CS 3360: Design and Implementation of Programming Languages
Spring 2017

CRN: 22284
Lecture: TR 1:30-2:50 pm in CCSB G.0208
Website: http://www.cs.utep.edu/cheon/cs3360
Instructor: Yoonsik Cheon (x-8028, ycheon@utep.edu); office hours: TR 3:00-4:30 pm in CCSB 3.0606
TA: TBA
Prerequisite: CS 2302 with a grade of C or better (Recommended: CS 3331 and CS 3432)

Course Goals

In this course we will study concepts and examples of programming languages with the goal of acquiring the tools necessary for critical evaluation and rapid mastery of programming languages and constructs.

The course attempts to balance theory and hands-on experience. We will survey the constructs and capabilities typically found in modern programming languages with attention to design trade-offs and implementation considerations. By gaining an understanding on the range of possibilities likely to be encountered in a language, students will be prepared to learn new languages quickly throughout their careers. By understanding the implications of design alternatives, students will be better able to anticipate the problems likely to arise in using a new language. Also, the presentation of design alternatives and trade-offs lays the groundwork for future advanced study of compilers and programming language semantics. To instantiate the discussion of general programming language characteristics, several languages will be presented in more detail: e.g., AspectJ (an aspect and object-oriented language), Haskell (a functional language), Prolog (a logic-programming language), and PHP (a Web scripting language). Students will gain practical experience with each programming paradigm by completing a programming project in each of the chosen languages.

Upon successful completion of this course, students will be able to:

- [Level 3: *Synthesis and evaluation*] Evaluate modern, representative programming languages critically
- [Level 3] Choose a suitable programming paradigm and language for a given problem or domain
- [Level 3] Design a small, domain-specific programming language, e.g., a test script language, by defining its formal syntax and semantics
- [Level 2: *Application and analysis*] Define syntax of a small context-free grammar in BNF and EBNF
- [Level 2] Use attribute grammars to describe the static semantics of small programming languages
- [Level 2] Define dynamic semantics of small subsets of programming languages, e.g., control structures
- [Level 2] Select and apply appropriate expressions and control structures for a given programming task
- [Level 2] Analyze and evaluate data and control abstractions of programming languages
- [Level 1: *Knowledge and comprehension*] Recognize major programming languages
- [Level 1] Explain operational semantics, axiomatic semantics, and denotational semantics as methods of expressing programming language semantics
- [Level 1] Explain design concepts, design alternatives and trade-offs, and implementation considerations for scope, binding, data types, expressions, control structures, subprograms, abstract data types, objects, concurrency structures, and exception handling in modern programming languages

Textbooks

The course textbook is Robert W. Sebesta, *Concepts of Programming Languages*, 11th edition, Addison Wesley, 2015. The textbook should be available at the UTEP bookstore, and you are expected to acquire a copy for your use in this course, as reading assignments will be taken from the textbook. In addition, the following books are recommended for references:

Rasmus Lerdorf, Peter MacIntyre and Kevin Tatroe, *Programming PHP*, 3rd edition, O'Reilly, 2013 (Chapters 2-6).

Robin Nixon, *Learning PHP, MySQL & JavaScript*, 4th edition, O'Reilly, 2015 (Chapters 3-6).

Joe Gradecki, *Mastering AspectJ: Aspect-Oriented Programming in Java*, Wiley, 2003.

Alejandro Seraano Mena, *Beginning Haskell, A Project-Based Approach*, Apress, 2014.

Electronic books of the recommended references are available for free through UTEP Library from UTEP domain; use VPN from outside UTEP domain (see course website). Other supplemental readings will be taken from the Internet or hard copies will be handed out in class.

Assignments

There will be three types of assignments: *reading*, *written*, and *programming*. All assignments will be handed out or announced in class (see page 4 for planned assignments). If you miss a class session, it is your responsibility to find out what you missed. You should expect to spend about 3-4 hours per week for reading and written homework assignments, and an average of 3 hours per week for programming assignments. Note, however, that each programming assignment is estimated to require 8-10 hours, so your work load in weeks that programming assignments are due may be higher (with other weeks being correspondingly lower).

Reading assignments ask to read the textbook and prepare for the coming week's lectures; you will have quizzes at the beginning of classes. Written assignments ask you do exercises from the textbook; exercises that use material not discussed in lecture will be graded generously. Programming assignments are designed to allow you to gain hands-on experience with a specific language and programming paradigm (PHP for Web scripting, AspectJ for aspect-oriented programming, Haskell for functional programming, and Prolog for logic programming). For all assignments, no late submission will be accepted unless arrangements have been made in advance or unless unusual circumstances warrant an exception. All assignments are to be done individually. While you may discuss the assignment in general terms with others, your solutions should be composed, designed, written and tested by yourself alone. If you need help, consult the TA or the instructor.

Exams

There will be two exams: *mid-term* and *final*. The final exam may be comprehensive. The mid-term exams will take place during the regular class session and each will be 80 minutes in length, and the final exam will take place on the date specified by the university (see page 4 for tentative exam dates). Make-up exams will be given only when you have unusual circumstances, such as incapacitating illness or presenting a research paper at a conference. If you believe that you have an unusual circumstance that warrants a make-up exam, notify us as soon as possible. If you will be attending a conference or other event, you must make arrangements for a make-up exam *in advance*. Under all circumstances, you are required to provide official documentation before a make-up will be administered.

Grading

Your semester grade will be based on a combination of homework assignments, programming projects, quizzes, and exams. The approximate percentages are as follows:

Homework and quizzes:	35%
Programming assignments:	30%
Exams:	35%

In addition, a bonus of up to 5% is available for lecture attendance and participation. To earn this bonus, you must arrive at lecture on time and participate in class discussion in a constructive and prepared manner, e.g., by asking or answering questions that demonstrate that you have read and attempted to understand the material.
Your grade is independent of anyone else's grade in this class; that is, we do not grade on a curve. Everyone can get an A in this class. Our purpose in grading is to uphold a standard of quality and to give you feedback, it is not to rank students. Although we will not always make fine distinctions in points, the nominal minimum standards are:

90% or higher is an A
80-89% is a B
70-79% is a C

60-69% is a D
below 60% is an F

The instructor reserves the right to adjust these criteria downward, e.g., so that 88% or higher represents an A, based on overall class performance. The criteria will not be adjusted upward, however.

Attendance

Lecture attendance is required; you should understand that your success in the course will improve greatly by attending classes regularly. The instructor reserves the right to penalize unexcused absences; e.g., your final grade may be lowered by one point for each unexcused absence above three. The following is excerpted from the 2015-2016 Undergraduate Catalog.

> The student is expected to attend all classes and laboratory sessions. It is the responsibility of the student to inform each instructor of extended absences. When, in the judgment of the instructor, a student has been absent to such a degree as to impair his or her status relative to credit for the course, the instructor can drop the student from the class with a grade of W before the course drop deadline and with a grade of F after the course drop deadline.

Standards of Conduct

You are expected to conduct yourself in a professional and courteous manner, as prescribed by the Handbook of Operating Procedures: Student Conduct and Discipline. All graded work (homework, projects, exams) is to be completed independently and should be unmistakably your own work, although you may discuss your work with others in a general way. You may not represent as your own work material that is transcribed or copied from another source, including persons, books, or Web pages. "Plagiarism" means the appropriation, buying, receiving as a gift, or obtaining by any means another's work and the unacknowledged submission or incorporation of it in one's own academic work offered for credit, or using work in a paper or assignment for which the student had received credit in another course without direct permission of all involved instructors. Plagiarism is a serious violation of university policy and will not be tolerated. All cases of suspected plagiarism will be reported to the Dean of Students for further review.

Disabilities

If you have a disability and need classroom accommodations, please contact The Center for Accommodations and Support Services (CASS) at 747-5148, or by email to cass@utep.edu, or visit their office located in UTEP Union East, Room 106. For additional information, please visit the CASS website at www.sa.utep.edu/cass.

Schedule

The following table shows a planned schedule for the course; refer to the course website for an up-to-date schedule.

Dates		Topics	Readings	Assignments
Week 1	Jan. 17, 19	Introduction Preliminaries	Handout Chap 1	
Week 2	Jan. 24, 26	Describing syntax	Sec 3.1-3.3	Homework 1
Week 3	Jan. 31, Feb. 2	Describing syntax Attribute grammar	Sec 3.4	
Week 4	Feb. 7, 9	Web scripting Intro to PHP	[Lerdorf-etal13] [Nixon15]	Project 1
Week 5	Feb. 14, 16	PHP		
Week 6	Feb. 21, 23	Lab/demo (PHP) Names, binding, types, and scopes	Chap 5	Homework 2
Week 7	Feb. 28, Mar. 2	Names, binding, types, and scopes Data types	Sec 6.1-6.9	
Week 8	Mar. 7, 9	**Exam 1** Object-oriented programming	Sec 12.1-12.6	Homework 3
Week 9	Mar. 14, 16	Spring break		
Week 10	Mar. 21, 23	Aspect-oriented programming Intro to AspectJ	[Gradecki03]	Project 2
Week 11	Mar. 28, 30	AspectJ		
Week 12	Apr. 4, 6	Lab/demo (AspectJ) Functional programming	Sec 15.1-15.3	
Week 13	Apr. 11, 13	Intro to Haskell Haskell	Sec 15.8 [Mena14]	Project 3
Week 14	Apr. 18, 20	Haskell Lab/demo (Haskell)		
Week 15	Apr. 25, 27	Describing semantics Subprograms	Sec 3.5 Sec 9.1-9.6	
Week 16	May 2, 4	Logic programming and Prolog	Chap 16	
Week 17	May 11	**Final** at 1:00–3:45 pm		

Important Dates

January 16:	Martin Luther King, Jr. day (university closed)
January 17:	Classes begin
February 1:	Census day
March 9:	Exam 1
March 13-17:	Spring break (no classes)
March 25:	Study day (no classes)
March 30:	Course drop/withdrawal deadline
March 31:	Cesar Chavez birthday - observance (no classes)
May 4:	Last day of classes
May 5:	Dead day
May 11:	Final on Thursday at 1:00–3:45 pm