

CMPS 461
Programming Language Concepts
Fall Semester 2007
Required Course in Computer Engineering

Catalog Data:	Programming Language Concepts (3) Fundamental concepts of programming language design, specification, and implementation; programming language paradigms and features; program verification. Prerequisite: CMPS 221; CMPS 360.
Typical Textbook:	Michael L. Scott. <i>Programming Language Pragmatics</i> . Morgan Kaufmann 2006. Reference: Online resources including additional reading, program code, documentation and other material.
Course Objectives:	This course introduces students to a wide range of programming language concepts and principles, as well as several language paradigms.
Primary Course Outcomes:	<ul style="list-style-type: none">• Interpret and design language specifications.• Analyze programs' storage behavior and requirements.• Develop software in various language paradigms.
Relationship to Undergraduate Program Outcomes:	CMPS 461 supports the following Program Outcomes: <ul style="list-style-type: none">• Analyze code for correctness.• Demonstrate independent learning by using unfamiliar computer systems and software tools to solve technical problems.• Be able to discuss major trends in industry and current research activities within the discipline.
Required Topics:	Overview of the course (1 hour) Language Specification (5 hours) Datatypes and Type Checking (3 hours) Run-time Storage Organization (6 hours) Heap Management Garbage Collection (3 hours) Functional Programming (6 hours) Principles of Induction (3 hours) Polymorphism and Type Inference (3 hours) Logic Programming (6 hours) Object Oriented Programming (6 hours)
Class Format:	Three lectures per week; each lecture is 50 minutes long.
Professional Component:	CMPS 461 introduces students to a wide range of programming idioms and to several topics in the formal treatment of software. Since all four of the major programming language paradigms are covered (imperative, object-oriented, functional, and logic), most new programming and specification languages that graduates will later see will be understandable to them as being built on various components of what they have seen in this course. Basic engineering principles of sound design and correctness are addressed in this course by introducing them to notions of declarative specifications (grammars, type specifications, inductive definitions), declarative programming (functional programming, logic programming), and important notions of reasoning about code (inductive proofs). Computer usage: Students are required to design and implement several programming projects. Projects are conducted on an open lab basis.
Evaluation:	