

Syllabus

<i>Unique number</i>	52870	52875
<i>Meeting time</i>	MWF 1–2p	MWF 3–4p
<i>Classroom</i>	GEO 2.102	TAY 2.006
<i>Final-exam Date</i>	Sat 14 May 2–5p	Wed 11 May, 7–10p

	Instructor	Assistant	Assistant
<i>Name:</i>	Hamilton Richards	Feng Wang	Changhai Xu
<i>Mailbox</i>	Taylor 2.124	—	—
<i>Office hours</i>	see WWW site	see WWW site	see WWW site
<i>Office</i>	Taylor 5.138	see WWW site	see WWW site
<i>Phone</i>	471–9525	—	—
<i>e-mail</i>	ham@cs.utexas.edu	wangf@cs.utexas.edu	changhai@cs.utexas.edu

Content This course surveys significant concepts underlying modern programming languages, including syntax, parsing, scope, functions, relations, expressions, types, assignment, procedures, pointers, encapsulation, classes, exceptions, and inheritance, with some discussion of implementation issues. It covers examples of several prominent programming paradigms, including sequential, concurrent, object-oriented, and functional programming. The programming languages mainly used to illustrate these concepts and paradigms are C, Pascal, Modula-2, Ada, C++, Java, and Haskell.

Prerequisites To be eligible for CS 345, you must have completed CS 310(H), CS 336(H), and Math 408D with grades of C or better. Enrolling more than once in CS 345 requires written consent of a CS academic advisor.

Texts R. Sethi. *Programming Languages: Concepts and Constructs*, Second Edition. Addison-Wesley, 1996.

Antony J.T. Davie, *An Introduction to Functional Programming Systems Using Haskell*. Cambridge University Press, 1992. This book is out of print, but the parts of it we need are available from Co-op Custom Publishing as *Readings from "Intro to Functional Programming Systems Using Haskell"*.

P. Hudak, J. Peterson, J.H. Fasel. "A Gentle Introduction to Haskell". Available on the Web at <http://haskell.org/tutorial/index.html>. Assumes familiarity with some lazy functional language, and concentrates mainly on features that distinguish Haskell from other such languages.

Optional Texts The following texts may be useful in case you wish to pursue some topics in greater depth than we'll be able to cover them in this course. A text's appearance in this list is *not* meant to suggest that you're expected to purchase it, and no reading will be assigned in any of these books.

Richard Bird, *Introduction to Functional Programming using Haskell*, second edition. Prentice Hall International (UK), 1998. An update of the classic functional-programming text by Bird and Wadler.

Norman H. Cohen, *Ada as a Second Language*, second edition. McGraw-Hill, 1996.

Kenneth C. Loudon, *Programming Languages: Principles and Practice*, second edition. Brooks/Cole—Thomson Learning, 2002.

Bjarne Stroustrup, *The C++ Programming Language*, second edition. Addison-Wesley, 1992.

David Eck, "Introduction to Programming Using Java", second edition, <http://math.hws.edu/eck/cs124/notes/>.

Niklaus Wirth, *Programming in Modula-2*, fourth edition. Springer Verlag, 1988.

Computer Accounts If you are not eligible for your own permanent Computer Sciences Department account, you can get a temporary class account on the Computer Sciences Department unix network. To sign up for a class account, take your fee bill to the CS Undergraduate Office (Taylor 2.126).

Handouts Copies of lecture notes, homework and test solutions, and occasional other material are provided for you to pick up at the back of the lecture room. Leftover copies are placed in the pockets labeled "CS345" on the wall outside my office (Taylor 5.138).

Attendance Attendance is recorded by passing around a sign-up sheet. Attendance doesn't count explicitly, but regular attendance at all class meetings is expected (of course we understand that occasional absences are unavoidable). Absences are reported to the registrar, and become part of your permanent UT record.

Schedule The topics we'll cover, and the approximate lectures devoted to each:

1. **Introduction:** overview of course, administrative matters. Complexity, modularity, and abstraction. Programming paradigms.
- 2–4. **Syntactic structure.** Expression notations. Abstract syntax, tree notation. Context-free grammars, BNF syntax notation. Parsing. Ambiguity; precedence, associativity. Attribute grammars. Other syntax notations.
5. **Expression evaluation:** rewrite rules and substitution.
- 6–16. **Functional programming.** Introduction to a language of pure expressions: Haskell, using Hugs98.
Names, let-expressions, local definitions. Lambda expressions, curried functions, named function definitions. Innermost, outermost, and lazy evaluation; strictness. Selective evaluation: if and case expressions. Recursive definitions. Guards, patterns. List types. Strings, arithmetic sequences. List comprehensions, memoization. Prelude list-processing functions. Recursive list construction. Patterns using [] and (:). Higher-order functions: map, filter, foldr, foldl. Infinite lists: primes, square root. Input and output: monadic I/O.
Parsing using a monadic parsing-combinator library.
- 17–20. **Imperative programming:** Assignments, state, and control flow.
Guarded commands and weakest preconditions. Formal reasoning: correctness proofs in guarded-command notation.
Control flow in Pascal, Modula-2, and C: Selection and repetition; syntax issues. Irregular termination of loops.
- 21–27. **Data types.**
Type expressions, polymorphic types.
Type checking: static vs. dynamic; type checking for conventional languages, type inference for Haskell.
Type conversion and coercion.
Elementary types and their operators. C's side-effect operators.
Product types: records, structs, and algebraic types. Recursive product types. Sets.
Sequence types: arrays and lists. Array parameters in Modula-2.
Pointers; pointer arithmetic and arrays.
Overloading operators and function names in C++ and in Haskell; type classes, class derivation.
- 28–30. **Procedures.**
Scope rules: lexical and dynamic scoping.
Parameter-binding methods: by value, by reference, by value-result, and by name.
Macro expansion. Polymorphism: overloading vs. function templates.
Implementation: activation records. Tail recursion and iteration. Procedures as arguments. Local-variable layout.
- 31–33. **Concurrent programming**
Implicit synchronization: shell programs, function composition.
Explicit synchronization in Ada: the *rendezvous*. Examples: producer-consumer problem, print server, prime-number generator.
- 34–36. **Encapsulation** and information hiding.
Abstract data types: Classes (C++ and Java); shallow and deep copying, initialization, assignment, memory management. Haskell modules.
Polymorphic classes: C++ class templates.
- 37–39. Class derivation and member inheritance in C++ and Java. Virtual functions; subtypes and private base classes.
Prime-number generator. Private derivation.
- 40–42. Error handling using exceptions in C++, Java, and Haskell.
43. Wrap-up.

Homework A set of exercises is assigned each **Monday**, and is generally due on **Tuesday** of the following week. You hand in your paper by dropping it in the Taylor Hall homework box (between 2.132 and 2.136) by **4pm** (sorry—we're not set up to accept homework papers by e-mail). Suggested solutions for each set are handed out at the lecture following the day it is due, and leftover copies are available in a wall pocket outside my office (Taylor 5.138).

Except by prior arrangement with me, homework is not accepted after the deadline. System problems, printer failures, traffic delays, and the like are routine occurrences, and are not considered cause for extending homework deadlines. To avoid problems, get started on homework assignments early, and allow a margin for contingencies.

It's OK for you to work on the homework exercises with other students, and for groups of up to four students to hand in a single paper. It's tempting to split up an assignment among group members, but each group should work *together* on all exercises, to ensure that all members learn what the exercises have to offer.

Graded homework papers are returned in class. You should bring any questions about the grading to the grader's attention within *one week* after the set has been handed back. If you are unable to resolve a disagreement with the grader, then you are welcome to bring the problem to me.

Although homework assignments' point totals may vary, all count equally towards the course grade. The average of your homework exercises' grades—with the lowest grade dropped—counts as 10% of your course grade.

Office hours My scheduled office hours (see the class web site) are reserved for consultations with students on a "drop-in" basis. If these hours are inconvenient for you, please make an appointment with me for some other time in person after class, or by telephone or electronic mail. You may also simply take a chance on finding me in my office and available, but not—please!—during the hour immediately before a class (check my weekly schedule, which is posted outside my door and on the class web site).

Web pages The CS 345 home page is <http://www.cs.utexas.edu/users/ham/UTCS/CS345/>. Please make sure you're aware of the information presented in these pages. Class-related announcements are posted at <http://www.cs.utexas.edu/users/ham/UTCS/CS345/announce.html>.

Newsgroup The course has a usenet newsgroup, `utexas.class.cs345-richards`, which is intended for questions and discussions of general interest. *If you have a question for the instructor or one of the TAs, don't post it on the newsgroup only—you'll get much quicker results using e-mail.*

Electronic mail It is now UT policy that official communications can be sent to students by e-mail <<http://www.utexas.edu/its/policies/emailnotify.html>>, and the University assumes that you receive e-mail communications and read them promptly. It is your responsibility to keep the University informed of changes in your official e-mail address. E-mail returned to the University with "User Unknown" or "Mailbox Full" is not an acceptable excuse for missed communication.

Because considerations of privacy have led the university to advise instructors not to post grades, I will use e-mail to report your grades to you after each homework and each exam, and at the end of the semester. These messages will be sent to the address the university has for you; if you would prefer some other address, please send me an e-mail from the original address or hand me a note in class. If you do not have an e-mail account, you can get one from the CS Department (take your fee bill to Taylor 2.126) or from the university (for details, see <http://www.utexas.edu/computer/email.html>).

Examinations This course has three 1.5-hour midterm tests, of which only your best two count. The 3-hour final examination covers the entire course, with some extra emphasis on material not covered on any midterm test.

Exam	weight	date, time, classroom	lectures covered	homework covered
test 1	25%	Tue 22 Feb, 7:30–9:00pm, UTC 2.102A	1–10	1–3
test 2	25%	Tue 29 Mar, 7:30–9:00pm, UTC 2.102A	11–22	4–6
test 3	25%	Tue 26 Apr, 7:30–9:00pm, WEL 1.316	23–34	7–9
final exam	40%	for dates, see page 1; classrooms TBA	1–43	1–10

Note: The dates for the mid-term tests are *subject to change* in case of serious conflicts with other classes. The actual dates will be announced in class.

Books and notes are excluded from tests and examinations, except that you may bring *one* sheet of notes (i.e., one sheet of 8.5"×11" paper, written on one or both sides) to each of the tests, and *two* such sheets to the final.

Requests for changes in test grades must be submitted *in writing* within *one week* after the test papers are handed back. Delay in picking up a graded test paper does not extend this deadline.

Students with disabilities Please notify me of any modification/adaptation you may require in order to accommodate a disability-related need. You will be requested to provide documentation to the Dean of Students' Office, in order that the most appropriate accommodations can be determined. Specialized services are available on campus through Services for Students with Disabilities.

Illnesses and Absences The provisions for dropping your lowest homework and test grades are intended to take care of routine cases of personal business (including employment interviews) or illness. If university business or an extraordinary illness prevents you from handing in an assignment or taking a test, please contact me as soon as possible.

Absences for the observance of religious holy days will be excused, provided you hand me a written notice for me to sign, at least two weeks in advance (for holy days in the first two weeks of the semester, notify me on the first class day). For further information, see *General Information 2004-2005*, Chapter 4 (available on the web at <http://www.utexas.edu/student/registrar/catalogs/gi04-05/ch4/ch4g.html#attend>.)

Important dates

Friday **21 January** (4th class day)

Last day to add or drop a class using TEX or ROSE.

Wednesday **2 February** (12th class day)

Last day to add a class except for unusual circumstances.

Last day to drop a class for possible refund.

Last day a course can be dropped without counting as an enrollment under the CSDepartment's limited-repetition rule.

Monday **14 February**

Last day to drop a class with an automatic Q.

Monday **28 March**

Last day to drop a class (with a possible F) or withdraw from the university, except for urgent and substantiated nonacademic reasons.

Last day to change registration in a class to or from Pass/Fail.

Last day to apply for an undergraduate degree.

Finally, welcome to the course. I hope you'll find it worthwhile, and that you'll enjoy taking it as much as I enjoy teaching it.

— Hamilton Richards
Instructor, CS345