SYLLABUS
CSC 372—Comparative Programming Languages
The University of Arizona

GS 906, MW 9:30-10:45
Fall 2022

Description of Course

Catalog Description: Introduction to several major high-level programming languages and their characteristics. Programming projects are required in at least three languages.

Course Prerequisites

Major admission; CSC 210

Because CSC 120 is required for major admission, working knowledge of Python and Java is assumed.

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.

Instructor and Contact Information

Instructor: William H. Mitchell

Email: whm

Office: Gould-Simpson 854

Office hours are on the 372 Google Calendar

Discord: whm#5716 (ok to try 24/7)

Skype: x77686d (ok to try 24/7)

Mobile phone: 520-870-6488

Ok to call 9am to 10pm, seven days a week, minus Sunday mornings.

If no answer, send email or try Discord or Skype. Don't leave voice mail!

Thanks.

Graduate Teaching Assistant: H. M. Abdul Fattah

Email: hmfattah

Office: Gould-Simpson 749, Desk 5

Office hours are on the 372 Google Calendar

Contact information: TBD

Graduate Teaching Assistant: Amy Paul

Email: amypaul

Office: TBD

Office hours are on the 372 Google Calendar

Contact information: TBD

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; just come on in and take a seat! If several students have questions, I'll typically work for 5-7 minutes with each student in turn.

If for some reason you would like to speak with me in private, let me know. 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 working for an assignment (Covey's "Quadrant I"—important and urgent).

Website

The course website is https://www2.cs.arizona.edu/classes/cs372/fall22.

Piazza

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

D2L

D2L will be used for posting grades.

Obtaining Help

- **Academic advising:** If you have questions about your academic progress this semester, or your chosen degree program, consider contacting your department's academic advisor(s). Your academic advisor and the Advising Resource Center can guide you toward university resources to help you succeed. **Computer Science major students**

are encouraged to email advising@cs.arizona.edu for academic advising related questions.

- **CS Tutor Center:** The Department of Computer Science offers FREE tutoring for students enrolled in CSC courses. You can view tutor schedules and sign up for tutoring sessions by visiting our CS Tutoring Page.
- **CS Help Desk:** The Computer Science IT team can help students with department technology issues including logging into/resetting your Lectura account, printing in the 930 lab, etc. You can submit a ticket for help by visiting the Computer Science Lab Helpdesk (note, requires UA login).
- **Life challenges:** If you are experiencing unexpected barriers to your success in your courses, please note the Dean of Students Office is a central support resource for all students and may be helpful. The Dean of Students Office can be reached at 520-621-2057 or DOS-deanofstudents@email.arizona.edu.
- **Physical and mental-health challenges**: If you are facing physical or mental health challenges this semester, please note that Campus Health provides quality medical and mental health care. For medical appointments, call (520-621-9202. For After Hours care, call (520) 570-7898. For the Counseling & Psych Services (CAPS) 24/7 hotline, call (520) 621-3334.

Class Recordings

I currently plan to record lectures but that may change over the course of the semester. I encourage students to remind me to start recording if I fail to announce that recording has commenced.

Recordings will typically capture my laptop screen/Elmo/whiteboard plus my microphone. Regarding the audio track, my practice is to call on students as Mr./Ms. *last-name*, and those names will be captured on the audio track. If you prefer me to use just your last name or no name at all, express that preference via email.

Don't hesitate to remind me to repeat questions from classmates so that the question is captured on the recording.

Required statement per CS department policy:

> For lecture recordings, which are used at the discretion of the instructor, students must access content in D2L only. Students may not modify content or re-use content for any purpose other than personal educational reasons. All recordings are subject to government and university regulations. Therefore, students accessing unauthorized recordings or using them in a manner inconsistent with UArizona values and educational policies (Code of Academic Integrity and the Student Code of Conduct) are also subject to civil action.

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 most of our time working with three languages: Haskell, Racket, and Prolog. Functional programming will be studied using Haskell. We'll see that Racket, a dialect of Lisp, is a

chameleon of a language with little syntax, interesting semantics, and plenty of parentheses. Prolog will transport us into the very different world of logic programming. Depending on the Racket segment goes, we may spend a few lectures exploring Ruby and contrasting it with Python. We'll perhaps spend a day each on SNOBOL4 and Icon. You'll learn about some interesting elements of other languages.

Expected Learning Outcomes

Upon successfully completing the course you'll be around a "2.5" on a 1-5 knowledge scale (5 high) with Haskell, Racket, 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, perhaps by taking CSC 453.

Absence and Class Participation Policy

Attendance is expected but will not be recorded. Students are, however, fully responsible for all material presented during lectures. Class attendance is strongly recommended.

Students who appear to be chronically tardy may be asked to meet with me to discuss ways to improve their punctuality.

If you miss a graded activity for circumstances beyond your control or your absence is excused by university policy, I'll be happy to meet with you and discuss ways to recover any lost points.

The UA's policy concerning Class Attendance, Participation, and Administrative Drops is available at https://catalog.arizona.edu/policy/class-attendance-and-participation.

The UA policy regarding absences for any sincerely held religious belief, observance or practice will be accommodated where reasonable: http://policy.arizona.edu/human-resources/religious-accommodation-policy.

Absences pre-approved by the UA Dean of Students (or dean's designee) will be honored. See https://deanofstudents.arizona.edu/policies/attendance-policies-and-practices

Illnesses and Emergencies

- If you feel sick, or may have been in contact with someone who is infectious, stay home. Except for seeking medical care, avoid contact with others and do not travel.
- Notify your instructor(s) if you will be missing up to one week of course meetings and/or assignment deadlines.
- If you must miss the equivalent of more than one week of class and have an emergency, the Dean of Students is the proper office to contact (DOS-deanofstudents@email.arizona.edu). The Dean of Students considers the following as qualified emergencies: the birth of a child, mental health hospitalization, domestic violence matter, house fire, hospitalization for physical health (concussion/emergency surgery/coma/COVID-19 complications/ICU), death of immediate family, Title IX matters, etc.
- Please understand that there is no guarantee of an extension when you are absent from class and/or miss a deadline.

Statement on compliance with COVID-19 mitigation guidelines:

As we enter the semester, our health and safety remain the university's highest priority. To protect the health of everyone in this class, students are required to follow the university guidelines on COVID-19 mitigation. Please visit www.covid19.arizona.edu.

Makeup Policy for Students Who Register Late

Late registrants will be handled on a case-by-case basis depending on the date of registration. In general, if you want to take my class, I'm happy to have you, and I'll do what I can to get you on board.

Course Communications

Important: If I need to contact you for any reason, I will consider sending mail to *your-NetID*`@arizona.edu` address to be a guaranteed way to contact you in a timely fashion.

Email

For private communication with the TA(s) and/or me, mail to `372f22@cs...`. To ensure quality and accuracy, I want to see all mail between students and TA(s). If you're replying to the TA(s)' response to a question, use your mailer's "Reply All" to follow the Cc:'s.

If an email message 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 message specifically requests anonymity, in which case I'll say something like "A student wrote..."

Instant Messaging

IM can be very effective, especially for short questions. At present I mostly use Discord but I've still got my Skype account, too. My usernames can be found in the contact information.

Don't pay much attention to my IM 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?") or such, and then follow with your question if I respond.

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 things on the web, and whatever else you think is worth mentioning as long as it relates to the course material in some way. Posts about CS-related job opportunities, and events like programming contests and hackathons are welcome, too.

The initial post in any Piazza discussion thread that a TA or 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 on Piazza, course staff 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, use email instead. A common case for a post that gives away a solution is when a student is very close to a solution but doesn't realize it, perhaps because a tiny piece is missing. Haskell and Prolog solutions are often very short, sometimes just a line or two, so posting any code or describing any elements of a solution is very risky.

Piazza's "Private posts"—posts seen only by course staff—are disabled. Mail to `372f22@cs` instead.

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, we'll add a correction if time permits. 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 (or email) a question and later solve it yourself, BE SURE to post (or email) a follow-up saying the question has been resolved. If you don't, we'll needlessly spend time answering your question. In turn, that might lead to us not having enough time to answer a classmate's question. In a worst case that might cost a classmate a letter grade, and even make a pass/fail difference!

You can make anonymous posts, but be aware that such posts are only anonymous to classmates, not to course staff. Personally, I strongly discourage anonymous posts! Along with learning, college is a great place to make connections, and I believe that letting your name be known helps you build your reputation and your network, which is all-important when you go to look for job. If you're worried about looking stupid or silly, have a chat with your ego and get over it--an important skill for the workplace is to not be afraid to ask questions, and there's usually no anonymity in the workplace.

Required Texts or Readings

No textbooks are required.

It is my intention that the lectures, handouts, Piazza postings, and various no-cost materials on the web will provide all the information needed to successfully complete the course.

Handouts

A large portion of the course material will be presented using PowerPoint slides. Those slides will be available on the course website in the form of both .pptx files and PDFs. Students will be provided handouts with 4-up, two-sided copies of all slides, in installments. Selected answers/solutions will be elided from the handouts but will be present in versions on the web.

If you don't want handouts, just let us know by mailing to 372f22@cs. You can change your preference at any time.

Supplemental Books

It's often good to see things explained in multiple ways. Below are some books that I

recommend as good supplements but it's important to understand that the routes we'll be taking through the languages don't directly correspond to any book; the slides are the primary "text".

I hope that every CS student is aware of learning.oreilly.com. Formerly called O'Reilly Safari, it is a fabulous collection of thousands of software-related books and more. Some of your tuition dollars go towards paying for it! At places below I'll refer to the collection as "O'Reilly".

Haskell books

- I think that *Learn You a Haskell for Great Good!* by Miran Lipovača is the best beginner's book on Haskell. It's on O'Reilly, too.
- *Real World Haskell* by O'Sullivan, Stewart, and Goerzen, has some good introductory material as well as some interesting real-world examples.
- If you don't mind spending a little money to buy a Haskell book, I really like *Haskell: The Craft of Functional Programming, 3rd edition*, by Simon Thompson. It's very thorough; it doesn't make the sort of leaps that can leave beginners puzzled.
- I think of *Programming in Haskell* by Hutton to be a good book once you've got the basics down, but I think it moves far too fast to stand alone as a first book on Haskell unless one has had prior exposure to functional programming. It used to be available via O'Reilly but is no longer there.

Racket books

I worked with Lisp many years ago in graduate school and I still use Emacs Lisp but Racket is a new language for me. I'm continuing to explore the literature.

- Racket is based on Scheme, and *The Scheme Programming Language* by R. Kent Dybvig is by far the best Scheme book I've yet found. The fourth edition is the latest.
- *Teach Yourself Scheme in Fixnum Days* is still on my to-read list, but is freely available.
- *Beautiful Racket* by Matthew Butterick centers on the idea of "language-oriented programming". It's very interesting, but perhaps not well-aligned with the path through Racket that I'm currently picturing.
- Dr. Collberg really likes *The Little Schemer* by Daniel P. Friedman. I like it less than he does, but some readers enjoy its dialog-based approach.
- *Structure and Interpretation of Computer Programs* by Abelson and Sussman is a classic text on Scheme. Sussman is one of the creators of Scheme. For a number of years this book was used in the introductory CS class at MIT. More than a few study groups of professional programmers have formed to study this book.

Ruby books

As mentioned above, I don't yet know how much Ruby, if any, we'll study. In past semesters, when my 372 line-up was Haskell, Ruby, Prolog, I recommended these books:

- O'Reilly has numerous Ruby books but it doesn't have the Ruby book I like the best, which is Programming Ruby 1.9 & 2.0 (4th edition): The Pragmatic Programmers' Guide
- The Ruby book on O'Reilly that I currently like the best is The Ruby Programming Language by David Flannagan and Yukihiro Matsumoto.

- I also like The Ruby Way, 3e by Hal Fulton and Andre Arko.

Prolog books

- O'Reilly has zero Prolog titles but an excellent text, Programming in Prolog, 5th edition, by Clocksin and Mellish, is available as a PDF that can be downloaded through the library. I recommend that you get a copy NOW, in case licensing agreements change during the semester.
- Another Prolog book I really like is freely available on the net, albeit as pages scanned as images: *Prolog Programming in Depth*, by Covington, Nute, and Vellino. In fact, this book is one of the best books I know of on any programming language.

  I'll make available on the course website an OCR'd version of Covington's book with a hidden text layer added, so it can be searched.

Assignments and Examinations

Exams

There will be two examinations: a midterm and a final.

The midterm exam will cover some general material, all of the Haskell material, and some of the Racket material. I'll want to give the midterm after the first or second Racket assignment is due. Taking all things into account my best guess for the midterm is Monday, October 24, but that could easily vary by a week either way. I'll be sure to let you know, both in class and on Piazza, when a good date for the midterm starts to become clear to me.

See the **Final Examination** section below for information on the final exam.

Quizzes

There will be "pop" quizzes from time to time. Quizzes will typically be a handful of questions displayed on-screen, be allocated three minutes or less, and be worth 1-5 quiz points. Quizzes may be conducted at any time during the class period. In some cases a quiz may be graded simply on the basis of participation rather than correctness.

Assignments

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. We may have some question sets on D2L or openclass.ai. We may do some in-lecture activities that will be counted as assignment grades. There will be a video project, too. As I write this I anticipate that there will be eleven assignments including the video project. Larger assignments will be typically assigned about two weeks before they are due; smaller assignments will have shorter "fuses".

For programming problems great emphasis will be placed on the ability to deliver code whose output exactly matches the specification. Failure 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, solutions that are non-general, with expected output hard-coded, for example, will 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.

Each assignment will specify an exact 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.

FAQs and Corrections for Assignments

I don't like the idea of students needing to dig through dozens of Piazza posts and follow-ups to resolve questions about an assignment. Instead, on the course website there will be a *FAQs and Corrections* page for each assignment that addresses whatever turn out to be good or common questions, along with any corrections that are needed

If everybody can get in the habit of checking the FAQs and Corrections before posting on Piazza, that'll raise the signal-to-noise ratio on Piazza.

To reduce clutter in the FAQs and Corrections, errors like typos, wrong fonts, and doubled words that don't affect meaning will be fixed on the spot and not mentioned in the list of corrections. Errors of consequence will be enumerated. A Git repo will be used to record, at minimum, the initially posted version of each assignment.

Extensions on Assignments

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 assignments or exams 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 strongly recommended. Unexpected hours at a job, such as needing to fill in for a sick co-worker, may warrant an extension. 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.

Here's a scenario involving an extension:

> *Your new laptop catches fire and burns itself into a cinder eight hours before a midnight deadline on a Wednesday. I would likely grant you an eight-hour extension. On Friday, your next chance to get to the store, you get a new laptop and you're operational by Friday night. You might spend four hours Friday night working on the assignment, take off all day on Saturday and Sunday, spend two more hours on Monday and get absolutely stuck, and finish it up after a couple of questions during office hours on Wednesday.*

If given an extension, you'll be required to keep written track of the time spent and not exceed the amount granted.

Computing Facilities

Your solutions for programming assignments will be graded on the CS machine named "lectura". Be sure to test them there!

By virtue of being enrolled in this class you should already have a CS computing account with the same name as your UA NetID but with a password that's likely different. With that account you can login on any of the CS instructional machines, including lectura. The files in your home directory tree are stored on a server and appear the same no matter which CS machine you're logged into.

My UNIX slides from 352 in Fall 2015 have details about logging into lectura, resetting your password, and more. Those slides are here. See slides 14-21 for details about logging in. Slides 74-102 discuss options for editing files directly on lectura, and SFTP clients like WinSCP and Cyberduck, to keep directories on lectura synchronized with corresponding directories on your laptop.

However, a very simple thing is to use remote editing, as shown with Cyberduck on slide 90 in those old 352 slides. With remote editing, the client, such as Cyberduck, copies data from your file on lectura to a temporary file on your machine, opens that temporary file with an editor of your choice on your machine, and then when you save, file contents are copied back to that file on lectura.

Some editors have remote editing built-in but beware of plugins! There was once a remote-editing plug-in for Sublime that would occasionally get one's file contents mixed up with someone else's file on lectura!

The CS computing facilities used to be described in http://cs.arizona.edu/computing/facilities but that link is now bad; I haven't yet determined if there's a current version of that document. I do know that the FAQs at http://faq.cs.arizona.edu have lots of answers.

Final Examination

My reading of the Fall 2022 Final Examination Schedule, under *Exam Schedule for MWF, 4/5 Day, & MWF Daily Classes* seems to specify that our final exam will be on Thursday, December 15, from 10:30am-12:30pm. (Let me know if you interpret the schedule differently!) The exam will be in our regular classroom. The final exam will be comprehensive but with more emphasis on post-midterm material.

On that same page, see also *Policy Memo: Final Examination Regulations and Information.*

Grading Scale and Policies

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

The final grade will comprise the following:

Assignments	60%
Pop quizzes	5%
Mid-term exam	13%
Final exam	22%

Final grades will be based on a ten-point scale: 90.0 or better is a guaranteed A, 80.0 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.

Each problem on an assignment will be assigned an exact number of *assignment points*. Ideally, the sum of the assignment points for all of the semester's assignments and in-lecture activities will be 600, making each assignment point correspond to 0.1% of the final grade. If for some reason the total number of assignment points is not exactly 600, assignment points will be scaled. For example, if a total 625 points of problems appear on assignments and in-class activities, each assignment point will correspond to about 0.096% of the final grade.

Similarly, a total of 50 quiz points will ideally be assigned, making each quiz point worth 0.1% of the final grade. If that ideal isn't met, quiz points will be scaled.

The TA(s) and my goal will be to get each assignment graded before the next assignment is due. Our goal will be to get the midterm exam graded in a week. The final exam will be graded in accordance with deadlines for final grade submission.

You are strongly encouraged to contest any assigned score that you feel is not fair. The absolute deadline for contesting a score is the moment your final exam begins but the sooner the better. Otherwise you may be faced with an incomplete, depending on the staff's end-of-semester workload.

Department of Computer Science Grading Policy:

1. Instructors will explicitly promise when every assignment and exam will be graded and returned to students. These promised dates will appear in the syllabus, associated with the corresponding due dates and exam dates.
2. Graded homework will be returned before the next homework is due.
3. Exams will be returned "promptly", as defined by the instructor (and as promised in the syllabus).
4. Grading delays beyond promised return-by dates will be announced as soon as possible with an explanation for the delay.

Incomplete (I) or Withdrawal (W):

Requests for incomplete (I) or withdrawal (W) must be made in accordance with University policies, which are available at http://catalog.arizona.edu/policy/grades-and-grading-system#incomplete and http://catalog.arizona.edu/policy/grades-and-grading-system#Withdrawal respectively.

Extra Credit Opportunities

Throughout the semester there will be a number of opportunities to earn an assignment point or two of extra credit. For example, simply providing a good estimate of the time spent on an assignment will be worth one assignment point.

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

Please report bugs by mail or IM, not via Piazza, to give us a chance to filter any false positives.

Don't interrupt lectures to point out minor bugs on slides such as typos but do speak up if an error seems serious or potentially confusing.

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 to wound his departing 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. 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.

Sometimes I'll point out that something you said in class or office hours, or wrote in your observations for an assignment, strikes me as an Original Thought. If you self-identify a potential Original Thought, let me know. In some cases I'll see a glimmer of an Original Thought in something, and encourage you to explore the idea a bit more.

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!

The "bar" rises with each Original Thought for an individual—it's harder to earn a second Original Thought than a first, and a third is harder still.

Honors Credit

Students wishing to contract this course for Honors Credit should email me to set up an appointment to discuss the terms of a possible contact. The Honors Course Contract Request Form is available at http://www.honors.arizona.edu/honors-contracts.

Scheduled Topics/Activities

I last taught this course in 2018. About a third of that course was on Ruby, as an example of a dynamically typed language. However, students this semester will be coming in with knowledge of Python in hand, and for that reason, I'm going to cover Racket, as an example of a Lisp, and perhaps only do a little Ruby, or maybe none. Additionally, I might also cut back on the amount of time we'll spend on Haskell. For core topics, we'll definitely start with Haskell and end with Prolog, but the middle is full of unknowns.

With that in mind here is a tentative schedule for the semester. The number of assignments and their scheduling is particularly s*ubject to change.*

Week #	Week of	Topics	Assignments Due / Exams
1	Aug 22	Syllabus, basic questions about programming languages, UA's programming language heritage, the idea of a paradigm, programming paradigms, value/type/side-effect	a1 (survey), due Aug 27 at 11pm; graded by Sep 7
2	Aug 29	The functional paradigm, Haskell basics, getting and running Haskell (ghci), functions and function types, type classes, writing simple functions, functions with multiple arguments, partial application	
3	Sep 5	Functions as values, type inferencing, type specifications, guards if-else, a little recursion, list basics, lists of lists, strings, cons lists, a little output, patterns	Language questions
4	Sep12	Patterns in functions, patterns vs. guards vs. if-else, recursive functions on lists, tuples, more on patterns and functions, the where clause, the layout rule, larger examples	

5	Sep 19	Errors, debugging, excursion into infinite lists and lazy evaluation, higher-order functions, map, sections, filtering, composition, point-free style, anonymous functions	Haskell #1
6	Sep 26	Hocus pocus with higher-order functions, syntactic sugar, DIY currying, flip/curry/uncurry et al., folding, user-defined types, a little I/O—sequencing and actions	
7	Oct 3	Racket: introduction, Racket by observation, getting and running Rack, S-expressions, and lots more	Haskell #2
8	Oct 10	More Racket basics	
9	Oct 17	More Racket, maybe getting into Macros	Racket #1
10	Oct 24	Racket, maybe some Ruby	Midterm exam
11	Oct 31	Ruby	Racket #2
12	Nov 7	Getting and running SWI Prolog, atoms, numbers, predicates/terms/structures, more on queries, unification, query evaluation mechanics, rules, instantiation as "return",	
13	Nov 14	Arithmetic, cut, can't prove, recursive predicates, lists	Ruby #1
14	Nov 21	Built-ins for lists, member, append, findall, typing in Prolog	Prolog #1
15	Nov 28	Low-level list processing, "univ", database manipulation, pit crossing puzzle	Prolog #2
16	Dec 5	Brick laying puzzle, the Zebra puzzle, Prolog conclusion, programming language humor, SNOBOL4 and Icon by observation	Prolog #3 and video project due Dec 7; graded by Dec 17

Department of Computer Science Code of Conduct

The Department of Computer Science is committed to providing and maintaining a supportive educational environment for all. We strive to be welcoming and inclusive, respect privacy and

confidentiality, behave respectfully and courteously, and practice intellectual honesty. Disruptive behaviors (such as physical or emotional harassment, dismissive attitudes, and abuse of department resources) will not be tolerated. The complete Code of Conduct is available on our department web site. We expect that you will adhere to this code, as well as the UA Student Code of Conduct, while you are a member of this class.

Classroom Behavior Policy

To foster a positive learning environment, students and instructors have a shared responsibility. We want a safe, welcoming, and inclusive environment where all of us feel comfortable with each other and where we can challenge ourselves to succeed. To that end, our focus is on the tasks at hand and not on extraneous activities (e.g., texting, chatting, reading a newspaper, making phone calls, web surfing, etc.).

You are free to use your laptops and phones during class but only for course-related work, such as experimenting with code, taking notes, getting a picture of the projection screen, etc. Further, to avoid visual distractions for classmates who can see your screen, display of graphics of any sort aside from GUIs, or anything "eye-catching", including but not limited to pictures, should be avoided unless a class-wide activity with such material is in progress. Any student persisting with disruptive or distracting behavior will be asked to leave the classroom, and may be reported to the Dean of Students.

Threatening Behavior Policy

The UA Threatening Behavior by Students Policy prohibits threats of physical harm to any member of the University community, including to oneself. See http://policy.arizona.edu/education-and-student-affairs/threatening-behavior-students.

Notification of Objectionable Materials

This course may contain material of a mature nature, which may include explicit language, Perl code, depictions of nudity, sexual situations, and/or violence. The instructor will provide advance notice when such materials will be used. Students are excused from interacting with such materials, but they are encouraged to speak with the instructor to voice concerns and to provide feedback.

Accessibility and Accommodations

At the University of Arizona, we strive to make learning experiences as accessible as possible. If you anticipate or experience barriers based on disability or pregnancy, please contact the Disability Resource Center (520-621-3268, https://drc.arizona.edu/) to establish reasonable accommodations.

Code of Academic Integrity

Students are encouraged to share intellectual views and discuss freely the principles and applications of course materials. However, graded work/exercises must be the product of independent effort unless otherwise instructed. Students are expected to adhere to the UA Code of Academic Integrity as described in the UA General Catalog. See https://deanofstudents.arizona.edu/student-rights-responsibilities/academic-integrity .

Uploading material from this course to a website other than D2L or Piazza is strictly prohibited and will be considered a violation of the course policy and a violation of the Code of Academic Integrity. Obtaining material associated with this course (or previous offerings of this course) on a site other than D2L or Piazza, such as Chegg, Course Hero, etc. or accessing these sites during a quiz or exam is a violation of the Code of Academic Integrity. Any student determined to have uploaded or accessed material in an unauthorized manner will be reported to the Dean of Students for a Code of Academic Integrity violation, with a

recommended sanction of a failing grade in the course.

Selling class notes and/or other course materials to other students or to a third party for resale is not permitted without the instructor's express written consent. Violations to this and other course rules are subject to the Code of Academic Integrity and may result in course sanctions. Additionally, students who use D2L or UA email to sell or buy these copyrighted materials are subject to Code of Conduct Violations for misuse of student e-mail addresses. This conduct may also constitute copyright infringement.

My Thoughts and Policies on Cheating

The above is mostly required wording on academic integrity. My thoughts and policies follow.

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 cheat, 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. 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.** Even the very simplest of cheating cases often takes 3-4 hours of my time, not to mention the time of office staff, lab staff, and others, sometimes up to the Dean's Office and beyond! In my mind, the time to handle a cheating case is simply unpaid overtime for everybody involved.

In this class, a violation of the Code of Academic Integrity will typically result in ALL of the following sanctions:

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

When a student is caught cheating and I remind them of the above sanctions they often respond by sending me an email message that is hundreds of words long. They usually start by saying that a failing grade is surely sufficient punishment and that they have learned their lesson; they won't cheat again. They go on to say that some employers won't consider hiring a student whose transcript shows evidence of cheating. (True!) Then they talk about how a suspension from the university will cause them to lose scholarships, job offers, visas, and

more. Some mention ill or aging family members who might not live to see them graduate if a suspension is imposed.

If you ever find yourself considering whether cheating is worth the risk, first imagine how the three sanctions above would impact you and your family. I'm willing to go to great lengths to help students learn the course material but if a student cheats, I'm equally willing to impose great penalties.

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 send another student any portion of a solution.
- It is surely cheating to submit code or text that was not written by you. Exception: If a TA or I works you through any part of a solution, either individually or 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.
- I consider it to be reasonable to work together on assignments to get to the point of understanding the problems and the language or library 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. If you find yourself about to dictate code to a classmate, STOP! (Note that occasionally during office hours you may see me or a TA dictate some code to a student. If so, that's because we've decided that's the best way to help the student forward given the full situation at hand. You do not have that prerogative.)
- 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.

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 and any recipients will 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, such as a recommendation for a multiple-semester suspension or outright 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 unguessable, uncommon passwords and never share any of them. 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 sweetheart'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.

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 hunt-and-peck typist was surreptitiously observed. Don't hesitate to ask someone, even a friend, to turn their head when you're typing your password.

For some interesting reading about password choice, see *Unmasked: What 10 million passwords reveal about the people who choose them*

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 will 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.)

Posts on websites, IRC channels, mailing lists, etc. that solicit the answer for a problem or a significant piece thereof will be considered to be cheating. Example:

> "I'm learning Haskell and trying to write a function that returns True iff the parentheses in a string are properly matched. Any suggestions?"

You might think that using a disposable email address, Tor, etc. will make it impossible to connect you with an improper question but I've caught some like-minded students.

Malware is a Federal Crime

Malware, short for malicious software, is any software used to disrupt computer operation, gather sensitive information, or gain access to private computer systems.—Wikipedia

In various situations the TA(s) and I will be running your code. Do not be tempted to slip some malware into your code, even for "harmless fun". Introducing malware into a computer system is a federal crime—see https://sgp.fas.org/crs/misc/97-1025.pdf. I will request permanent expulsion from the university for any student who transmits malware, and recommend that the university notify the FBI for possible further action.

Nondiscrimination and Anti-harassment Policy

The University of Arizona is committed to creating and maintaining an environment free of discrimination. In support of this commitment, the University prohibits discrimination, including harassment and retaliation, based on a protected classification, including race, color, religion, sex, national origin, age, disability, veteran status, sexual orientation, gender identity, or genetic information. For more information, including how to report a concern, please see
http://policy.arizona.edu/human-resources/nondiscrimination-and-anti-harassment-policy

Our classroom is a place where everyone is encouraged to express well-formed opinions and their reasons for those opinions. We also want to create a tolerant and open environment

where such opinions can be expressed without resorting to bullying or discrimination of others.

Additional Resources for Students

UA Academic policies and procedures are available at http://catalog.arizona.edu/policies

Visit the UArizona COVID-19 page for regular updates.

Campus Health
http://www.health.arizona.edu/
Campus Health provides quality medical and mental health care services through virtual and in-person care. Voluntary, free, and convenient COVID-19 testing is available for students on Main Campus. COVID-19 vaccine is available for all students at Campus Health.
Phone: 520-621-9202

Counseling and Psych Services (CAPS)
https://health.arizona.edu/counseling-psych-services
CAPS provides mental health care, including short-term counseling services.
Phone: 520-621-3334

The Dean of Students Office's Student Assistance Program
https://deanofstudents.arizona.edu/support/student-assistance
Student Assistance helps students manage crises, life traumas, and other barriers that impede success. The staff addresses the needs of students who experience issues related to social adjustment, academic challenges, psychological health, physical health, victimization, and relationship issues, through a variety of interventions, referrals, and follow up services.
Email: DOS-deanofstudents@email.arizona.edu
Phone: 520-621-7057

Survivor Advocacy Program
https://survivoradvocacy.arizona.edu/
The Survivor Advocacy Program provides confidential support and advocacy services to student survivors of sexual and gender-based violence. The Program can also advise students about relevant non-UA resources available within the local community for support.
Email: survivoradvocacy@email.arizona.edu
Phone: 520-621-5767

Campus Pantry

Any student who has difficulty affording groceries or accessing sufficient food to eat every day, or who lacks a safe and stable place to live and believes this may affect their performance in the course, is urged to contact the Dean of Students for support. In addition, the University of Arizona Campus Pantry is open for students to receive supplemental groceries at no cost. Please see their website at: *campuspantry.arizona.edu* for open times.

Furthermore, please notify me if you are comfortable in doing so. This will enable me to provide any resources that I may possess.

Safety on Campus and in the Classroom

Familiarize yourself with the UA Critical Incident Response Team plans.

The Department of Computer Science Evacuation Plan for Gould-Simpson is here:

Also watch the video available at
https://ua-saem-aiss.narrasys.com/#/story/university-of-arizona-cert/active-shooter

Confidentiality of Student Records

http://www.registrar.arizona.edu/personal-information/family-educational-rights-and-privacy-act-1974-ferpa?topic=ferpa

Subject to Change Statement

Information contained in the course syllabus, other than the grade and absence policy, may be subject to change with advance notice, as deemed appropriate by the instructor.