CISC 3150 Object-Oriented Programming

3 hours; 3 credits

Principles and implementation issues in object-oriented programming languages, including: memory and run-time models; encapsulation, inheritance and polymorphism; generics. Collections and other frameworks and hierarchies. Effects of binding time considerations on language design and implementation. Introduction to design patterns, such as adapter, singleton, and model-view-controller. Formal design specifications such as UML. Case studies chosen from multiple languages such as C++, Java and Smalltalk.

Prerequisite: CISC 3120 [20.1] and 3130 [22].

Syllabus

Review of OOP Basics
- Encapsulaiton
- Inheritance
- Polymorphism

Advanced Concepts and Techniques
- Forms of inheritance: interface, implementation
- Programming by contract
- Subtyping vs subclassing
- Double-dispatching
- Reflection and runtime type information
- Multiple inheritance

Design Patterns
- Overview
- Creational
  - Abstract Factory, Builder, Factory, Lazy Initialization, Object Pool, Singleton
- Structural
  - Adaptor, Bridge, Composite, Flyweight
- Behavioral
  - Chain of Responsibility, Command, Iterator, Observer, Strategy
- MModel-View-Controller

Collection Hierarches
- Java Collections Framework
- C++ STL containers

Generic Programming in the Context of OOP
- Java Generics
- C++'s STL <algorithm> library

Smalltalk
- History and philosophy
- Basic programming: syntax, semantics, the environment; images
- Meta classes

Implementation of Object-Oriented Languages
Formal models of OOP systems
- Unified Modelling Language (UML)
- Linear temporal logic
- Computational tree logic
- Formal specification languages (Z, B, etc.)