Programming Languages Course Syllabus

Andrés Gómez de Silva Garza

Programming languages are fundamental in the implementation of algorithms. There exist languages with varying characteristics, designed with different objectives in mind, which nonetheless share properties from a theoretical, conceptual point of view. In this course students will study the theory behind programming languages and several case studies of specific languages that exemplify diverse programming paradigms. Throughout the course, comparisons and contrasts will be made between languages from the theoretical point of view, enabling students' understanding of the reasons behind the design of such a diversity of programming languages and paradigms. The topics to be covered (roughly in the order they are listed) are:

1. **Introduction.** History of programming languages and paradigms. Some criteria for comparing and contrasting languages.
2. **Syntax.** Grammars. BNF. Parsing.
3. **Semantics.** Type checking. Subprograms and parameter passing mechanisms. Variable scope. Overloading. Operational semantics. Denotational semantics. Axiomatic semantics.
4. **Pragmatics.** Memory management. Garbage collection. Exception handling.
5. **Functional programming languages.** Examples of possible case studies: ML, Miranda, Scheme, Haskell, LISP.
6. **Logic programming languages.** Examples of possible case studies: Gödel, Janus, PROLOG.
7. **Parallel/concurrent programming languages.** Examples of possible case studies: MODULA-2, Ada, Occam.
8. **Business-oriented programming languages.** Examples of possible case studies: COBOL, RPG, PL/I.
9. **Other programming languages.** Examples of possible case studies: SNOBOL, APL, XL, REBOL, Smalltalk, Forth, AspectJ, Moo, Piet.

Bibliography

1. Alice E. Fischer & Frances S. Gordzinsky. The Anatomy of Programming Languages. Prentice Hall. 1992. ISBN-10: 0130351555. ISBN-13: 978-0130351555.

2. Daniel P. Friedman & Mitchell Wand. *Essentials of Programming Languages (3rd edition)*. MIT Press. 2008. ISBN-10: 0262062798. ISBN-13: 978-0262062794. [ITAM's library has the second edition. Call number: 005.13 F911E 2001]

3. Kenneth C. Louden. *Programming Languages: Principles and Practice (3rd edition)*. Course Technology. 2011. ISBN-10: 1111529418. ISBN-13: 978-1111529413. [ITAM's library has the second edition (call number: 005.13 L887P 2003) and the third edition (call number: 005.13 L886P 2012)]

4. Terrence W. Pratt & Marvin V. Zelkowitz. *Programming Languages: Design and Implementation (4th edition)*. Prentice Hall. 2000. ISBN-10: 0130276782. ISBN-13: 978-0130276780. [ITAM's library has it. Call number: 005.13 P917P 2001]

5. Robert W. Sebesta. *Concepts of Programming Languages (10th edition)*. Addison Wesley. 2009. ISBN-10: 0136073476. ISBN-13: 978-0136073475. [ITAM's library has the third edition. Call number: 005.13 S443C 1996]

6. Ravi Sethi. *Programming Languages: Concepts and Constructs (2nd edition)*. Addison Wesley. 1996. ISBN-10: 0201590654. ISBN-13: 978-0201590654. [ITAM's library has it. Call number: 005.13 S495P 1996]

7. Franklyn Turbak, David Gifford, & Mark A. Sheldon. *Design Concepts in Programming Languages*. MIT Press. 2008. ISBN-10: 0262201755. ISBN-13: 978-0262201759. [ITAM's library has it. Call number: 005.13 T931D]

Evaluation method

Surprise (and non-surprise) quizzes: 35%
Language research project (written analysis/description of language plus classroom-style tutorial): 50%
Quality of designed quiz: 15%
(There will be no final exam.)

Quizzes

Short surprise quizzes will be applied sporadically throughout the semester. These may occur at any time during a class. If you are not present at the time a quiz is applied, there are no opportunities for a re-take, and your grade in that quiz will be 0.

Language research project

Each individual student or team of students will be asked to select a specific language (from the lists of examples given in topics 5-9 above, or similar ones) as a case study. The student or team will have to prepare a written report on their selected language. The structure and content of the report is expected to be written and designed as if it were a chapter for a textbook on programming languages covering the specific case study language. The student or team will also have to give an oral presentation on their selected language (the structure and content of which should roughly be the same as the written report…it should be thought of as teaching a quick class/tutorial on the selected language, its design features, and how to program in it). Each of these two components will be graded by the course professor using a rubric that will be distributed later on in the semester, and will count towards 50% of the "research project" part which counts towards 50% of the final course grade.

The student or team will also have to design and apply, at the end of the last class session in which the tutorial on their selected language is taught, a short quiz on the language. The quality of this quiz will be assessed by the course professor (criteria that will be applied include wideness of coverage, easy/difficulty for takers to respond in the available time, quickness with which final grades are reported by quiz designers, and distribution of grades obtained by takers, and are explained in further detail in a separate document) and counts towards 15% of the final course grade. The rest of the students in the class (those that did not participate in designing these quizzes) will have to take the quiz, under the same conditions as mentioned in the previous section above.

The average of the grades obtained in all professor- and classmate-designed quizzes, where each quiz will be given the same weight irrespective of its designer, will form the item described above as "surprise quizzes," which counts towards 35% of the final course grade.