Computing Science Course Outlines  

CMPT 383 - D100 Comparative Programming Languages

Instructor(s): Gregory Baker

SFU Burnaby

Calendar Objective/Description:

Various concepts and principles underlying the design and use of modern programming languages are considered in the context of procedural, object-oriented, functional and logic programming languages. Topics include data and control structuring constructs, facilities for modularity and data abstraction, polymorphism, syntax, and formal semantics.

Instructor’s Objectives:

The objective of this course is to give the student a better understanding of non-imperative programming, and other important distinctions between languages. Various concepts and principles underlying the design and use of modern programming languages are considered. We will take a detailed look at a pure functional programming language, and a language that promotes concurrency.

Prerequisites:

CMPT 225, and (MACM 101 or (ENSC 251 and ENSC 252)).

Topics:

- Expressing algorithms functionally
- Functional programming in Haskell
- Type systems in programming languages
- Compilers, interpreters, and runtime environments
- Challenges and techniques in concurrent programming
- Concurrent programming in Go

Grading:

Weekly exercises 15%; assignments 35%; midterm exam 10%; final exam 40%.

Students must attain an overall passing grade on the weighted average of exams in the course in order to obtain a clear pass (C- or better).

Recommended Books:

Programming in Haskell, Graham Hutton, Cambridge University Press, 9781316626221

Academic Honesty Statement:

Academic honesty plays a key role in our efforts to maintain a high standard of academic excellence and integrity. Students are advised that ALL acts of intellectual dishonesty will be handled in accordance with the SFU Academic Honesty and Student Conduct Policies ( http://www.sfu.ca/policies/gazette/student.html ).