

<http://www.cs.umd.edu/class/fall2005/cmsc330>

1 Prerequisites and description

Prerequisites: C or better in CMSC 212 (or CMSC 214), and in CMSC 250

Credits: 3 credits

A study of programming languages, including their syntax, semantics, and implementation. Several different models of languages are discussed, including procedural (e.g., C, Pascal), functional (e.g., ML, LISP, Scheme), rule-based (e.g., Prolog), and object-oriented (e.g., Java, C++, Smalltalk). Language features such as formal syntax, scoping and binding of variables, higher-order programming, typing and type polymorphism, and object inheritance are explored.

2 Instructors

Sections 0101 and 0102	Sections 0201 and 0202
Jeff Foster 4129 A. V. Williams, 301-405-2751 jfoster at cs dot umd dot edu (*) Tu 10:30-12:00, W 12:00-1:30 Or by appointment	Larry Herman 1111 A. V. Williams, 301-405-2762 larry at glue dot umd dot edu (*) M 10:00-10:50, W 4:00-5:00, F 4:00-5:00 Or by appointment

(*) Please refrain from using email for matters better discussed in person. In particular, we will not provide help with programming projects, homework, or lecture material over individual email, since doing so often leads to incomplete or inadequate information. Instead, please discuss such matters in person, either during office hours or before or after class, and post questions of a general nature to the class newsgroup.

3 Teaching assistants

Sections	TA	Email
0101, 0102	Nik Swamy	TBA
0201, 0202	Asad Sayeed	asayeed at glue dot umd dot edu
—	Denis Filimonov	den at glue dot umd dot edu
—	George Caragea	gcaragea at glue dot umd dot edu

TA office hours will be posted on the course web page a few days after classes begin.

While the TAs will provide assistance with assignments during office hours, you are responsible for developing and debugging your own programs. Do not rely on the instructional staff to make your project work. Lower-level CMSC courses provide extensive debugging and development help in office hours, but upper-level CMSC courses expect students to complete projects with minimal extra help. Therefore in CMSC 330, we will provide less debugging help than some students may be used to. If you come in with a question, expect to be pointed in the right direction, but it will be up to you to finish solving the problem on your own.

4 Textbooks, newsgroup, and computing resources

There is no required or recommended text for this course. The class web page contains links to on-line reference information for the new languages in the course, and we will also provide suggestions for other sources of information throughout the semester. Slides used during the course will be posted on-line after lecture.

Important announcements will be made in class and on the class web page, so please make it a habit to check the web page daily. The class newsgroup, `csd.cmsc330`, can be used to ask general questions of interest to the class as a whole, such as administrative issues and project clarification questions. Please do not post any information to the newsgroup that would violate the university academic integrity policy.

Programming projects will be developed on the new OIT Grace UNIX Cluster, `grace.umd.edu`. You will use your own Glue account to access the cluster and do coursework, so if you do not have a Glue account, request one immediately online at <http://www.oit.umd.edu/new>.

If you have access to another system you are welcome to do your development there instead, but all project submissions **must** work correctly on the Grace cluster, and your projects will be graded solely based on their results on the cluster. Because language and library versions may vary with the installation, in unfortunate circumstances a program might work perfectly on your system but not work at all on the cluster. Thus we strongly recommend that if you develop any project on another system, you should complete it **several days early** to have time to address any compatibility problems.

5 Exam and final dates

The class includes two midterms and a final exam, given at the following dates and times:

Exam #1: (Tentative) Monday, October 3, during discussion time
Exam #2: (Tentative) Monday, November 14, during discussion time
Final exam: Thursday, December 15, 4:00–6:00 p.m.

The exact dates for the midterm will be confirmed later, and may vary depending on lecture progress and other factors (we will let you know well in advance). Midterms will be held during the discussion time, but probably in a different location, to be announced. The final exam date is **fixed** by the University. The final exam will be rescheduled **only** for students having another final at **exactly** the same time, or for students with three or more final exams scheduled on the same day. (The only students whose finals are at the same time as this course's final are those also taking BIOM 301 or BMGT 350.) If either of these situations applies to you, you must inform the instructor **during the schedule adjustment period** for any allowances to be made. Please also let us know immediately if you have a conflict with a midterm date.

6 Attendance, homework, and general grading policies

You are responsible for all material discussed in lecture and discussion section (even if you were not in class) and posted on the class web page, including announcements, deadlines, policies, etc. During the semester we may provide ungraded practice homework exercises and solution. While we will not collect these exercises, completing them is essential preparation for exams. You may work together on these ungraded homeworks, and you may of course come to office hours for additional help.

Your final course grade will be determined according to the following percentages:

Intro project	0.5%	
6 Programming projects	39.5%	(equally weighted; 6.58% each)
Quizzes in discussion section	5%	(equally weighted)
2 Midterms	30%	(equally weighted; 15% each)
Final	25%	

We expect to assign six programming projects, which will be equally weighted out of 39.5% (the last 0.5% is for the introductory project). There is a chance we will assign more or fewer projects, in which case they will be given equal weight out of 39.5%. Quizzes will be given in discussion section and will cover discussion and lecture material. We will announce quizzes in an earlier class.

Any request for reconsideration of the grading on any coursework **must** be submitted within **one week** of when it is returned. Exam regrading requests must be made in writing. Information about resolving project grading questions will be provided when the first project is returned. Any coursework submitted for reconsideration may be regraded in its entirety, which could result in a lower score if warranted.

Final course grades will be curved as necessary, based on each student's total numeric score for all coursework at the end of the semester.

Note: This is a programming course, and completing the projects is an essential part of the course. Therefore, **no student will pass the course (with a grade of C– or higher) if at the end of the semester they have a zero grade for any project**, regardless of their performance or scores on the other coursework. Complete project grading policies are below.

7 Project submission and grading policies

7.1 Project submission method and deadlines

Projects must be submitted electronically, following the directions provided with the first project assignment. Projects **may not** be submitted by any other means (such as by email). It is **your responsibility** to test your program and **verify that it works properly** before submitting.

All projects are due at 10:00 p.m. on the day indicated on the project assignment, according to the UNIX time of day on the submission server. Each project is graded out of a maximum of 100 points, and for every day (24-hour period) that a project is late, 10 points will be deducted. Submission deadlines are **firm** and no exceptions can be made. Note there is **no grace period** for project submissions—deadlines will be enforced at exactly 10 p.m. the day a project is due, and every 24 hours later.

Project extensions will not be given to individual students as a result of system problems, network problems, power outages, etc., so **do not wait** to submit a project until the night it is due. You may submit multiple times up to the deadline, and only your last on-time submission is graded. Hence it is strongly suggested you finish and submit your program **at least** one day early, to allow time to reread the project assignment and all relevant articles in the class newsgroup, to insure you have not missed anything which could cause you to lose credit on the project.

7.2 Project grading policies

Projects will be graded by running them against automated test cases that check your program against the project specification. Note that unlike lower-level programming classes, we will **not** provide you with extensive automatic testing before projects are due. Instead, you will be responsible for developing your own techniques for testing your projects. To reiterate: **your projects will be graded based on test cases not provided in advance**. Because the results of these test cases are compared automatically, **you must follow the project specification exactly**. Also, while projects will generally not be graded on style or documentation, we reserve the right to manually grade program source code for some projects.

All projects will be graded on the OIT Grace UNIX Cluster. Having a working version on another system at any other time (or even another working version in your class account) will not be considered. No consideration in grading will be made for errors made in transferring files, or submitting the wrong version of your project.

Based on its results against the test cases we use during grading, your projects will be graded as follows:

- A project that was not submitted will receive a score of zero (but see below).
- Each project assignment will specify certain **minimum requirements** to be given as part of the project assignment. Projects which do not satisfy the minimum requirements will receive a score of zero. Note that in some cases we may supply you with limited automated test cases ahead of time, but these tests may cover only a subset of the minimum requirements.
- Projects that satisfy the minimum requirements will be graded out of 100 points as follows:
 - The project's score will be computed by summing up the point values of each automated test case that it satisfies. There will be no partial credit for these test cases.
 - Then ten points will be deducted for each day late the project was submitted, up to a maximum 90-point deduction.
 - No project that satisfies the minimum requirements will receive a score less than 10, even if the late penalty and deductions for incorrect test cases would be greater than 90 points.
- If you submit more than once for a project (on-time, one day late, two days late, etc.), then all will be graded, and you will receive the highest score as the project grade.

As mentioned above, a student will not be permitted to pass the course (with a grade of C- or better) with a zero score for any project. **However**, project scores of zero may be **removed** and replaced with a higher score by correcting and resubmitting the project to satisfy the minimum requirements. Thus to be able to pass the course, you must submit for every project a version that satisfies the minimum requirements. The sooner the corrected project is resubmitted, the more partial credit it can receive. Following the rules above, a project submitted after eight days that satisfies the minimum requirements will receive a score of 10 points. **All projects must be submitted by the last day of classes.**

Finally, any “hardcoding” in a project assignment will result in a score of zero for that project (which like any zero project score can be replaced by correcting and resubmitting the project, without hardcoding). Hardcoding refers to attempting to make a program appear as if it works correctly, when in fact it does not. Examples would be a program which prints the desired output instead of computing it, or a program which works only because it takes advantage of properties which test cases which are part of the minimum requirements happen to have, etc. These are only a few examples; if you have any question about whether a particular situation would constitute hardcoding be sure to ask ahead of time.

8 Excused absences and accommodations

8.1 Excused absences

Besides the policies in this syllabus, the University’s policies apply during the semester. Various policies that may be relevant appear in the Undergraduate Catalog at <http://www.umd.edu/catalog>.

If you experience difficulty during the semester keeping up with the academic demands of your courses, you may consider contacting the Learning Assistance Service in 2201 Shoemaker Building at (301) 314-7693. Their educational counselors can help with time management issues, reading, note-taking, and exam preparation skills.

Missing a quiz or exam for reasons such as illness, religious observance, participation in required university activities, or family or personal emergency (such as a serious automobile accident or close relative’s funeral) will be an excused absence. However, excused absences must be requested in writing and must include documentation that the absence qualifies as excused.

For example, for medical absences, you must furnish documentation from the health care professional who treated you. The documentation should include the name and phone number of the health care professional and should explicitly state the dates and times that you were **incapacitated** and therefore unable to attend. The dates and times must **include** the date of the missed exam or quiz; an illness preceding an exam or quiz does not constitute an excused absence. Also, simply being seen by a health care professional **does not constitute an excused absence**. Furthermore, self-documentation of illness is not sufficient support to excuse an absence. Excused absences will not be given unless documentation as described is provided. **If you become ill, keep in mind that the University Health Center will not provide medical documentation.**

It is the University’s policy to provide accommodations for students with religious observances conflicting with exams, but it is the **your responsibility** to inform the instructor **in advance** of intended religious observances. Written notice must be provided **immediately** upon an exam date being announced or confirmed in order for an absence to be excused. If you have a conflict with one of the planned exams, you **must** inform us prior to the end of the schedule adjustment period. Excused absences for quizzes will also be considered in the case of religious obligation; contact the instructor as soon as possible after the quiz date.

There will be no makeups for missed quizzes; with an excused absence that quiz will be discarded when computing your final score. For missed exams due to excused absences, the instructor will arrange a makeup exam. However, unless **immediate** notice is given as early as possible of the reason for any missed coursework, an excused absence may not be granted. If you might miss an exam for any other reason other than those above, you must contact the instructor **in advance** to discuss the circumstances. We are not under obligation to offer a substitute assignment or to provide a makeup exam unless the failure to perform was due to an excused absence.

The policies for excused absences do not apply to project assignments. Projects will be assigned with sufficient time to be completed by students who have a reasonable understanding of the necessary material and begin promptly. In cases of **extremely serious** documented illness of **lengthy duration** or other protracted, severe emergency situations, the instructor may consider extensions on project assignments, depending upon the specific circumstances.

8.2 Students with disabilities

Students with disabilities who have been certified by Disability Support Services as needing any type of special accommodations should see the instructor as soon as possible, during the schedule adjustment period.

All arrangements for exam accommodations as a result of disability **must** be made and arranged with the instructor **at least** three business days prior to the exam date, or accommodations **will not** be made.

9 Academic integrity statement

The Campus Senate has adopted a policy asking students to include the following statement on each examination or assignment in every course: “I pledge on my honor that I have not given or received any unauthorized assistance on this examination (or assignment).” Consequently, you will be requested to include this pledge on each exam and project.

Please carefully read the Office of Information Technology’s policy regarding acceptable use of computer accounts provided for instructional use at www.inform.umd.edu/CompRes/NEThics/aug.

Programming projects are to be written INDIVIDUALLY, therefore cooperation or use of unauthorized materials on projects is a violation of the University’s Code of Academic Integrity. **Any evidence** of this, or of unacceptable use of computer accounts, use of unauthorized materials or cooperation on exams or quizzes, or other possible violations of the Honor Code, **will be submitted** to the Student Honor Council, which could result in an XF for the course, suspension, or expulsion.

- For learning the course concepts (including the programming languages), students are welcome to study together or to receive help from anyone else. You may discuss with others the project requirements, the features of the programming languages used, what was discussed in class and in the class newsgroup, and general syntax errors. Examples of questions that would be allowed are “Does a Java class definition end in a semicolon?” or “What does a ‘class not found’ error indicate?”, because they convey no information about the contents of a project.
- When it comes to actually writing a project assignment, other than help from the instructional staff a project must solely and entirely be your own work. Working with another student or individual, or using anyone else’s work IN ANY WAY except as noted in this paragraph, is a violation of the code of academic integrity and WILL BE REPORTED to the Honor Council. You may not discuss design of any part of a project with **anyone** except the instructor or teaching assistants. Examples of questions you may **not** ask others might be “How did you implement this part of the project?” or “Please look at my code and help me find my stupid syntax error!”. You may not use any disallowed source of information in creating either their project design or code. When writing projects you are free to use ideas or **short fragments** of code from **published** textbooks or **publicly available** information, but the specific source must be cited in a comment in the relevant section of the program.

VIOLATIONS OF THE CODE OF ACADEMIC INTEGRITY MAY INCLUDE, BUT ARE NOT LIMITED TO:

1. Failing to do all or any of the work on a project by yourself, other than assistance from the instructional staff.
2. Using any ideas or any part of another person’s project, or copying any other individual’s work in any way.
3. Giving any parts or ideas from your project, including test data, to another student.
4. Allowing any other students access to your program on any computer system.
5. Transferring any part of a project to or from another student or individual by any means, electronic or otherwise.

If you have any question about a particular situation or source then consult with the instructors in advance. Should you have difficulty with a programming assignment you should **see the teaching assistants in office hours**, NOT solicit help from anyone else in violation of these rules.

IT IS THE RESPONSIBILITY, UNDER THE HONOR POLICY, OF ANYONE WHO SUSPECTS AN INCIDENT OF ACADEMIC DISHONESTY HAS OCCURRED TO REPORT IT TO THEIR INSTRUCTOR, OR DIRECTLY TO THE HONOR COUNCIL.

Every semester the department has discovered a number of students attempting to cheat on project assignments, in violation of academic integrity requirements. Students’ academic careers have been significantly affected by a decision to cheat. Think about whether you want to join them before contemplating cheating, or before helping a friend to cheat.

You are welcome and encouraged to study and compare or discuss their implementations of the programming projects with any others after they are graded, **provided that** all of the students in question have received nonzero scores for that project assignment, and if that project will not be extended upon in a later project assignment.

10 Course topics and dates (SUBJECT TO CHANGE)

The following list of lecture topics will vary according to the pace of lecture.

- Administrative and course introduction (1 lecture)
- Scripting Languages (Ruby, 2 lectures)
 - Implicit vs. explicit declarations
 - Dynamic vs. static typing

- Text processing and string manipulation
- Regular expressions and finite automata (3 lectures)
- Functional programming (OCaml, 7 lectures)
 - Lists and recursion
 - Higher-order programming
 - Types and polymorphism
 - Data types and pattern matching
 - Modules
 - Closures
- Programming Language Theory (4 lectures)
- Environments, scoping, and binding (6 lectures)
 - Functions and procedures
 - Parameter passing mechanisms
 - Dynamic vs. static scoping
 - Run-time implementations
- Context-free grammars (2 lectures)
- Polymorphism and generics (2 lectures)
- Concurrency (Java, 4 lectures)
- Advanced Topics (7 lectures)
 - Garbage collection
 - Object representations
 - Historical overview

11 Right to change information

Although every effort has been made to be complete and accurate, unforeseen circumstances arising during the semester could require the adjustment of any material given here. Consequently, given due notice to students, the instructors reserve the right to change any information on this syllabus or in other course materials.

12 Copyright

All course materials are copyright Jeff Foster and Larry Herman © 2005. All rights reserved. Students are permitted to use course materials for their own personal use only. Course materials may not be distributed publicly or provided to others (excepting other students in the course), in any way or format.