). Sometimes there may be classroom noise that bothers you, but that I simply can’t hear. Therefore I’m asking for your help. If any classroom behavior of your classmates ever gets in the way of your learning, please let me know, in class or after class. All too often, we think that enforcing classroom decorum is a matter of instructor’s pride and self-importance. Well, not so in my case. As much as I like to hear myself talk, this is really about your learning. You are paying a lot for it (in time at least, if not money), and you deserve to get full value. And I’ve noticed that one word from a fellow student is ten times as effective as an instructor begging for students to quiet down and pay attention. I’ll try to do my part, but I’m asking your help too.

Use of a Cell phone is not allowed inside the classroom. Unless you have good reason to expect an emergency call, please turn your cell phone off for the duration of the class. If you are expecting an emergency call, set your phone on vibrate, and quietly exit the class to handle any call you receive.

Caffeinated beverages and light munchies sometimes aid in learning programming. Just remember: the janitor never finds out! (And no computer is harmed in the process) 😊

ETIQUETTE FOR CLASSROOM DISPUTE RESOLUTION:

If you have questions or concerns about this class or me, please come to talk with me first. If we are unable to resolve your concerns, you may talk next with the Chair of the Engineering Department, Dr. Rebecca Sliger. The Chair can assist with information about additional steps, if needed.

FINAL EXAM:

The day of the final exam will be used to present group projects. All students are expected to participate in the assessment of:

1. The projects assembled by other project teams.
2. The contributions of other members of their team.

SYLLABUS CHANGES:

This syllabus is a contract between the student and the instructor. Nevertheless, sometimes (rarely) this contract may need to be changed. If it is, the change will be announced in class and posted on the class web site.
**TENTATIVE SCHEDULE:**

Expect changes in the following schedule, but with many chapters to, we need to keep up a pace of about one chapter per week. This syllabus and schedule are subject to change in the event of extenuating circumstances. If you are absent from class, it is your responsibility to check on announcements made while you were absent.

<table>
<thead>
<tr>
<th>Week</th>
<th>Chapters, topics</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Jan 02 - 03</td>
<td>Syllabus, setup, etc.</td>
</tr>
<tr>
<td>2 Jan 06 - 10</td>
<td>Chapter 11 - Inheritance</td>
</tr>
<tr>
<td>3 Jan 13-17</td>
<td>Chapter 11 – Inheritance (Quiz 1 on Ch. 11 is Friday the 17th)</td>
</tr>
<tr>
<td>4 Jan 21 – 24 (Note: Mon, Jan 20 is MLK day, no class)</td>
<td>Chapter 12 – Exceptions</td>
</tr>
<tr>
<td>5 Jan 27-31</td>
<td>Chapter 13 – Advanced GUI applications (Quiz 2 on Ch. 12&amp;13 is Friday the 24th)</td>
</tr>
<tr>
<td>6 Feb 3 - 7 (Note, Thu, Feb 6th is Ed. Planning Day. Extra credit for attending Dr. Sliger’s quarterly presentation)</td>
<td>Chapter 14 – Applets and graphics</td>
</tr>
<tr>
<td>7 Feb 10 - 14</td>
<td>Chapter 14 – Applets and graphics</td>
</tr>
<tr>
<td>8 Feb 18 – 21 (Note: Mon, Feb 17 is President’s day, no class)</td>
<td>Chapter 15 – Recursion (Quiz 3 on Ch. 14&amp;15 is Friday the 21st)</td>
</tr>
<tr>
<td>9 Feb 14 - 28</td>
<td>Chapter 16 – Sorting, Searching, and Algorithm Analysis</td>
</tr>
<tr>
<td>10 Mar 03 - 07</td>
<td>Chapter 17 – Generics</td>
</tr>
<tr>
<td>11 Mar 10 - 13</td>
<td>Chapter 18 – Collections</td>
</tr>
<tr>
<td>11 – 12 Fri Mar 14 &amp; Mon Mar 17</td>
<td>Term Project Team presentations</td>
</tr>
<tr>
<td>12 Wed, Mar 19</td>
<td>Final Exam, covering mainly chapters 17&amp;18</td>
</tr>
</tbody>
</table>