CS 320 Spring 2011
COMPUTER GRAPHICS SYLLABUS

Betsy Williams Sanders
420 Olendorf
williamsb@rhodes.edu

January 13, 2011

1 Topics

This course is about *computer graphics*. In this course, computer graphics means the procedural or programmatic use of a computer to create or manipulate images. We'll first talk about what images are, then we'll learn how to manipulate them, but the majority of the course will be spent on generating images.

The course requirements consist of five programming projects. There are no tests. The programming projects are quite intensive, and so a working knowledge of C++ or C is required.

2 Textbook

The textbook for this course is *Fundamentals of Computer Graphics*, 3rd ed., Peter Shirley and Steve Marschner, A.K. Peters, ISBN 978-1-56881-469-8. This is an excellent book on the modern fundamentals of graphics. However, the material we cover in this course is taught from a more signal-processing approach to graphics than the book. As a result, we jump around in the book quite a bit, and often material is covered in class that is somewhat different from the book. Thus, often there won't be material in the book discussing the particulars of the class lecture or the projects. I will point out what sections of the book are relevant whey they occur.

3 Grading

Grades are computed on a point basis. Each project is given a maximum number of points. The score you receive on the project is recorded and the total summed over the quarter. Typically, each assignment will be worth approximately 50 points. Grades will either be evaluated as the best of a strict percentage basis ($90+\%\rightarrow$ A, 80-90%$\rightarrow$ B, etc.) or by determing the modes of the class grade distribution and assigning letter grades to each mode.

Assignments will be handed out in class, and are also available in my public folder.

4 Assignments Due

The due dates for assignments are strictly enforced. The assignments are due at midnight on the deadline day, usually two or three weeks (depending on the assignment) after the assignment is given.

4.1 Late Policy

The late policy works as follows: You have five (5) "floating" days that you can use to turn in assignments late. Weekends, holidays, power outages, and other natural disasters count in these floating days. So, for example, let's say

that a project is due by midnight on Thursday, February 4, 2011. If you turn it in at 12:01 a.m. on Friday you've used up one of your floating days. If you turn it in at 11:59 p.m. on the same Friday, you've still only used up one of your floating days. If you turn it in at 12:01 a.m. on Saturday, you've used up two floating days.

Projects can be turned in late as long as you have an appropriate number of floating days remaining for the time amount indicated. The exception is the last project, which must be turned in by 5:30pm on May 3 whether you have multiple floating days remaining or not. If you use up all your floating days, then projects must be turned in on time. If they're turned in late, they will have 50% automatically deducted from them, OR they won't be counted at all, completely at the discretion of myself.

USE YOUR FLOATING DAYS WISELY! It's usually not a good idea to use floating days because you didn't allocate enough time for the project, i.e., you started the project Sunday night and it was due on Tuesday. The projects take time to code – I'm giving you plenty of time to do them. Floating days are best used when you have 2 assignments due and a test on the same day, and you need an extra day to do some final checks and modifications.

If truly terrible unforeseen circumstances arise you might be able to arrange an extension with me. Giving an extension is completely dependent on what my workload is like at the time — don't expect one and don't take it for granted.

5 Assignments

I will be *running* your assignments in order to test them. You will be turning in files for each assignment comprising your source code and your test data. Details on how to turn in the assignments will be provided later.

Each assignment should contain everything needed to compile and test your program. DO NOT make links to files located in other places. Keep these directories around and the files in them until after your assignment has been graded. You don't need to copy over executables as your program will be remade before testing. Please provide either a Makefile or a shell script which will build your executable from scratch.

All of your projects require you to provide test data of some sort. Part of your project's grade will be based on how thorough your test data is. The test files should be named something completely obvious like `test1.dat`, `test2.dat`, etc. You can also provide a file named README in case you wish to provide special instructions or comments before I attempt to run your program. You will **lose** 5 points if you don't provide any test data.

Note that any program which crashes while being tested *automatically* loses 5 points. Thus, you should pay special attention in checking for strange input or handling internal errors before they propagate. Also, note that your program should pretty much work as advertised. If it doesn't, I won't be spending too much time trying to find out why when I'm grading it.

6 Collaboration

I encourage collaboration as long as you're learning something. But, when it gets to the point that you are collaborating to save work at the expense of learning, stop.

This means that, in the early stages of thinking about an assignment, it's good to discuss the project with your fellow students and to learn about various library functions and algorithms together. You can discuss algorithms and data structures in a general way.

As work progresses on an assignment you should collaborate less and less. For example, you aren't allowed to COPY source code or specific data structures. You can certainly look at someone else's source code. But if you have it in front of you while you're typing into the computer, or if you just edit a copy of their file, then you're stepping over the line. It's easy to see when a program has been copied—the variable names may have been changed but they have exactly the same structure and do exactly the same things; the control structures all go exactly the same way, etc.

To summarize, you can work together at a high level but I insist that each one of you write your own programs. *The Rhodes Honor Code governs all work in this course.*

7 Office Hours and Recitation

My office is 420 Olendorf. My office extension is 3791. Email is usually the best way to reach me. Office hours will be from 1:30 p.m. to 3 p.m. on Tuesday/Thursday, or by appointment. However, you are free to drop by any time. If I have some free time, I'll be happy to help you, or we can schedule an appointment. There will be information about this course in my public folder.

Additionally, there will be a recitation session I will usually conduct for this class at a time to be determined. If there's someone there and questions are asked, I'll answer them until there are no more questions (or the hour gets late). If there's no one there with 5 to 10 minutes, I'll leave. My intent with these sessions is to answer questions about the assignments in more detail than I may have time to go into them during class, clear up confusing points, and generally answer a question once to everyone rather than ten times to ten different people.

As mentioned earlier, this course requires an inordinate amount of programming. If you find yourself having problems with the coding or compiling of end of the assignments, as opposed to the theory and algorithms end, then you're probably in over your head.

8 Special Accommodation

If you are in need of special accommodations, please register with the Office of Student Disability Services (http://www.rhodes.edu/disabi as soon as possible so that all necessary arrangements can be made.

9 Important Dates

Drop/Add Period Ends: 1/19/2011 Extended Drop Period Ends: 2/2/2011 Withdrawal Period Ends: 3/25/2011