Computer Science 210

Spring, 2014

Instructor: Ray Morehead, MD

717 Engineering Sciences Building

293-9129

The structure of this course is rather unique, so it will require some adjustment on your part. Read this syllabus over quite carefully, as there are several unique aspects to the requirements that are listed here.

Office Hours: Before or after class.   MWF 1-3.   I am often in the office at other times and you are welcome to drop in anytime, or email me to set a time. I encourage you to make frequent use of office visits, no matter how trivial you may think your question is. This is a very hands on course and I am very committed to making every student in this class a good programmer. If you are having problems with your code – if you are finding it difficult to make it work or to debug it, it's probably because your approach is faulty.

Don't sit on issues for too long as you can churn up a lot of time with a bad approach to a problem. Ask for help. The next time you have a similar issue, chances are you will know how to deal with it.

Email: ray.morehead@gmail.com .   Email is a great way to ask questions if they are simple – for example, if you have an error in your code that has you stumped.   The BEST time to ask a question is in class so we can talk over the issue in detail and you can get a clear understanding. *I generally will not answer complicated or conceptual questions by email, because its easier to discuss in class, so don't be offended if I respond to your email with a suggestion that we talk about it in class.* If you're not sure if a question is appropriate, you are better off asking!

Text: ***Clean Code: A Handbook of Agile Software Craftsmanship*** by Robert C. Martin. I read this book just a year ago. This guy NAILS what this course is about so I definitely would try to read it early in the semester.   It's easy reading and isn't too too long. You can get a free e-version on the web thru the University library (Safari books). You won't be tested on it and we won't cover it directly in class but I think it's definitely worth your time. Every semester students tell me how much of a help it was.

Java SDK: 7.0

Eclipse Integrated Development Environment (http://www.eclipse.org) .. Version 4.0/4.3 is preferred Several additional plugins will be required as described in the lab.

Lab: This course is a project oriented course. There will be 9 deliverables. See class discussion and notes. All classes will be held in 756. I try to kill as little time as possible lecturing. There will be a lot of lectures early in the semester, and these tend to dwindle out as the semester ensues and you get the drift of the requirements. I start lecturing PROMPTLY at the beginning of class and a lot of days my entire lecture is done in 10 minutes or so. I tend to restrict my lecture material to IMPORTANT stuff that will directly impact the project, and I DON'T like to repeat myself because you were late or in bed. So show up for class on time.

Attendance: The attendance policy may seem tough to some of you. I simply want you to actually show up for class everyday (my, my! so hard!) and I will penalize you if you don't. It is obvious to me (mainly through personal, painful experience) that it is ABSOLUTELY imperative that you attend class regularly if you are to keep your head in the game and accomplish the goals of this class. I get the argument that "I can do this at home" occasionally, and I am pretty sure it isn't true. When people stop coming to class, they stop learning and stop participating. Therefore, attendance will be taken every day. There are no "excused absences". Each missed class will result in a deduction in the point score that you accumulate in this course. It's really hard for me to judge whether or not your class absence was "legitimate", and so I avoid that discussion – you're either present or your not. If, for some reason, you absolutely cannot attend a class, you can make it up by finishing another module.

Makeup policy: I know that some students have unavoidable absences, so I do allow makeups by attending the other class. Because I think developing good class attendance habits is important, I penalize you for makeups. You are allowed to makeup a class by attending **three** lectures or **two** labs of the other section.

Grading: I believe that as you progress through this course, you will find that you cannot avoid learning a huge amount about Java and about programming in general. This course is about concepts and process, which I don't think can be expressed or measured very well by testing. Therefore there will be no tests in this course. Since grading is a necessary component of college life, I have developed a system of grading that is as transparent as possible so that you have a clear idea of where you stand at all times. The course is broken up into 9 modules, and each module is sequential, meaning that you must successfully complete one module before going on to the next.

Please note that this course is really about process and not about destination. Whether or not you finish module or all 9 modules is really not the point of this course and is really immaterial in the final analysis; this course is about learning and applying complex and advanced concepts. I have developed the module grading system so that students have a clear idea of expectations of this course, but I enforce the assimilation of concepts by insisting that you perform well on each module. Procrastination is a major enemy in this course. Unlike in many courses, where procrastination is subtly rewarded, in this course it will cause you great harm. The name of the game in CS210 is to attack the workload early and aggressively. Nevertheless, each student will necessarily work at their own pace. The grading strategy offers incentives for early completion and penalties for late submissions.

This course is different than most CS courses. By now, you have taken programming courses and have submitted programming assignments. Your program may have worked reasonably well but not perfectly but you submitted it and you were done with it. You received a grade based on whatever arbitrary criteria the grader came up with. This course is different. When you are done with a module, you will submit it for review. I will test your code against the specs and conduct a code review. Based on my assessment, your assignment will either receive an ACCEPT or a RESUBMIT. There will be no grade attached. As a MINIMAL level of acceptance, your code must function perfectly. I will generally list for you the problems I found with either function or code. If you receive a RESUBMIT, you will be able to correct the issues I raised and submit your code repeatedly . If your code has a lot of issues, once you fix all of them, other issues may become apparent on subsequent submissions. Many students have some trouble initially with the concept of making their code run bugfree!

YOUR FINAL GRADE IN THIS COURSE WILL BE AT THE DISCRETION OF THE INSTRUCTOR. A point based rubric is supplied, so you have a clear and transparent idea of what is required in this course. I, the instructor, reserve the right to override the rubric at any time (although overrides are uncommon and for specific reasons). The course is challenging, and, because I believe the skills acquired in this course are critical for success in Computer Science, I am not reluctant to fail students that I feel have not acquired a set of basic skills.

CHEATING: This course has an open structure, and there are no “secrets” to this course. If you don't know how to do something, just ask, and I am OK with you asking other students for their advice. Ultimately, if you write up the modules and make them work, even if you get help you will still learn a gigantic amount about programming. However, wholesale copying of code and other acts of cheating destroys the moral fabric of the course. It helps neither the copier nor the source, and it can generally create great ill will and anger among other students and is inconsistent with the University Honor Code. Therefore, this course has a zero tolerance policy towards cheating. If I find evidence of cheating you get an F in the course and whatever other punishment the University rules allow.

TOPICS:

1. Introduction

2. Class tools: Eclipse, subversion
3. Inheritance and polymorphism
4. Code Reuse
5. Regular Expressions
6. Java Collections Review and Practical Application
7. XML
8. Binary Files
9. Advanced Java Collections and Generics
10. Serialization
11. Binary search trees. Implementation in Java.

Other Class Policies

1. The final due date for all submissions is precisely Midnight on Friday of the last week in the course. Under no circumstances will ANY submissions be accepted beyond that date.

2. The Laboratory Fee is not refundable after the first week of classes. A registration restriction for the succeeding semester will be imposed if the fee is not paid.

3. Students may not audit nor may they "sit in for free."

4. West Virginia University is committed to social justice. I concur with that commitment and expect to foster a nurturing learning environment based upon open communication, mutual respect, and nondiscrimination.

Our University does not discriminate on the basis of race, sex, age, disability, veteran status, religion,

sexual orientation, color or national origin. Any suggestions to further such a positive and open environment in this class will be appreciated and given serious consideration. If you area person with a disability and anticipate needing any type of accommodation in order to participate in this class,

please advise me of the same and make appropriate arrangements with Disability Services (2936700).