CSC 372: Comparative Programming Languages
The University of Arizona
Spring 2015

Instructor

William H. Mitchell (whm)
Hours and contact info: *Posted on Piazza*

Teaching Assistants

None

Website

We'll be using Piazza for announcements, discussions, and more. **If you haven't already signed up, do it now!** Go to piazza.com and follow their instructions.

Course Objectives

The purpose of this course is to explore some alternative ways of specifying computation and to help you understand and harness the forces that a programming language can exert. We'll spend a lot of time in the trenches with three languages: Haskell, Ruby, and Prolog. Functional programming will be studied using Haskell. Ruby will be used to explore imperative and object-oriented programing using a language with dynamic type checking. Prolog will transport us into the very different world of logic programming. You'll learn about some interesting elements of other languages.

Upon successfully completing the course you'll be around a "2" on a 1-5 (high) scale with Haskell, Ruby, and Prolog. You will understand the characteristics of the programming paradigms supported by those languages and be able to apply some of the techniques in other languages. You'll have an increased ability to learn new languages. You'll have some idea about whether you want to pursue further study of programming languages.

Prerequisites

CSC 127B or CSC 227

The prerequisites are very modest but this is a 300-level computer science class with a lot of diverse content. As an analogy, imagine a 300-level math class with only high-school algebra as a prerequisite but that will venture into some exotic math that's not much like algebra at all. My task is to respect the prerequisites but also cover a body of material that's appropriate for a 300-level CS class.

Textbooks

No textbooks are required.

It is my intention that the lectures, handouts, and Piazza postings will provide all the information needed to successfully complete the course; but it's often good to see things explained in multiple ways. Below are some books that I recommend as good supplements.

Haskell:

Learn You a Haskell for Great Good!, by Miran Lipovača

Real World Haskell, by O'Sullivan, Stewart, and Goerzen

Programing in Haskell, by Hutton

All three are available on Safari.

Ruby:

There are numerous Ruby books on Safari but they don't have the Ruby book I currently like the best, which is this:

Programming Ruby 1.9 & 2.0 (4th edition): The Pragmatic Programmers' Guide
pragprog.com/book/ruby4/programming-ruby-1-9-2-0

The Ruby book on Safari that I currently like the best is *The Ruby Programming Language* by David Flannagan and Yukihiro Matsumoto.

Prolog:

Safari has zero Prolog titles but one of the Prolog books I really like is freely available on the net, albeit as pages scanned as images:

Prolog Programming in Depth, by Covington, Nute, and Vellino
www.covingtoninnovations.com/books/PPID.pdf

I'll make available on Piazza an OCR'd version of this book with a hidden text layer, so it can be searched.

An excellent Prolog text that I've asked the UA bookstore to stock is this:

Programming in Prolog, 5th edition, by Clocksin and Mellish

Note that Safari can be accessed from off-campus using the VPN or by using the library's proxy. Here's the proxy-based URL for Safari: proquest.safaribooksonline.com.ezproxy1.library.arizona.edu/

Grading Structure

The final grade will comprise the following:

Assignments	60%
Quizzes	5%
Mid-term exam	13%
Final exam	22%

Final grades will be based on a ten-point scale: 90 or better is a guaranteed A, 80 or better is a B, and so forth. The lower bounds may be adjusted downwards to accommodate clustering of grades and/or other factors.

It is my goal that you will need to spend, on average, no more than ten hours per week, counting lectures, to learn the course material and get an "A" in this course. If you find that you're needing to spend more time than that, let's talk about it.

You are strongly encouraged to contest any assigned score that you feel is not equitable.

There is no attendance component in the grade—if you find that my lectures aren't worth your time, feel free to cut class!

My goal is for everyone to earn an "A" in this course.

The mid-term exam is tentatively scheduled for Thursday, March 12, the last class before Spring Break. It will cover all the Haskell material and some amount of Ruby.

See registrar.arizona.edu/schedules/finals.htm for the date and time of the final exam.

Assignments

Homework assignments will largely consist of programming problems but other types of problems, such as short answer questions, essay questions, and diagrams, may appear as well. There will be a video project, too. It is anticipated that there will be seven major assignments and possibly a few minor assignments. There will be a total of 600 points worth of assignments. Based on 600 total assignment points, a ten point homework problem corresponds to one point on your final average.

For programming problems great emphasis will be placed on the ability to deliver code whose output exactly matches the specification. Failing to achieve that will typically result in large point deductions, sometimes the full value of the problem.

My view is that it's a Bad Thing to give any credit for code that doesn't work. **Programs that don't compile or that mostly don't work will earn a grade of zero, regardless of how close to working they might be.** Additionally, non-general solutions, which might have the expected output "wired-in", will very likely earn a grade of zero.

Unless specifically requested in an assignment write-up, no comments are required in any code. Use comments as you see fit.

My view is that programming assignments are to help you learn the course material—I don't view an assignment as a take-home exam. As a rule, you'll learn more if you can get through an assignment without asking for a lot of help; but if you reach a point where you simply aren't making progress and you're running out of ideas or time, then you surely should ask for help.

Each assignment will specify a precise due date and time. **As a rule, late assignment submissions are not accepted and result in a grade of zero. There are no late days.**

Extensions may be granted to the class as a whole if problems arise with an assignment or with departmental or university computing facilities.

Extensions for individuals may be granted in certain cases. **The key factor that leads to an extension is that due to circumstances beyond the student's control, the student's work is, was, or will be impeded, and it is impractical or impossible for the student to make up for the lost time.**

Accident, illness, friend/family crises, and significantly disruptive failures of technology are examples of circumstances that I generally consider to be beyond a student's control. On the other hand, for example, an extension due to lots of work being due in other classes is extremely unlikely. Travel, such as an interviewing trip, may merit an extension, but pre-trip discussion and approval of the extension is required. **Ultimately, however, each situation is unique; you are strongly encouraged to contact me if you believe an extension may be warranted.** If you believe an extension is warranted, DO NOT work on an assignment (or even think about it) past the deadline; wait for an extension to be granted.

Extensions are granted in the form of an amount of time, such as eight hours. An eight-hour extension can be pictured as a count-down timer with an initial setting of eight hours. The timer runs whenever you are working on the assignment, whether that be typing in code or simply thinking about it. You will be on your honor to keep track of the time spent during an extension and not exceed the amount granted.

All holidays or special events observed by organized religions will be honored for those students who show affiliation with that particular religion. Absences pre-approved by the UA Dean of Students (or Dean's designee) will be honored.

Bug Bounties

A "bug bounty" of one assignment point of extra-credit will be awarded to the first student to report a particular bug in an assignment. Bugs might take the form of errors in examples, ambiguous statements, incomplete specifications, etc. As a rule, simple misspellings and minor typographical errors won't qualify for a bug bounty point; but each situation is unique—you are encouraged to report any bugs you find. Any number of bug bounty points may be earned for an assignment and will be added to the grade for that assignment.

Bug bounty points may also awarded for bugs in the slides, my Piazza postings, quizzes, exams, and this syllabus. Such points are added to the next assignment.

Quizzes

There will be some number of "pop" quizzes. Quizzes will typically be a handful of questions, be allocated three minutes or less, and be worth 1-5 quiz points. There will be a total of 50 quiz points, corresponding to 5% of the final grade. Quizzes may be conducted at any time during the class period. In some cases a quiz may be given simply to see what portion of the class grasped some just-presented material and be graded on the basis of participation rather than correctness.

Computing Facilities

There are various options for working with the software we'll be using: (1) Use systems maintained by the CS department, either remotely or in the CS labs. (2) Install the required software on your own machine. (3) Use the OSCR labs (uits.arizona.edu/departments/oscr/locations/hours).

Even if you plan to work on your own machine, go ahead and IMMEDIATELY get a CS computing account if you don't have one already. Details on opening an account can be found at cs.arizona.edu/computing/accounts/accts-key.html.

The CS computing facilities are described in cs.arizona.edu/computing/facilities. The FAQs at faq.cs.arizona.edu have lots of answers but particularly useful is *How do I use the Remote Access servers?*, at faq.cs.arizona.edu/index.php?solution_id=1014.

Office Hours

I truly enjoy working with students. I believe that interaction with students outside the classroom is a vital aspect of teaching a course. I will do everything possible to make myself accessible to you.

I prefer to conduct office hours in a group-style, round-robin manner. You needn't wait in the hallway if I'm working with another student; you may join us and listen in if you so desire. If several persons each have questions, I will handle one question at a time from each person in turn. I will often give priority to short questions (i.e., questions with short answers) and to persons having other commitments that constrain their waiting time. (Speak up when you fall in either of these categories.) If for some reason you would like to speak with me in private, let me know; I will clear the office. Or make an appointment to meet with me outside of office hours.

Students who make proactive, not reactive, use of office hours usually achieve the best results. Proactive use of office hours includes asking questions about material on the slides and in the texts, asking questions about how to better use tools, discussing how to approach problems, etc. If you're familiar with Covey's *The Seven Habits of Highly Effective People* you might recognize those as "Quadrant II" activities—important, but not urgent.

Reactive use of office hours is typically centered around simply getting code to work one way or another (Covey's "Quadrant I"—important and urgent).

Piazza

As mentioned above, we'll be using Piazza. All students are strongly encouraged to participate freely. I hope you’ll post questions and comments related to the course material, recommendations for handy tools, URLs for interesting web posts and videos, and whatever else you think is worth mentioning as long as it relates to the course and/or programming languages.

The initial post in any discussion thread that I start is considered to be part of the course material—you are responsible for reading each and every one, and taking action when appropriate. For example, I might ask you to read an article, or experiment with a web site of interest. You may learn more by reading follow-ups by classmates and by me, but at exam- or quiz-time I won't expect you to have read each and every follow-up.

When answering questions I will give priority to well-focused questions. And, it's often the case that the task of developing a well-focused question will lead you to an answer on your own.

Needless to say, BE CAREFUL that Piazza posts don't give away the solution for a problem. If at all in doubt, make it a private post to "Instructors". A typical penalty for "blowing" a problem is that the guilty student will be required to create an equally great problem as a replacement.

If a private post asks a question or raises a point that I think should be shared with the class, I'll share it. If I think the post deserves kudos, I'll identify the author unless the post specifically requests anonymity, in which case I'll say something like "A student wrote..."

You can make anonymous posts, but be aware that such posts are only anonymous to classmates, not me.

Part of the idea of Piazza is to facilitate students helping each other, so don't hesitate to answer questions when you're so inclined. If a post is plain wrong, I'll add a correction. If a post is great, I'll endorse it. If I'm silent, then its quality is probably somewhere in the middle.

If you post a question and later solve it yourself, save everybody some time by saying the question has been resolved.

Instant Messaging

IM can be very effective, especially for short questions. Skype is my IM tool of choice. My id on Skype and other popular IM services is `x77686d`. If an IM session starts to run long I'll often suggest adding in voice and maybe http://join.me, too. Don't pay much attention to my Skype status—sometimes I'm not available when I'm "Online", and at other times I might be invisible but happy to answer. I recommend you start with an "ayt" ("Are you there?") and then follow with your question if I respond.

It's hard to characterize what's best posted on Piazza versus asking on IM so I won't wrestle with that question here.

Original Thoughts

In the movie comedy "Broadcast News" a 14 year-old high-school valedictorian receives a post-commencement beating from a group of bullies. After picking himself up, one of the things he shouts at his attackers is, "You'll never have an original thought!" That notion of an "original thought" has stayed with me. I hope that you'll have some original thoughts during this semester.

I offer an award of a half-point on your final average for each Original Thought that you claim as such and that strikes me as significant. Observations, analogies, quotable quotes, and clever uses of tools and language constructs are some examples of things that have qualified as Original Thoughts in my classes. Note that an Original Thought does not need to be something that's probably never been thought of before; it just needs to be something that I consider to be reasonably original for you.

Of course, an Original Thought needs to be something you've thought up yourself—don't send in something you found elsewhere, like a quote, just because it strikes you as being original!

Academic Integrity

It is unfortunate that this section need be included but experience sadly shows that some students are willing to sacrifice their integrity to obtain a grade they have not earned. For those students who would never do such a thing, I apologize for the inclusion of this section.

Capsule summary: Don't cheat in my class. Don't make it possible for anybody else to cheat. One strike and you're out!

You are responsible for understanding and complying with the University's Code of Academic Integrity. It can be found at deanofstudents.arizona.edu/codeofacademicintegrity. Among other provisions, the Code demands that the work you submit is your own and that submitted work will not subsequently be tampered with. Copying of another student's programs or data is prohibited when they are part of an assignment; it is immaterial whether the copying is by computer, photograph or other means. Allowing such copying, thus facilitating academic dishonesty, is also a Code violation.

In addition to ruining one's grade and damaging one's future, the processing of an academic integrity case requires hours of work by myself and others. I am happy to spend hours helping a student who is earnestly trying to learn the material, but I truly loathe every minute spent on academic integrity cases. As a data point, I'll mention that a relatively open-and-shut case in the Fall of 2012 required ten hours of my time, not to mention the time of several other faculty and staff members.

A violation of the Code will typically result in ALL of the following:

1. **Assignment of failing grade for the course**
2. **A permanent transcript annotation: "FAILING GRADE ASSIGNED DUE TO CHEATING"**
3. **Disallowance of GRO for the failing grade**
4. **Recommendation of a one-semester suspension from the University**

It is difficult to concisely and completely describe where reasonable collaboration stops and cheating begins, but here are some guidelines:

- It is surely cheating to submit code or text that was not written by you. Exception: If I work you through any part of a solution, either individually in a group setting, you may freely use any code developed in that process. However, you may not pass along that code to anyone else.

- It is surely cheating to give another student any portion of a solution.

- I consider it to be reasonable to work together on assignments to get to the point of understanding the problems and the language elements that are required for solutions.

- I consider it to be reasonable to help another student find a bug if (1) you've finished that problem, and (2) your help consists of asking questions that help the other student to see the problem for themselves.

- I consider it to be reasonable to exchange test cases unless test cases are one of the deliverables for a problem.

- If you receive help on a solution but are unable to fully explain how it works, it is surely a mistake to submit it as your own work.

- If your gut feeling is that you're cheating on an assignment or helping somebody else cheat, you probably are.

- **The vast resources of the Internet raise some interesting issues. See the section below on *Google, Stack Overflow, and more*.**

- If in the heat of the moment you submit a solution that is not fully yours, or give your work away, and you later reconsider your actions, you'll at worst lose points for the work involved if you confess before your act is discovered. Conversely, further dishonesty when confronted, which invariably increases the time expenditure, raises the likelihood of more extensive penalties.

Cheating almost always starts when one student has easy access to the solutions of another. You are expected to take whatever steps are necessary to guard your solutions from others in the class. For example, you should have an unguessable, uncommon password and never share it. Hardcopy should not go into a recycling bin—take it home and dump it in a recycle bin when the course is over. Personal machines, be them laptops or home desktops, should be behind a hardware firewall or running a software firewall.

Failure to take reasonable precautions to ensure the privacy of your solutions may be construed to be facilitating dishonesty, a Code violation. For example, having a weak password such as, but not limited to, your first name, your last name, your phone number, your initials, your sweetie's name, your pet's name, a reversed name, a sequence of consecutive keys, or a common password could be viewed as facilitation of dishonesty. For one list of common passwords see www.openwall.com/passwords/wordlists/password-2011.lst

Leaving a logged-in and unlocked machine unattended in the presence of another person, even a friend, is questionable. Use a screen locker!

Be very careful when typing a password in the presence of others, be them friends or strangers. The single worst cheating case I've ever handled started when the password of a two-fingered typist was surreptitiously observed. **Don't hesitate to ask someone, even a friend, to turn their head when you're typing your password.**

Google, Stack Overflow, and more

Things like Google and Stack Overflow are incredible resources for working professionals but when used to directly search for solutions for problems on an academic assignment they can cause learning and/or creative opportunities to be forever lost.

My expectation is that the material presented in class, practice via exercises, suggested readings, resources cited, and any prior assignments provide everything you need to do every problem on each assignment. I challenge you to limit your Googling to searches that expand your knowledge of the material, and not try to dig up quick solutions to problems. (And, of course, using any code found in such searches would be cheating.)

In this class it is forbidden to post problem-specific questions on websites, IRC channels, mailing lists, etc. ANY SUCH POST WILL BE CONSIDERED TO BE CHEATING. A problem-specific question is a question that poses any portion of a problem or question from an assignment and asks for help, advice, etc. on how to solve it. Here is an example of a problem-specific question: "*I'm learning Haskell and trying to write a function that returns True iff the parentheses in a string are properly matched. Any suggestions?*"

Accessibility and Accommodations

It is the University's goal that learning experiences be as accessible as possible. If you anticipate or experience physical or academic barriers based on disability, please let me know immediately so that we can discuss options. You are also welcome to contact Disability Resources (520-621-3268) to establish reasonable accommodations.

Please be aware that the accessible table and chairs in this room should remain available for students who find that standard classroom seating is not usable.

Threatening Behavior

Threatening behavior is not tolerated. University policies on threatening behavior can be found at policy.web.arizona.edu/threatening-behavior-students.