1 Prerequisites and description

The prerequisites are a grade of C or better in CMSC 212 (or CMSC 214), and in CMSC 250.

This course is a study of programming languages, including their syntax, semantics, and implementation. Methods for describing language syntax to be covered include regular expressions, finite-state automata, and grammars. The semantics of programming languages and their runtime organization are covered. Different models of languages will be discussed, including procedural (e.g., C, Pascal), functional (e.g., OCaml, Scheme), rule–based (e.g., Prolog), and object–oriented (e.g., Ruby, Java, Smalltalk, C++). Language features such as scoping and binding of variables, higher–order programming, typing and type polymorphism, pointers, object inheritance, and exceptions are explored.

2 Resources

There are no required or recommended texts for this course this semester. Some students may find reference texts for the languages taught to be necessary. Suggestions and other sources of information will be provided during the semester.

The most important resource for finding information about the course is the class website: http://www.cs.umd.edu/class/summer2006/cmsc330/

Office hours schedule, lecture slides, links and documents will be available in the web site. Some documents may require a username and password, which are “student” and “330”.

Programming will be done on the OIT Grace Cluster, grace.umd.edu. Students will use their own Glue accounts to access the Grace cluster and do coursework, so students who don’t have a Glue account should request one online immediately at www.oit.umd.edu/new.

3 Grading

The course will have 3 exams, during discussion section time. The tentative dates are:
<table>
<thead>
<tr>
<th>Exam</th>
<th>Weight</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>Midterm 1</td>
<td>15%</td>
<td>Wed, June 21</td>
</tr>
<tr>
<td>Midterm 2</td>
<td>20%</td>
<td>Fri, July 7</td>
</tr>
<tr>
<td>Midterm 3</td>
<td>20%</td>
<td>Fri, July 28</td>
</tr>
</tbody>
</table>

The exams are individual, and closed books, closed notes. The exception is that you can bring one letter-size paper hand-written by yourself on one side. The other side should be blank. Feel free to write anything you want on this paper, but do not use it to replace proper studying. If you don’t understand the subject you wrote, you probably won’t have time to understand it during the exam.

The course will also have 5 programming projects. The total project grade is worth 45% of the final grade. All projects have the same weight. Each project will be due about one week after it is assigned. All projects are individual, and should be tested thoroughly in the grace cluster before submission.

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.

Regrading requests for the exams should be submitted in paper attached to the exam. Regrading requests for the projects can be done by email to the person who graded the project with the word “regrade” in the subject line. Regrade requests need to be submitted no longer than one week from the date the grade was available.

Note: this is a programming course, which teaches languages and concepts essential for later CMSC courses. As a result, the ability to submit working versions of the projects is necessary. Therefore, no student will be able to 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.

4 Excused absences and accommodations

4.1 Excused absences

Besides the policies in this syllabus, various University policies may apply to students during the semester. Various policies which may be relevant appear in the Undergraduate Catalog at [www.umd.edu/catalog](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 an 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 considered to be an excused absence. However, students requesting an excused absence for any reason must apply in writing and must furnish documentary support for the assertion that the absence qualifies as an excused absence. For an absence due to medical reasons, for example, documentation would be from a health care
professional who treated the student. In cases of illness simply being seen by a health professional is insufficient—medical documentation must state that a student was incapacitated and therefore unable to attend for an excused absence to be justified. For medical absences the documentation must include the phone number of the health care professional and must explicitly indicate the exact dates or times of incapacitation due to illness. The dates of incapacitation must include the date of the missed exam; an illness preceding an exam may not be considered to justify an excused absence. 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 student’s 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, and if a known conflict exists with one of the planned midterm dates appearing below, notice must be given prior to the end of the schedule adjustment period.

When a student has an excused absence for an exam the score will be determined either by averaging their scores for the other exams (possibly a weighted average), or by giving 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.

A student who might miss an exam for any reason other than those mentioned above must contact the instructor in advance to discuss the circumstances. An instructor is not under obligation to offer a substitute assignment or to give a student a makeup assessment 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.

4.2 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.
5 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.nethics.umd.edu/aug](http://www.nethics.umd.edu/aug).

Note that 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 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. Students 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 allowable questions 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 student’s project solution.

- When it comes to actually writing a project assignment, other than help from the instructional staff a project must solely and entirely be a student’s 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. Students may not discuss design of any part of a project with anyone except the instructor or teaching assistants. Examples of questions which students 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!” Students may not use any disallowed source of information in creating either their project design or code. When writing projects students are free to use ideas or short fragments of code from published textbooks or publicly available information, if the specific source is 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
instructor 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.

Students 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.

### 6 Course topics (SUBJECT TO CHANGE)

The following list of lecture topics may vary according to the pace of lecture:

- Administrative and course introduction
- Scripting languages (Ruby)
  - Implicit vs. explicit declarations
  - Dynamic vs. static typing
  - Text processing and string manipulation
  - Data structures in Ruby
  - Codeblocks in Ruby
- Regular expressions and finite automata
- Environments, scoping, and binding
  - Functions and procedures
  - Parameter passing mechanisms
  - Dynamic vs. static scoping
– Runtime implementations

• Functional programming (OCaml)
  – Lists and recursion
  – Higher-order programming
  – Types and polymorphism
  – Data types and pattern matching
  – Modules
  – Closures

• Lambda Calculus

• Grammars
  – Hierarchy of formal languages.
  – Regular grammars.
  – Context-free grammars.
  – Context-sensitive grammars

• Polymorphism and generics in Java

• Concurrency

• Pointers and garbage collection

• Exceptions

• History of programming languages

7 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 instructor reserves the right to change any information on this syllabus or in other course materials.

8 Copyright

All course materials are copyright 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.