CSE 3301 Programming Language Fundamentals:

Summer 2006

Administrative Details

Monday, July 17 2006

Bill Kapralos

Course Preliminaries (1):

Instructor:	Bill Kapralos
Email:	billk@cs.yorku.ca
Office hours:	Tuesday 4:00-5:00pm Room CSE 2015
	Thursday 4:00-5:00pm Room CSE 2015
Textbook:	Programming Language Pragmatics. 2nd edition. By Michael L. Scott. 2006, Morgan Kaufmann Publishers. ISBN: 0-12-633951-1

Course Preliminaries (2):

Official course website:

http://www.cs.yorku.ca/~billk/cse3301_S06/generalInfo.html

Check course website regularly as all course related

information will be conveyed via the course website, including

assignments, announcements etc.

Course Preliminaries (3):

Tentative Grading Scheme:

Mid-term test	30%
Final test	30%
Exercises	40%

There will be three exercises:

Exercise One	10%
Exercise Two	15%
Exercise Three	15%

Prism Lab (1):

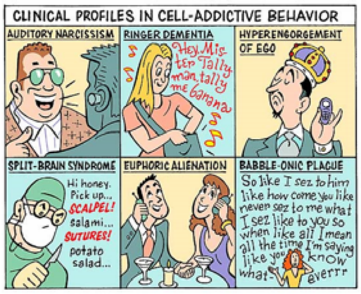
- **Course is Essentially Theoretically Based**
  - No explicit need for the lab is required
    - Won't really need to produce any significant code → although you may have to produce some code fragments etc. and a computer may be useful…
  - If you do not have a Prism account, you need to contact the Tech staff
    - Do so immediately so you can access the course website etc. → of course, you can also access the website on your own computer from home…

Obtaining Help (1):

- **Various Ways to Obtain Help**
  - Instructor office hours → please take advantage of the opportunity to approach me!
  - Instructor will also respond to e-mail questions at times other than his/her office hours, although the reply will necessarily come after a short delayed → make use of e-mail!

A Notes (1):

- **Cell Phones OFF!**
  - Please keep your cell phones from ringing during the lecture!



Course Objectives (1):

- **Course Description**
  - The topic of programming languages is an important and rapidly changing area of computer science. This course introduces students to the basic concepts and terminology used to describe programming languages. Instead of studying particular programming languages, the course focuses on the "linguistics" of programming languages, that is, on the common, unifying themes that are relevant to programming languages in general. The algorithmic, or procedural, programming languages are particularly emphasized. Examples are drawn from early and contemporary programming languages, including Fortran, Algol 60, PL/I, Algol 68, Pascal, C, C++, Eiffel, Ada and Java.

Course Objectives (2):

- **Topics May Include**
  - Classification of programming languages: language levels, language generations, language paradigms.
  - Programming language specification: lexical, syntactic, and semantic levels of language definition.
  - Data, data types, and type systems; simple types, structured types, type composition rules.
  - Control primitives, control structures, control composition rules.
  - Subprograms: functions and procedures; argument-parameter binding; overloading.
  - Global program structure: modules, generic units, tasks, exceptions.
  - Object-oriented language features: classes, encapsulation, inheritance, polymorphism.
  - Critical and comparative evaluation of programming languages.

Course Objectives (3):

- **Course Prerequisites**
  - General prerequisites including CSE 2001 3.0
    - Not a bad idea to review some of the topics related to DFAs and regular languages from CSE 2001
    - Relevant to scanning and parsing and initially, I was planning to spend a considerable amount of time to cover these topics but I decided not too!

Course Objectives (4):

- **Important Note (cont.)**
  - Keep in mind that this course is a full-semester course condensed into three weeks!
    - This means your schedule for the next three weeks will be hectic!
    - You must ensure you study → read the book, read my lecture notes etc. regularly as the tests and assignment due dates are fast approaching!

Course Content ("Roadmap") (1):

- **Material We Will Cover**
  - Keep in mind that due to the nature of the course (e.g., condensed), the topics may vary slightly
    - We may not cover all the material listed in the following "roadmap" slides
    - We may cover material not listed altogether

Course Content ("Roadmap") (2):

- **Chapter One:**
  - Introduction
    - The art of language design
    - The programming language spectrum
    - Why study programming languages
    - Compilation and interpretation
    - Programming environments
    - Over of compilation

Course Content ("Roadmap") (3):

- **Chapter Two:**
  - Programming Language Syntax
    - Specifying syntax
- **Chapter Four:**
  - Names, Scopes and Bindings
    - Object lifetime and storage management
    - Scope rules
    - Implementing scope
    - Binding of referencing environments
    - Binding within a scope
    - Separate compilation

Course Content ("Roadmap") (4):

- **Chapter Five:**
  - Target Machine Architecture
    - Memory hierarchy
    - Data representation
    - Instruction set architecture
    - Compiling for modern processors

Course Content ("Roadmap") (5):

- **Chapter Six:**
  - Control Flow
    - Expression evaluation
    - Structured and unstructured flow
    - Sequencing
    - Selection
    - Iteration
    - Recursion
    - Nondeterminacy

Course Content ("Roadmap") (6):

- **Chapter Seven:**
  - Data Types
    - Type systems
    - Type checking
    - Records (structures) and Variants (Unions)
    - Arrays, strings
    - Sets
    - Pointers and recursive types
    - Lists
    - File I/O
    - Equality testing and assignment

Course Content ("Roadmap") (7):

- **Chapter Eight:**
  - Subroutines and Control Abstraction
    - Review of stack layout
    - Calling sequences
    - Parameter passing
    - Generic subroutines and modules
    - Exception handling
    - Coroutines

Course Content ("Roadmap") (8):

- **Chapter Nine:**
  - Data Abstraction and Object Orientation
    - Object oriented programming
    - Encapsulation and inheritance
    - Initialization and finalization
    - Dynamic method binding
    - Multiple inheritance

Course Content ("Roadmap") (9):

- **Chapter Twelve:**
  - Concurrency
    - Background and Motivation
    - Concurrent programming fundamentals
    - Shared memory
    - Message passing

Course Content ("Roadmap") (10):

- **Chapter Thirteen (time permitting):**
  - Scripting Languages
    - What is a scripting language
    - Problem domains
    - Scripting the world wide web
    - Innovative features

Course Content ("Roadmap") (11):

- **Chapter Fourteen (time permitting):**
  - Building a Runnable Program
    - Back-end compiler structure
    - Intermediate forms
    - Code generation
    - Address space organization
    - Assembly
    - Linking
    - Dynamic linking

Course Content ("Roadmap") (12):

- **Chapter Fifteen (time permitting):**
  - Code Improvement
    - Phases of code improvement
    - Peephole optimization
    - Global redundancy and data flow analysis
    - Loop improvements I
    - Instruction scheduling
    - Loop improvements II
    - Register allocation