Quick Facts

Essentials

Course Code: CS 131  
Course Title: Programming Languages  
Website: [http://www.cs.hmc.edu/cs131/](http://www.cs.hmc.edu/cs131/)  
Help Email: cs131help@cs.hmc.edu  
Professor: Melissa O’Neill oneill@cs.hmc.edu, Olin B157, x79661  
Prerequisites: CS 81, CS 60, CS 70  
Credit Hours: 3  
Class Times: Mon/Wed 11:00–12:15  
Class Times: Mon/Wed 2:45–4:00  
Lab Times: Tue, 6:30–7:30  

Overview

The official course description says:

*A thorough examination of issues and features in language design and implementation including language-provided data structuring and data-typing, modularity, scoping, inheritance, and concurrency. Compilation and run-time issues. Introduction to formal semantics.*

but what does that really mean?

This course provides you with a systematic framework for thinking about programming languages from a new perspective. CS 131 is organized around ideas that recur in many different languages (with less emphasis on particular choices of syntax). Understanding these ideas will help you better understand the languages you know, and help you to quickly master new languages—even languages that haven’t been invented yet. The concepts you learn will also help you to devise new languages of your own (which happens more often than you may think). We will study both the formal topics that arise when specifying the behavior of programming languages, and the practical issues that arise in implementing them.

By the end of the course, I am hoping that you will be thinking about programming languages more analytically. For example, you should be able to:

1. Decompose languages into atomic features, and hence
   - Learn new languages more quickly
   - Compare/critique programming language designs (beyond just syntax)
2. Choose the programming language (or programming idioms) best suited for a task
3. Find information in language specifications, when necessary
4. Use standard PL vocabulary when discussing languages
5. Understand basic implementation techniques and tradeoffs
6. Treat programs as data that can be analyzed or mechanically produced
7. See connections between theory, language design, and language implementation

A good background in different programming languages has given me the ability to think outside the box of one language style or another. Without this background, I would not be as valuable to my decidedly practically-minded company—even though we do not generally use theoretically interesting languages.

John Prevost, Software Engineer, ArsDigita Corp.

The assignments in the class give you practical experience in implementing key programming-language concepts, as well as doing significant work in Haskell, a language that has some valuable differences from the languages you may be used to. We will also use other languages to illustrate specific language features and concepts.

Electronic Access

Much of the communication for this course will be handled electronically through the class website and mailing list(s). You are responsible for being familiar with announcements posted to the class mailing list(s) and with the contents of the class website.

WEBSITE & Wiki

The class website is available at

http://www.cs.hmc.edu/cs131/

This website is also reachable from the CS department’s home page (via the Course Schedule link). As well as providing useful general information (such as how to find me when you have questions), homework assignments will be posted on the class website.

Wiki

The CS 131 website is also a “Wiki”. The pages are editable, allowing you to post material about CS 131. If you would like to create a discussion area for any topic in CS 131, you can do so (e.g., every assignment has a Hints and Tips page). The most important area of the Wiki is the ClassNotes area, where we post the content from our lecture slides and you add all the material that was not on those slides (including oral material and material written on the board). Included on the Wiki is a ClassNotesRoster for listing the a “designated note taker” for each class. These note takers have primary responsibility for the notes for a given
class, but you should check their notes regularly and make sure that they are complete and free of errors.

You should register on the CS departmental Wiki site as soon as you can, and add your name to the lecture-notes roster as soon as possible (see the [WikiHowTo](#) page for details on registering and tips on using the wiki well).

**Mailing Lists & Sending Email**

Most class-related questions should be sent to the help alias, cs131help@cs.hmc.edu. This address is the one you should use for *any and all* course-related questions; you should *only* send email to Prof. O’Neill’s personal email address if you have an issue so personal that you would not want anyone other than her to see it.

When you do have questions, it is vital to provide adequate context. The [AskingForHelp](#) page has lots of useful tips on asking for help effectively.

The class mailing list is cs131@cs.hmc.edu. If you were registered in the course on the first day of classes, you should already be on the list. In theory, these lists are maintained automatically, but if you are not getting mail, please investigate the problem (e.g., contact CIS) and keep us in the loop (e.g., via cs131help@cs.hmc.edu) as you work to resolve the issue.

**Computing Resources**

You undertake your homework assignments using the computer resources provided by the CS department, or you may use your own machine, if you can set it up appropriately. All the necessary software for CS 131 is set up on the Macintosh computers in our labs and on knuth, which is a fast Linux-based CPU server.

To use knuth or the lab Macs, you need a CS account. If you do not already have an account (and you haven’t heard from us that we’re getting you one), you should fill out an account-request form (available from Tim Buchheim, our CS system administrator in Beckman B159). If you have an account that is no longer active, see the system administrator or a member of the CS student staff to reactivate your account.

**Using Your Own Computer**

Instead of using the lab machines directly, you may instead use your own computer. The easiest way to use your own machine is to use it to log into knuth (via ssh) and then work there, but it is often possible for you to use your own machine directly. Helpful advice and tips on both of these options can be found on the [UsingYourOwnMachine](#) wiki page.

Because students individual machines can vary greatly in configuration, the course staff cannot guarantee to be able to resolve all possible issues that may arise when students use their own machines. Sometimes we can help, but sometimes you may be better off asking your peers in the class for help. Feel free to make the world better by adding useful tips to the wiki (e.g., the [UsingYourOwnMachine](#) page).
All homework assignments will be tested on knuth and are expected to follow standard Unix conventions (thus, converting files from any Windows-specific format is your responsibility). If you use your own machine, it is your responsibility to check that your code also works on knuth.

**Coursework & Grading**

Your final grade in CS 131 will be calculated by combining your grades each assessed course component in the proportions shown in Table 1.²

<table>
<thead>
<tr>
<th>Component</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>Assignments &amp; Labs</td>
<td>50%</td>
</tr>
<tr>
<td>Midterm</td>
<td>15%</td>
</tr>
<tr>
<td>Final</td>
<td>20%</td>
</tr>
<tr>
<td>Readings &amp; Quizzes</td>
<td>5%</td>
</tr>
<tr>
<td>Class Participation</td>
<td>5%</td>
</tr>
<tr>
<td>Wiki Participation</td>
<td>5%</td>
</tr>
</tbody>
</table>

Table 1: Course Components

Sometimes it can be necessary to make small adjustments to these proportions during the semester. If any adjustments are necessary, the changes will be announced in class and posted on the course website.

**Assignments**

There will be approximately ten homework assignments during the term. You will have one week to complete some of them, but others may require two weeks, due to their difficulty or other factors such as school breaks. Assignments are posted on the class website. Assignments will typically be due at 11:59 p.m. on Mondays.

**Group Programming in Labs**

Lab assignments should be undertaken in groups of two. You can work with the same person throughout the semester, or alternate as you see fit. It may be a wise move to decide who you’ll partner with before the lab begins to avoid wasting valuable lab time waiting for another unpaired person to show up. Come to lab on time.

**Readings & Quizzes**

Overlapping with the weekly assignments will be a number of smaller tasks that culminate in an on-line quiz, typically due “before class” on Monday. The quiz itself is intended to be quick and straightforward to complete, but often the quiz serves as a checkpoint on some other task (e.g., a reading), and completing the reading may not be quick.

Thus we recommend that you look at the quiz well in advance of its deadline so that you can plan your time accordingly.

²These weights apply in the normal case, but different rules may apply in extreme situations. You cannot, for example, pass the class, if you score 0% on the final, and 100% everywhere else.
EXAMS

There are two exams (a midterm and a final). These exams are three-hour open-book take-home exams. In general, exams test do not test recall of facts, but your ability to apply what you know.

Your Responsibilities

You have a number of responsibilities in this class. It is important that you understand them so that you can avoid costly mistakes such as loss of credit on an assignment or accidental violation of the honor code.

READING & SENDING E-MAIL

It is your responsibility to have working email infrastructure, both for sending and receiving email. E-mail messages may be sent to any of your valid addresses, including

- Your official campus email address (often via the mailing list cs-131-l@hmc.edu);
- Your CS email account; and,
- Any account that you have used to contact the course staff (e.g., your gmail account).

If you do not want to check all your email sources every day, be sure to set up forwarding from one account to the other. (See [http://www.cs.hmc.edu/wiki/QREF/Mail](http://www.cs.hmc.edu/wiki/QREF/Mail)). CS accounts are usually created so that they forward email to your main Mudd account, so for most students no action is required.

Sending email with “forged” from addresses (e.g., telling Yahoo pretend that your email came from your HMC account) causes problems; your mail may be rejected by our mail servers (because spammers do this!). Instead, if you’re sending email from your Yahoo account, let it come from that account. Just realize that we may not necessarily know who mrbuzzkill97@yahoo.com actually is, so be sure to include your name.

PAIR PROGRAMMING

CS 131 recommends that students pair-program on homework assignments. You may decide on a per-homework basis with whom you wish to work. Although pair-programming is preferred, it is not required.

If/when you choose to pair program, you must use proper pair-programming methodology; namely, you must spend the bulk of your time working together as a team, with one person at the keyboard and the other at their side watching and making suggestions. All work should be a joint effort. You will be violating the honor code if you divide the work such that you work separately with one person doing one half of the work and the other person doing the other half.
**Collaboration Outside of Pair Pairgramming**

We would love for you to discuss the lecture and reading topics with any or all of your classmates; your discussions can range anywhere from informal chats in the hallway to formal study groups organized on the course wiki.

You can even discuss high-level features of assignments and the ideas involved, including general approaches to the problems, bugs in the specification, how long you’ve spent working on a problem, and so forth. You may also help each other with basic issues related to completing the assignments—how to use Unix, Haskell syntax, and the like.

Two good rules of thumb for appropriate collaboration on homework assignments are the “in your head rule” and “understanding, not rote learning” rules. When students help each other, they should leave with understanding in their heads, not physical or electronic artifacts. Thus you are not allowed to meet as a group and leave with notes on paper, nor can you help someone fix a bug and then leave without first reverting the bug to its unfixed state. Similarly, it is not okay to leave with an answer you don’t understand (in the worst case, where such a situation seems unavoidable because you have found out the answer but still don’t understand it and have to leave, it is *vital* for you to say so, since not doing so is dishonest). Also, if you are helping someone, remember that giving away answers can be detrimental to someone’s understanding—often it is better to ask leading questions instead. Working together as peers and coming to an understanding together is, of course, even better.

**Reminder: The Honor Code**

All students—even those from other colleges—are expected to understand and comply with Harvey Mudd College’s Honor Code.² If you haven’t already done so, you must read, sign, and abide by the computer-science department’s interpretation of the Honor Code to participate in this course.³ Specifically:

- You must not exchange literal copies of material, whether that material consists of code, program output, or English-language text (e.g., documentation). You also may not copy material from published or online sources, with or without cosmetic changes (such as altering variable names), without explicit permission. If you do have permission to use externally written material, you must attribute it properly and clearly indicate which material is yours and which material is not yours.

- You should not do anything that a reasonable student peer would describe as “subverting the clear intent of the assignment”, unless you have asked for and received permission to do so. Finding open-sourced code that you can use to solve an assigned problem, for example, would typically be subverting the intent of the assignment because your shortcut means that you do not learn what the assignment aims to teach.


³See [https://tinyurl.com/hmc-cs-honorcode](https://tinyurl.com/hmc-cs-honorcode) for details.
• If you use any sources to assist you, you must document them. For example, all assignments have a hints-and-tips page on the wiki. If a “tip” from that page ends up incorporated into your graded work, you must credit the source. (A clarification about assignment requirements or a debugging tip, however, need not be credited, although it doesn’t hurt to do so if you wish.)

• If any assigned material is substantially similar to material you have done before please contact me. In particular, in an situation where I am expecting you to do new work, you may not reuse or refer back to substantially similar work you have previously done. It is just as bad to copy from your past self as from someone else.

• Academic integrity also involves being careful enough to avoid unintentionally breaking the rules. Thus, you must read instructions in assignments and exams carefully so that you are aware of any limitations they place on you, such as time restrictions or restrictions on information sources you may consult. Similarly, if you see something that plausibly seems like it ought to be off-limits to you, such as a Subversion directory belonging to another student or files from a previous semester, you should immediately contact us to let us know that something doesn’t seem right, rather than looking further at something that perhaps you should not see at all.

• If you aren’t sure whether something you’ve done or plan to do is allowed, you should explicitly document what you did and—if at all possible—consult with the course staff, ideally before you take the questionable action. Similarly, document any extensive or particularly important help you obtain, even if that help seems legitimate. If you’ve been helped so much that we can’t consider the work truly your own, you might not be able to get full credit for it but proper attribution will avoid an Honor Code violation.

• The honor code requires you to take action when you see other people doing something questionable (in this course, or any other). If you like, you can begin by talking to the person concerned to find out what is going on, and perhaps encourage them to talk to their course staff if what they are doing seems questionable. But you should probably also report the issue to the course instructor as well, just to be sure. Don’t just be a passive observer; taking action might not always be pleasant, but without vigilance from everyone, the honor code cannot live up to its ideals.

These principles apply to all methods and media of discussion or exchange (voice, writing, email, etc.).

While obviously it’s best to avoid honor code related issues, it’s worth remembering that everyone makes mistakes an error of judgement once in a while—where you show your true character is not just in how careful you are to avoid errors of judgement but also in how you handle them if they occur.
ATTENDANCE & PARTICIPATION

You are expected to attend every class. I will not be formally taking attendance, but many classes will have group exercises that will affect your final grade (both directly, because I grade for class participation, and indirectly, because questions on the exams are often similar to the group-exercise questions). If you wish to miss a class for any reason, you should ask beforehand about the make-up work you will need to do. If you are sick, you can send word to me through another student.

You are expected to participate actively in each class. The only way to receive a high grade for class participation is to be an active participant in the class (duh!).

DUE DATES & LATE POLICY

LATE WORK IS STRONGLY DISCOURAGED

The penalties for submitting late almost always outweigh the benefits. With proper planning, there is almost always a way to avoid suffering the penalties that arise from turning work in late. For example, if you know of an upcoming commitment that (such as an exam in another course, or a family event you must attend) that could affect your ability to get an assignment done by its deadline, you may ask for an assignment early. Similarly, if, after starting an assignment, you think it is impossible to accomplish the necessary work in the available time, consult me before the due date. It may be that there is some help I can offer you that would allow you to proceed more quickly, or it may be that everyone in the class is having the same problem, in which case we might adjust the assignment.

LATE PENALTY FORMULA

If, against my advice above, you do submit work late, your score will be scaled using the multiplier returned by the following Haskell function

\[
\text{latemult } \text{mins} = \max (0, (1.0 - (\text{root} * (1.0 - \text{root}) + \text{squared} * \text{root})))
\]

where

\[
\text{lateness} = \frac{\text{mins}}{12 \times 60}
\]

\[
\text{root} = \sqrt{\text{lateness}}
\]

\[
\text{squared} = \text{lateness}^2
\]

![Figure 1: The Late Penalty Function](image-url)
This function (which is shown graphically in Figure 1) is swift to penalize lateness—even one minute late results in a lateness multiplier of 0.964 (i.e., a 3.6% penalty). At nine minutes late, the multiplier is 0.9, a 10% penalty. But with time the rate slows down, the 0.8 point is reached at a little before the first hour, and the 0.6 multiplier is not reached until the work is about 6.5 hours late. At, and beyond, the 12-hour point, the multiplier is zero. Times are based on the time that the submission process finished, not when you began submitting. For assignments due at 11:59 PM, we count late minutes from midnight and round down to whole numbers of minutes. Thus, you have about two minutes of grace after the clock ticks over to reading 11:59 PM. (It is, of course, risky to cut things close, especially since the submit system takes time to perform its work.)

Remember, unless you’ve very much underestimated the assignment, it is almost never a good idea to submit late. The added points you gain from the extra work won’t compensate for the effect of the lateness multiplier.

If you wish to finish an assignment but have your predeadline work graded, do not submit your later work using the regular submit system. Instead, send mail to cs131help and ask for special instructions for submitting your finished work for grader comments.

Extemuating Circumstances

Extenuating circumstances (such as illness) are dealt with on a case-by-case basis. In general, you are only excused for situations you could not have foreseen, and only if you explain the situation at your soonest opportunity (either directly or via someone else, such as the Dean of Students), before the due date.

For situations that you can plan ahead for, you are expected to do so. If you have an upcoming personal commitment that will impact your ability to work at the usual time, tell all your instructors as soon as possible. If we know, we may be able to provide you with work early. Similarly, if the deadlines for two courses fall on the same day, tell both instructors immediately.

Illness

If you get sick during the term, notify me immediately, even if you think that being sick will not affect your ability to complete your assignments. You should also notify me any time that you’re sick enough to miss any classes (not just CS 131) or find that your performance is below par for any reason.

Getting Help

Recognizing when you should ask for help, and how to do so effectively, is an important skill that you should cultivate. If you’re stuck or you find yourself making random changes to your Haskell code (“Maybe now it will compile??”), recognize that you might benefit from getting some help from a grutor and/or myself.

When you need help with a course-related problem, come and see me or one of our graders. Talking in person is often the fastest way to resolve a problem, especially if it’s conceptual. If you speak to a grader and are not happy with (or convinced by) their answer, you should
see it as their failure rather than yours, and seek me out for a better answer.

When you have a simple question, a clarification, or a simple request, you may prefer to use email. You should send email to the cs131help@cs.hmc.edu mailing list, which is read by me and the graders, maximizing your chances of getting a quick answer to your question. The AskingForHelp wiki page has useful tips on how to use email most productively.

If you have sensitive issues—such as personal issues or Honor Code violations—that you need help with, contact me directly.

**Enjoy the Class**

Inevitably, a policies document like this one focuses on legalistic issues (whose rationale is making the class fair and manageable for everyone participating). But I want you to learn a lot and enjoy doing so. Some parts of the class may take you outside of your comfort zone as part of a process of learning, but nevertheless I want the class to be a welcoming and encouraging place for you. If some aspect of the class leaves you unhappy or not feeling valued, please talk with me immediately—I want to know, because I want every student to have a great experience taking this class.