CSC 123/252 Course Syllabus

Dr. Chuck C. Liang

Class Location and Times: Mondays and Wednesdays 4:31pm to 5:56pm in Adams 208

Office Hours: In-Person in Adams 116: MW 11am-12:30pm, Virtual (Zoom only) Th 5-6pm. All times subject to change.

Office Phone: (516 463) 5559

Email: chuck.c.liang@hofstra.edu

University's Zero-Tolerance Policy on Face Masks:

https://www.hofstra.edu/safe-start/index.html

Information Specific to CSC123/252, Fall 2021:

I. **IN-PERSON INSTRUCTION ONLY**. This class will attempt to return to a normal mode of teaching. Classes will be taught in-person, without being broadcast on zoom. Students will not be able to attend classes remotely. Office hours may still be attended on zoom, however.

II. **Attendance is mandatory.** Absences due to a medical nature must be documented. The instructor will determine on a case-by-case basis on how to provide make-up materials to students who are absent for legitimate reasons. Students who cannot be physically present during quizzes and exams (for legitimate reasons) may be subject to oral, one-on-one exams.

Course Description:

A study of the semantics, specification and behavior of programming languages. The course will focus on various programming language paradigms including functional, imperative, object-oriented and aspect-oriented programming. Programming assignments using example languages from these paradigms will be required. Emphasis will be placed on learning languages such as Scheme, Perl, C#, ML, C++ and Rust. Other topics covered include language syntax, control structures, objects and functions.

Prerequisite: CSC 16, 17; CSC 14 and 24 strongly recommended

Recommended Text: "Programming Languages: Concepts and Constructs, 2nd Edition" by Ravi Sethi.

Tentative List of Topics:

1. Introduction and review of programming concepts:

a. Runtime stack

b. Type System

c. Functional and class Abstraction

d. Recursion

2. Mathematical Foundations: The Lambda Calculus

3. Formal Models of Computation:

4. Lazy versus Eager Evaluation: The Church-Rosser Theorem

5. Dynamic versus Lexical Scoping

6. Side-effects and State

7. Object Orientation

8. Types

a. the typed lambda calculus

b. Safe versus unsafe type systems

c. runtime versus static typing

d. various forms of polymorphism

e. principal types

f. inheritance

g. templates

9. Typed functional programming

10. Comparison between approaches to modularity

11. Approaches to Memory safety (garbage collection, Rust)

12. Aspect Oriented Programming.

13. Advanced topics (if time allow)

Exams, Assignments and Grading

Assignments will be given regularly. There will be a midterm exam and a final. The final exam will be cumulative. Periodic quizzes, including one-on-one quizzes may also be given. The grade distribution will be roughly 60% exams and quizzes and 40% attendance, programming assignments and other homeworks. Grading will be curved but the exact curve can be modified by the instructor. Students are required to keep copies of all programming assignments throughout the semester. When working in a group, all group members must possess current versions of the assignment. Final Note: The contents of this syllabus may be modified depending on the progress of the course.