



THE UNIVERSITY *of*
NEW MEXICO

CS 351 Design of Large Programs

Spring 2009

Course Instructor:

Joel Castellanos

e-mail: joel@unm.edu

Office:

Room 321 of the Farris Engineering Center (FEC).

Located near the northeast corner of Central Avenue and University Boulevard. This is building #119 in section I-2 of the campus map.

Office Hours:

Mon & Wed: 10:30-11:30 AM

Tues & Thurs: 8:00-9:00 AM and by appointment

Lab Instructor:

David Godinez

e-mail: dgodinez@cs.unm.edu

Office: FEC 301A

Office Hours: Mondays: 10:00AM - noon

Textbook:

Throughout the semester, various required readings will be placed on electronic reserve at the Library.

Description:

This class is about designing big software, where big refers to projects with a scope too large to be handled by any one person at any one time. This course primarily deals with software design, time management, and strategies for completing complex coding tasks.

Programming Language and Environment:

All programming examples and assignments will use Java and the Eclipse Integrated Development Environment (IDE). As the final project will require fully integrated teamwork, it is important that all students be comfortable a common, integrated environment.



Grading:

- 50% Programming Projects. Three individual, one group.
- 20% Exams (midterm and final)
- 15% laboratory quizzes and class programming assignments. These assignments are designed to take only about 40 minutes. You will be given until midnight to turn them in, but you should not need to spend more than a few minutes outside of class to finish up.
- 15% Lecture quizzes (i-clicker), group project presentations, and group project presentation evaluations.

Late projects/assignments will receive a 5% per day penalty.

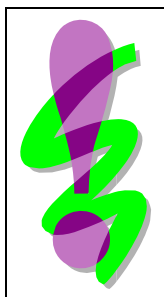
Working Together:

Working together and helping one another on all projects (but not on exams and quizzes) is highly encouraged. This includes discussion of project *specification*, *algorithms*, *data structures*, and *test cases*. It does not include code. Each person (or group) must author his or her own code.

Cheating:

Cheating will be dealt with very harshly, and includes:

- Copying code from another person or having someone else write your code.
- Copying code from the Internet or another source. (If there's some code that you would really, really like to use, please check with us before you do it.)
- Attempting to disassemble, decompile, or otherwise reverse engineer compiled example programs.
- Allowing another person to copy your code.
- Leaving your code (paper or electronic copies) where others can find it. You responsible for the security of your intellectual property.
- Use of an external Java library (i.e., libraries other than those included with the JDK) without documenting it. *Note: If you do document usages of external libraries, it will not be considered cheating. However, you still might not receive full marks if the library covers too much of the assignment. It is best to check with one of the instructors before using an external library.*
- Violation copyright or license agreements on external libraries. If you use external library code, it is your responsibility to understand and comply with the appropriate copyright and license issues.
- Violation the University [policy on acceptable computer use](#).



Not being able to explain how some significant part of your code works will result in a zero for the assignment. It does not matter if the reason you do not understand your code is because you did not do the work or because you got your code working by trial and error. If I suspect someone of cheating, the first thing I do is ask that person to explain the code. This is not a quiz you ever want fail. Too much code in the real world is build and maintained by trial and error. It makes for a house of cards, and is exactly what this class about preventing.



Submitting Assignments:

All assignments must be in WebCT in order to receive credit for them. If WebCT is down, then you can e-mail the assignment to the lab instructor in order to prove it was done on time. However, it must be inside WebCT before you can receive credit for it.

When submitting an assignment in WebCT, submit a single .JAR file. This file must include all source files and resource files used to build the executable. Do not include class files unless they are third party libraries. Include your name in the class name of whichever of your source files contains the main function that you want run during grading. For example: JoelCastellanos-Snowflake.java.

Syllabus:

Week 1

- JUnit testing
- Project 1: Scientific Modeling
This project is the implementation of a science based model. Externally visible classes must follow a strict *interface specification document* and must compile and run with a pre-existing (instructor written) Graphical User Interface (GUI). JUnit testing must be used. This will be a three week project with milestones.

Week 2

- JUnit testing for exceptions
- Subtleties of Java argument passing.

Week 3

- Java Interfaces, and Generics
- Detailed examination of *static* in Java

Week 4

- Refactoring
- Design Patterns (Singleton Pattern)
- Project 1: Finished

Week 5

- Project 2: Genetic Algorithm applied to Image Processing
- Image processing with Java2D

Week 6

- Design Patterns (Adapter Pattern)

Week 7

- Multithreaded programming in Java

Week 8

- Design Patterns (Command Pattern)
- Project 2: Finished.

Week 9

- Spring Break



Week 10

- Review & Midterm exam.

Week 11

- Project 3: Agent Based Simulation

This will be group project and will be significant enough to require a group of talented programmers to get it done within six weeks. The project will address a very specific case of a common real-world problem. The topic will be such that without detailed design, success will be nigh impossible. The project will be many faceted including:

- Research general problem and what other programmers have done in the field. Professional programmers build on existing work as much as is possible (and legal). Unlike most undergraduate programming assignments, you will be following professional practices in this.
 - Design.
 - Data collection.
 - Data representation.
 - Objects with complex and well defined interactions.
 - Graphical user input.
 - Graphical display of data, model dynamics and results.
 - Unit testing
 - Integration testing (does your code do what you say it does?)
 - Model Validation (do changes in your model's data produce changes in the model's outcomes that correspond to the real-world?)
- Examples of agent based simulations
 - Student led discussion of the use of Java verses C/C++ for large programs

Week 12

- Aspects of Extreme Programming
- Student PowerPoint (or OpenOffice) presentations: Design Reviews

Week 13

- Design Patterns (Observer Pattern)
- Student presentations: Design Reviews

Week 14

- Design Patterns (Strategy Pattern)
- Student presentations: Code Reviews

Week 15

- Design Patterns (Bridge Pattern)
- Student presentations: Code Reviews

Week 16

- Advanced topic
- Student presentations: Results - external guests will be invited who might be expected to have some interest the real-world issue being modeled.

Final Exam (cumulative)