CSc 372: Comparative Programming Languages
Fall 2006

Instructor

Name:	William H. Mitchell
E-mail:	`whm` (on the CS machines)
Campus office:	Gould-Simpson 923A; 626-8470
Office hours:	*Posted on website*
Home office phone:	577-6431 (8 a.m.-10:30 p.m., seven days a week)

Teaching Assistant

Name:	Poornananda Gaddehosur
E-mail:	`poorna` (on the CS machines)
Office hours:	*Posted on website*

Prerequisites

CSc 127B or CSc 227.

Course Objectives

The purpose of this course is to teach you 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: ML, Ruby, and Prolog. Functional programming will be studied using ML. Ruby will be used to explore the mind set of a "scripting language." 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 will be moderately fluent in ML, 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.

Website

The class website is `www.cs.arizona.edu/classes/cs372/fall06`.

Textbooks

It is my intention that the lectures, handouts, mailing list postings, and materials on the class website will provide all the information needed to successfully complete the course. The textbooks should be thought of as suggested references for the languages we will be studying. I will cite readings in the texts to supplement material presented in lectures, but it is my intention to avoid situations that will negatively impact a student who does not have a copy of the texts. For example, there will be no open-book exams, and there will be no homework problems that simply cite a page and problem number in one of the books.

The three texts I suggest are these:

Elements of ML Programming, ML97 Edition (2nd Edition) by Jeffrey D. Ullman

Programming Ruby: The Pragmatic Programmers' Guide, Second Edition, by Dave Thomas, Chad Fowler, and Andy Hunt

Prolog Programming in Depth by Michael A. Covington, Donald Nute, and Andre Vellino

We'll be covering ML, Ruby, and Prolog in that order, with almost four weeks on each, so you might delay purchase of the Ruby and/or Prolog books if you so desire. Note that the Ruby book is available as a PDF at `www.pragmaticprogrammer.com/titles/ruby`.

Grading Structure

The composition of the final grade is as follows:

Assignments and quizzes	64%
Mid-term examination	14%
Final examination	22%

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

It is my goal that you will need to spend 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 spending more time than that, let's talk about it.

As its name implies, the mid-term exam will be given somewhere around the middle of the semester, most likely after we complete the material on Ruby. The exact date will be determined by progress through the material, assignment due dates, and possibly other factors. At least two week's notice will be provided. The final examination will be on Thursday, December 14, 11:00 a.m. to 1:00 p.m.

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!

Assignments

Homework assignments will largely consist of programming problems but other types of problems, such as short answer questions and hand-drawn diagrams, may appear as well. It is anticipated that there will be six major assignments and possibly a few minor assignments. There will be a total of 640 points worth of assignments (and quizzes—see below). Based on 640 total 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 typically have the expected output "wired-in", will very likely earn a

grade of zero.

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 homework assignment will specify a precise due date and time; typically the stroke of midnight on Thursday nights. (Use `www.time.gov` or some other reliable time source to be sure you know what time it really is.) **As a rule, late homework solutions, even if only one second late, are not accepted and result in a grade of zero.**

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, and friend/family crises 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 is warranted. If you believe an extension is warranted, DO NOT work on an assignment (or even think about it) past the deadline; wait until an extension is 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 that time 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 homework 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 are also awarded for bugs in the slides, e-mail messages, and this syllabus. Such points are added to the next homework assignment.

Quizzes

I will conduct some number of unscheduled short quizzes. Quizzes will typically be a question or two, be allocated five minutes or less, and be worth between one and five homework points (0.1 to 0.5% of the final grade). Quizzes may be conducted at any time during the class period. In some cases a quiz may be conducted simply to see what portion of the class grasped some just-presented material. In such cases simply taking the

quiz will earn full credit.

It is anticipated that about 50 of the 640 "homework" points will in fact be quiz points, but that number (50) may vary.

If circumstances beyond your control (as outlined above for assignments) cause you to miss a quiz, let me know.

Computing Facilities

At present, there are two options for working with the various languages we'll be using: (1) Login to `lectura` (a Linux machine) via `ssh` and work with them there. (2) Install the language systems on your own machine—the versions of the languages on `lectura` are freely available for Windows as well as Linux. Option (1) will require you to learn a few things about Linux but we'll be happy to help you with that.

If for some reason you don't already have a departmental computing account (which provides access to `lectura` and the Windows machines) apply for one IMMEDIATELY. Details on opening an account can be found at `www.cs.arizona.edu/computing/accounts/accts-key.html`.

Office Hours

I truly enjoy working with students whether it is in person, via e-mail, IM, or the phone. 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.

For the time being I'll be holding office hours in my office, GS 923A but if there's interest, I'll consider holding office hours in a lab, such as GS 228. I welcome your input on this possibility.

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 and I will clear the office.

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 use tools, discussing how to approach problems, etc. If you're familiar with Covey's *The Seven Habits of Highly Effective People* you might view those as "Quadrant II" activities—important, but not urgent.

Reactive use of office hours is typically centered around simply getting a program to work one way or another ("Quadrant I"—important and urgent) rather than viewing it as an exercise to put new concepts and techniques into practice.

E-Mail

E-mail can be a very effective way to resolve a variety of questions but instead of mailing directly to me or Poorna, **please use the address `cs372-staff`**, which in turn is forwarded to me and Poorna. Mailing to `cs372-staff` ensures that your message gets attention as soon as possible. If you respond to a response, please "follow the CCs". For example, if I respond to your message and CC to Poorna, your response (if any) should also CC to Poorna.

Our goal is to answer mail promptly, overnight at worst. If you think a message has been overlooked, please resend it.

When answering mail, priority is given 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.

If you mail us with a problem and then solve it yourself before either of us replies, please send a follow-up note to let us know that the problem is resolved and that we needn't respond to your original note.

The first step we usually take when a problem is reported is to attempt to reproduce the problem. A common mistake is to send only a "relevant" excerpt of the code being developed when in fact additional code is needed to reproduce the problem. **When reporting a problem, be sure to include enough material so that we can reproduce the problem.** For example, if you are seeing an odd result from `sml`, send a text clipping that shows both the expression you typed and the output produced. If the code you're having trouble with is in a file, send the full contents of the file, either as an attachment or in the body of the message. (I strongly prefer the latter.)

As a rule it's not a good idea to put a long list of questions in a single message—we'll answer the message as a whole; the time required to address the full list may delay our response.

If you mail from an address that is not at a subdomain of `arizona.edu`, you may receive a note from my spam-blocker, TMDA, asking you to "Please confirm your message to whm". Replying to that note will put your address on my "whitelist", and you won't be asked again to confirm unless you use a different address. (Let me know if you receive more than one confirmation request.)

I aim to use my `cs.arizona.edu` address on outgoing class-related mail but you may see my mail coming from `msweng.com` and `MitchellSoftwareEngineering.com`, too.

Instant Messaging

IM can be very effective, especially for short questions. Skype is my IM tool of choice. My id is `x77686d`. If you have a Skype account and tell me the id, I'll occasionally use that if some back and forth is needed to more quickly resolve a question you've sent by e-mail.

Phone Calls

The telephone can be an excellent way to resolve complex questions but students often hesitate to call my home office for fear they will be disturbing me. That concern is completely unfounded—if I don't want to bother with a phone call, I simply don't answer the phone. It's often the case that a five minute phone call can accomplish the same result as an e-mail message that takes fifteen minutes to write and ten minutes to respond to.

The best time to catch me in is between 8:00 and 10:30 in the evenings but you'll often have luck with daytime calls on Mondays, Wednesdays, Fridays, Saturdays, and Sundays.

I am happy to return phone calls to you or answer an e-mail with a phone call if that's what you'd like. If you give me your phone number, state the latest time when I can call you. If not specified, I'll feel free to call you as late as 10 p.m. but not after.

Mailing List

A mailing list, rather than a `cs.` news group, will be used as the electronic forum for this class. Messages from me might include information such as supplemental material, assignment corrections, lecture clarifications, due-

date changes, etc. **The material I post to the list is considered to be part of the course material** so it is important that you be on the list.

Please don't view the list as read-only! I hope you'll participate with questions and comments related to the course material, testimonials for handy tools, URLs for interesting websites or blogs, and the like.

I looked at several options for a mailing list and my choice, at least for the moment, is a Google Group. The group's name is `cs372-ua`. Go to `groups.google.com/group/cs372-ua/about` to join the group or simply send me the e-mail address(es) you'd like to have on the list. You do not need to have a Google sign-in to receive messages but you do to view the archives, adjust your subscription, etc.

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 spears he casts 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 and 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 `dos.web.arizona.edu/uapolicies/cai1.html`. 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, Xerox 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.

A violation of the Code will typically result in all of the following: (1) failure of the course, (2) the following permanent transcript annotation: "FAILING GRADE ASSIGNED DUE TO CHEATING, and (3) disallowance of GRO for the failing grade.

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

- I consider it to be reasonable to work together on assignments to get to the point of understanding the problem and the facilities that are required for the solution.

- I consider it to be reasonable to help another student find a bug only if your solution for that problem is further along than the one being examined. For example, if you haven't started on a problem, you shouldn't be hunting bugs in another student's code for that problem.

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

- It is surely a mistake to give another student a copy of a solution or a significant portion thereof.

- 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.

- 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 action, you stand an excellent chance of great leniency 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, including a recommendation for suspension or expulsion from the university.

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 a non-guessable 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 like ZoneAlarm or XP's ICF. If you and another student share a computing facility off-campus, perhaps as co-workers or roommates, using that system to develop solutions may be of very questionable merit, depending on the security facility on the system and the privileges of the other student.

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 initials, your pet's name, a team mascot, or your CSID could be viewed as facilitation of dishonesty. (See www.openwall.com/passwords/wordlists/password.lst for one list of common passwords.)

Leaving a logged-in and unlocked machine unattended in the presence of another person, even a friend, is very questionable. If password-based security on a personal system, such as your laptop, can be bypassed, perhaps by booting from a floppy, then leaving that machine unattended is questionable.

It would be great if login authentication on lab systems was handled biometrically or with a hardware key of some sort. Until that day comes, however, you must **be very careful when typing your 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.**

Accommodation

Students with disabilities, who may require academic adjustments or reasonable accommodations in order to participate fully in course activities or to meet course requirements, must first register with the Disability Resource Center (drc.arizona.edu). DRC staff will qualify students for services, and provide a letter to the instructor listing accommodations to be made. This letter should be submitted by the student directly to the instructor as soon as possible during the first week of classes. The student should meet as soon as possible with the instructor by appointment or during office hours to discuss accommodations and how course requirements and activities may impact the student's ability to fully participate.

Threatening Behavior

Threatening behavior is not tolerated. University policies on threatening behavior can be found at policy.web.arizona.edu/~policy/threaten.shtml.