CPS323 - PROGRAMMING LANGUAGES

Professor:	Russell C. Bjork	Spring Semester, 2008-2009
Office:	MacDonald 217 x4377	MWF 9:10-10:10 am
	bjork@gordon.edu	
Hours:	MWF 2:10-3:10 pm;	Blackboard course site plus public site:
	Th 1:30-4:30 pm and by appointment	http://www.cps.gordon.edu/courses/cps323

CATALOG DESCRIPTION:

Introduces formal study of programming language syntax, data types, and control structures; methods of executing higher-level constructs at run-time; data structures and algorithms used in compilation and interpretation. Laboratory work emphasizes acquisition of skill in numerous programming languages. Prerequisite: CPS212.

COURSE OBJECTIVES:

In general, this course is intended to equip you to make effective use of higher level programming languages - both those which now exist and those which will be developed in the future. Any scientific discipline is heavily dependent on having an adequate system of notation. As Computer Science continues to grow and develop, the programming languages we use to describe computation will of necessity have to evolve to keep pace.

In particular, upon completion of this course, you should be familiar with:

1. The design of and evaluation of programming languages - what needs have motivated the development of various languages, and what criteria can be used in evaluating a language design.
2. The description of programming language syntax and semantics - including syntactic metalanguages and various methods of semantic description.
3. The spectrum of programming language features - control structures, data types, and facilities for procedural and data abstraction.
4. Programming languages whose model of computation departs from the traditional VonNeumann model.
5. The run-time implementation of programming languages - how the various data and control structures are represented in memory.
6. The rudiments of the translation of programming languages.

Further, as part of accomplishing the above objectives, you will get a brief exposure to a number of programming languages that embody one or more important design concepts.

TEXT: Scott, Michael. *Programming Language Pragmatics* (2nd ed). (San Francisco: Morgan-Kaufmann, 2006).

There is also a lot of material on the web about various programming languages which you will need to explore as you do your projects.

COURSE TECHNIQUES AND PROCEDURES:

There are two general approaches that might be taken to a course on programming languages: a topical approach which takes various aspects of language design in turn, illustrating each by examples from various languages; or a language centered approach that focuses around the syntax and semantics of a number of different languages. Our class sessions will follow more of the first approach, with periodic case studies of a language that is relevant to, or illustrates, the topic on hand. Most case studies will be followed by a programming project using the language in question.

You should also note that this course is a bit more philosophical in nature than others you have taken. The merits and detriments of various approaches to language design are by no means a matter of universal agreement, especially when one considers different contexts in which languages are used.. As you proceed through the course, you should be seeking to form your own convictions about the various issues under discussion.

COURSE REQUIREMENTS AND EVALUATION:

1. You will be expected to read most of the textbook, as well as supplementary material online, as assigned in the topic schedule below. Reading assignments should be completed **BEFORE** the class hour in which the topic is discussed. Lecture presentations will assume that you have read the text, and it is expected that your participation in the class will reflect that fact. However, our classroom discussion will not rigidly follow the order of material in the text, nor will it be confined to material covered there, and some of the important material in the course will come from only one of these two sources.

2. Four homework sets will be distributed during the semester and will be due as shown in the course schedule. Note that these are fairly substantial assignments; you would do well to work on the problems as material is covered in class, rather than waiting until just before the set is due to tackle the whole assignment. Solutions to each homewor set will be posted outside the professor's door after the set is turned in. Together, these homework sets will account for 30% of the final course grade.

Set	Emphases
1)	Syntax and Semantics; Names, Scope, Binding
2)	Data Types; Data Abstraction; Variables, Expressions, Assignment
3)	Control Flow; Subroutines; Exceptions
4)	Concurrency; Compilation

   The following guidelines should be observed when doing these homework sets

   - Homework sets will be due at the start of class on the date indicated. Late homework sets will **NOT** be accepted.

   - Homework sets must be done on one side only of 8-1/2 x 11 paper, and pages must be stapled in problem-number order. Problems must be numbered, and final answers (where appropriate) should be highlighted. (Homework sets not conforming to these standards will be returned ungraded.)

   - You may work together with another student on homework, provided each of you works on each problem.

   - Where an exercise calls for writing a program, it is sufficient to write it out by hand; you need not enter it into the computer unless explicitly directed to do so.

3. To give you experience with a broad spectrum of languages, you will do four of five mini programming projects, plus one larger project (using Ada), as shown in the course schedule. Along with each project, you will turn in a short written evaluation of the language in light of the criteria discussed in the text and lecture. The focus of the paper should be on how the language's design helped or hindered the task of carrying out the assignment. Evaluations should be about 1 page long. Each mini-project will account for 5% of the final course grade, and the larger project will account for 10% (30% total).

   The grade for each assignment will be based on 20% for the project quiz, 20% for the written evaluation of the language, and 60% on the following basis:

- Program minimally fulfills the stated requirements, with a minimal grasp of the relevant features of the language involved: C (45 points).
- Program thoroughly fulfills the stated requirements, and shows a solid grasp of the features and "style" of the language (i.e. it is not just a translation of a program written in a more familiar language such as Java): A (60 points)
- Program lies somewhere in between "A" and "C" criteria: C+ .. A- as appropriate (46-54 points)

4. It is a premise of this course that students will continue to learn new languages throughout their professional careers. One of the most effective ways to learn a new language is to teach the language to other competent professionals. Thus, the final two language case studies (Python and Ruby) will be team taught by members of the class. (The team teaching the language will not do the programming assignment for that language).

The team will present case study lectures for the other students of the class and, after consultation with the professor, will assign a mini programming project (comparable in size to the other mini projects) and project quiz. The team will provide the following in the course of their lectures and will submit same via web site (top-level page named `index.html`) submitted electronically:

- Description of the purpose/domain of the language
- Sample programs for walkthrough purposes
- Access to online reference sources
- Description of key features of the languages, to include, but not limited to:

  Identifiers
  Reserved words
  Data types supported (Constants, Primitive, User-defined)
  Data structures supported
  Expressions
  Control structures
  Input/Output facilities

- Directions for using the development tools on our Linux workstations.

IMPORTANT NOTE! It is likely that many resources will be used to create this web site; both traditional resources and online resources will abound. Your web site *must* document all sources used. No text or images from other sources may be included without explicit attribution. This is extremely important.

The team's grade for this process will account for 5% of the final course grade for each individual, assuming roughly equal contribution by each individual on the team.

5. There will be a midterm examination plus a final examination given as shown in the course schedule below. The midterm will account for 15% of the final course grade, and the final will account for 20% (total 35%). Exams will be open book, open notes.

6. Summary:

Four Homework sets	30%	
Programming projects	30%	
Teaching a Language to the Class	5%	
Midterm and Final Exams	35%	
	100%	

7. Your final grade will be computed on the basis of a weighted sum of the items listed above.

The following are minimum guaranteed grades for the percentages indicated:

	93% - 100%: A	90% - 92.9%: A-
87% - 89.9%: B+	83% - 86.9%: B	80% - 82.9%: B-
77% - 79.9%: C+	73% - 76.9%: C	70% - 72.9%: C-
67% - 69.9%: D+	63% - 66.9%: D	60% - 62.9%: D-

POLICY STATEMENT ON EXTENSIONS AND INCOMPLETES:

   1. Extensions of the due dates for homework or projects will be given in the event of extenuating circumstances (such as illness, personal emergency) IF you submit a brief written request to the professor as soon as possible after the circumstances arise. This request will be initialed (if approved) and will be returned to you. You must attach it to the piece of work for which the extension was granted.

2. A grade of Incomplete will be given without penalty IF you are unable to complete the course work by the last day of the term due to major illness or other similar emergency. Again, a written request should be submitted. Such a request will only be granted if you are substantially up-to-date with your course work and were making good progress in the course up to the time that the difficulty arose. Of course, you must complete all work for the course by the midpoint of the next semester in accordance with College policy.

3. A grade of Incomplete with a penalty of one letter grade to be applied in the final grade computation MAY be given if you are unable to complete all the course work for reasons other than those noted above. You must make a written request, and your progress in the course, class attendance etc. will be taken into consideration in determining whether to grant it. Again, you must complete all work for the course by the midpoint of the next semester.

ATTENDANCE POLICY:

Regular class attendance is expected of all students, and class attendance will be recorded. Absences from class will be classified as "documented" or "undocumented". A documented absence is one where written documentation is submitted supporting an absence from class due to circumstances beyond the student's control. An undocumented absence is any other absence, including one which could qualify as documented if proper documentation were submitted.

Students who have more than three absences (of any kind) during the semester should expect to see their final grade reduced by 1% for every undocumented absence and students who have more than 12 undocumented absences will fail the course automatically. Note that it is not necessary to document absences unless there are more than three total absences; for most students, this will avoid the need to submit documentation.   A student who anticipates the need to miss more than three classes due to athletic competitions or other student activities should review the college's attendance policy on page 31 of the catalog, and should then discuss alternatives to class attendance with the professor at the start of the semester.

A student who is habitually late will have late arrival for class counted as a half absence for that class, and a student who sleeps through most or all of a given class session will be counted as absent for that class.

You may ask the professor to waive this policy for you if you earned an A in the prerequisite course, or if you have an A average in this course as of the mid-term exam. If you wish to take advantage of this exemption, you must so inform the professor. However, the attendance policy will be reimposed if your subsequent work deteriorates.

STUDENTS WITH DISABILITIES:

Gordon College is committed to assisting students with documented disabilities (see Academic Catalog Appendix C, for documentation guidelines). A student with a disability who may need academic accommodations should follow this procedure:

1. Meet with a staff person from the Academic Support Center (Jenks 412 X4746) to:
    a. make sure documentation of your disability is on file in the ASC,
    b. discuss the accommodations for which you are eligible,
    c. discuss the procedures for obtaining the accommodations, and
    d. obtain a **Faculty Notification Form**.

2. Deliver a Faculty Notification Form to each course professor *within the first full week of the semester;* at that time make an appointment to discuss your needs with each professor.

Failure to register in time with your professor and the ASC may compromise our ability to provide the accommodations. Questions or disputes about accommodations should be immediately referred to the Academic Support Center. See Grievance Procedures available from the ASC.

TENTATIVE SCHEDULE OF TOPICS:

Date	Topic(s)	Reading	Homework/ Exams / Projects Due
W 1/14	Course Introduction	§1.1-1.3	
F 1/16	50 years of Language History; Language Paradigms	§1.4-1.5	
W 1/21	Language Evaluation Criteria		
F 1/23	(ctd)		
M 1/26	Case Study: FORTRAN		
W 1/28	Specifying Syntax	§ 2.1	
F 1/30	(ctd)		
M 2/2	Names, Scope, and Binding	ch. 3	
W 2/4	(ctd)		
F 2/6	(ctd)		
M 2/9	Data Types	ch. 7	**FORTRAN Mini-Project Due**
W 2/11	(ctd)		
F 2/13	(ctd)		
M 2/16	Data Abstraction and Object-Oriented Languages; Case Study: Ada	ch. 9	**HW #1 DUE**
W 2/18	(ctd)		
F 2/20	(ctd)		
M 2/23	Variables, Expressions and Assignment	§6.1	
W 2/25	(ctd)		

Date	Topic(s)	Reading	Homework/ Exams / Projects Due
F 2/27	(ctd)		**HW #2 DUE**
M 3/2	Review and Catch up		
W 3/4	**MIDTERM EXAMINATION (though**		

M 2/23	Variables, Expressions and Assignment	§6.1	

F 2/27	(ctd)		**HW #2 DUE**
M 3/2	Review and Catch up		
W 3/4	**MIDTERM EXAMINATION (though Variables/Expression/Assignment)**		
	Spring Break		
M 3/16	Functional Languages; Case Study: LISP	ch. 10	
W 3/18	(ctd)		**Ada Project Due**
F 3/20	Control Flow	Introduction to ch. 6; §6.2-6.8; Dijkstra letter	
M 3/23	(ctd)		
W 3/25	(ctd)		
F 3/27	Logic Languages; Case Study: PROLOG	ch. 11	
M 3/30	(ctd)		**LISP Mini-Project Due**
W 4/1	Subroutines and Control Abstraction; Exceptions	ch. 8 (including 8.4.4 on CD)	
F 4/3	(ctd)		
M 4/6	(ctd)		
W 4/8	Scripting Languages	ch. 13	
	Easter Break		
W 4/15	Case Study: Python	Python material on web	**PROLOG Mini-Project Due**
F 4/17	(ctd)		
M 4/20	Case Study: Ruby	Ruby material on web	**HW #3 DUE**
W 4/22	(ctd)		
F 4/24	Concurrency	ch. 12	
M 4/27	Introduction to Programming Language Translation	§1.6-1.7	
W 4/29	Scanning and Parsing	§ 2.2-2.3	
F 5/1	Analyzing Semantics	ch. 4	**Python or Ruby Mini-Project Due**
M 5/4	Building a Runnable Program	ch. 14	
W 5/6	Code Improvement	ch. 15 (all on CD)	
Th 5/7	By 4 PM on Reading Day		**HW #4 DUE**

F 5/8 - 8-10 AM - FINAL EXAMINATION