

BIL 424: PROGRAMMING LANGUAGES

Spring 2012

Instructor: Yrd. Doc. Dr. Ahmet Sayar

E-mail: ahmet.sayar@kocaeli.edu.tr

Office hours: By appointment or after the classes

Class web page: <http://www.ahmetsayar.com/lecturenotes/>

Students should normally contact me via email. Be sure to include the name of the course in the subject. Every effort will be made to respond to email within 24 hours except for weekends and holidays. Please use your university email account.

Day & Time: 1st Ed: Wednesday 14:00-16:50 (Amfi-B) 2nd Ed: Tuesday 19:45-22:25 (Amfi-B)

Aim: The purposes of this course are to analyze and evaluate important features found in a variety of programming languages, to study formalisms for specifying language syntax and semantics, and to gain understanding of the important programming language paradigms.

Course Description: This course covers study of various programming language paradigms: Imperative, Object Oriented, Functional, and Relational, with emphasis on advanced programming topics such as defining syntax and semantics, lexical and syntax analysis, type systems, binding rules and context-free higher order programming.

Textbooks:

- Robert W. Sebesta, Concepts of Programming Languages, 5th or 6th ed., Addison-Wesley, 2003.
- Not all the material we cover will be from the book, nor will the order in which we cover this material be the same as in the book. You will have to refer to copies of the slides (which I will distribute in class and on the course web page), your own notes, as well as the book in order to keep up with the course.

Resources:

- Michael Scott, Programming Language Pragmatics, 3rd ed., Morgan Kaufmann Publishers, 2009.
- Daniel P. Friedman, Mitchell Wand, and Christopher T. Haynes *Essentials of Programming Languages 2nd ed.*, MIT Press, 2001.

Grading: (tentative)

Midterm (%40)

Final exam (%60)

Course Learning Objectives/Outcomes:

1. Understand the need for different programming languages.

2. How names (identifiers) in programming languages are processed, validated and given meaning.
3. Describe how static memory and dynamic memory are allocated and reclaimed in typical runtime environments.
4. Choose a suitable programming paradigm and language for a given problem or domain.
5. Design a small, domain-specific programming language, e.g., a test script language, by defining its formal syntax and semantics.
6. Define syntax of a small context-free grammar in BNF and EBNF.
7. Use attribute grammars to describe the static semantics of small programming languages.
8. Define dynamic semantics of small subsets of programming languages, e.g., control structures.
9. Explain operational semantics, axiomatic semantics, and denotational semantics as methods of expressing programming language semantics.

Class Schedule:

1.week: Introduction, preliminaries	9.week: Midterm
2.week: Language creation, language design	10.week: Syntax and semantics studies
3.week: Student presentations - 1	11.week: Names, Bindings, Type Checking, Scopes
4.week: Student presentations - 1	12.week: Data Types
5.week: Compilation vs. interpretation	13.week: Expressions and Assignment Statements
6.week: Functional programming	14.week: Statement-Level Control Structures
7.week: Historical view of languages	15.week: Logic Programming Languages
8.week: Describing Syntax and Semantics	16.week: Overview of the course

Academic Dishonesty:

Cheating will not be tolerated and may result in serious sanctions, including immediate failure in the course. Serious incidents of academic dishonesty will also be brought to the attention of the university and may result in expulsion. All work in this class is meant to be an individual effort by the person receiving the grade. Any variation from this is considered cheating and all parties involved (giving or receiving) will be sanctioned.