Programming Languages

CSCI 230

SEMESTER (fall 2015)

PROFESSOR/CLASS INFORMATION

Shieu-Hong Lin

(Course) Title: Programming Languages
Term: fall, 2015
Location: Lib 141
Office Phone: 562 903-4741
Office Hours: **Monday-Thursday 8:30-10:30am**
E-Mail: shieu-hong.lin@biola.edu
University Website: www.biola.edu

Course Code/#: CSCI 230
Class Days/Time: T Th 12:00-1:15pm
Credit Hours/Units: 3
Office Location: Grove 8
Meetings with Professor: Make Appt via Email
Admin Assistant: Jerrianne Smith, x4741
Dept. Website: http://csci.biola.edu

COURSE DESCRIPTION

Organization and structure of programming languages. Runtime behavior and requirements of programs. Introduction to programming language specifications and analysis. Study of various alternative languages such as C++, Java, and Python. Offered every year.

COURSE ALIGNMENT WITH PROGRAM LEARNING OUTCOMES

This lower-division course is a core course required of all Computer Science majors designed to be taken within the second year of the program. Successful completion of this course (see next section) will prepare students to demonstrate a developing proficiency toward the accomplishment of the PLOs: programming and system integration.

COURSE OBJECTIVES AND STUDENT LEARNING OUTCOMES

By the completion of this course including class participation, class assignments (referred to as "Tasks"), class readings and group interaction, the following objectives and learning outcomes will be assessed and demonstrated:

IDEA Objective #4: Developing specific skills, competencies, and points of view needed by professionals in the field most closely related to this course (Essential emphasis).

STUDENT LEARNING OUTCOMES (The learner will demonstrate that he or she has satisfactorily fulfilled IDEA Objective #4 by being able to):

1. Develop the foundational understanding of key concepts such as context-free grammars, regular expressions, lexical and syntax analysis, interpreters and compilers and their roles in the design and implementation of the modern programming languages.

2. Cultivate the skills of problem solving, object-oriented programming, and basic software engineering using C++ through a semester-long programming project implementing an interpreter that can run programs for the very simple Biola programming language.

3. Develop an in-depth understanding of key features of object-oriented programming languages such as C++ and Java in areas such as memory management, data types and type checking, inheritance and polymorphism, templates and generic programming to effectively utilize them in programming tasks.

4. Cultivate the understanding of (i) scripting languages including Python and Java Script and (ii) markup languages such as HTML, XML, and their application in web programming by contrasting and relating the features between C++ and the new languages.

REQUIRED TEXTS

- Stanley Lippman, etc. *C++ Primer (5th Ed)*, Addison Wesley, 2012.
- R. Sebesta, *Programming the World Wide Web*, 8th ed. Pearson, 2014.

LEARNING TASKS (Assignments) & ASSESSMENT (Grading)

Description and Weighting of Assignments:

Task 1: Weekly Reading and Progress Report (15 assignments)

Weighting: 10%
Possible Points: 4 points each.
Description: The student needs to incorporate information such as the amount of time he/she spent for the reading, attendance, and the overall progress in reading into the report.
Assessment: The student need to **(i)** finish the reading on time (1 point), **(ii)** attend the class this week (1 point), **(iii)** gain a good understanding of 80% or more of the contents **or** have spent at least three hours in the reading (1 point), and **(iv)** put down 1~2 paragraphs of reflection.

Task 2: Programming Assignments (4 modules)

Weighting: 40%
Possible Points: 6 points each.
Description: There will be 4 modules to complete in the programming assignments, which flesh out a framework of an interpreter program. They require the student to incrementally develop

programming skills based on the concepts of data structures and object-oriented programming learned in the class.

- Peer discussion is most encouraged, but any copy-and-paste code from other people's programs is absolutely prohibited and will lead to serious discipline actions.
- Peer discussion based on the code shown on the screen and paper could be very helpful for debugging purpose and explanation of ideas. But you should never pass around your code as electronic files to others except for the TA and the instructor.
- You should make sure that you are able to reconstruct your code from scratch without any outside help when you submit a programming assignment as your own work.

Assessment: The student needs to email the source code files (all related .cpp and .h files) together with a self-evaluation report to the TA. We'll grade each programming assignment in a 0-6 scale based on the following rubric.

1. Source code is completed but the code fails to compile successfully.
2. Source code can compile and do something required, but has serious bugs or miss a couple of key features.
3. Source code can compile and do most of the features required, but has many minor bugs or miss a key required feature.
4. Source code can compile and do all the features required, nearly fully functional, only a couple of minor bugs.
5. Source code can compile and do all the features required, fully functional, no bugs.
6. In addition to the points received above, get one more point if the source code is well indented and commented to make it visually very readable.

Task 3: **Language Study Assignments**

Due Date: Distributed throughout the semester.
Weighting: 30%
Description: We'll assign a sequence of study assignments including (i) presentations regarding the textbook on web programming and (ii) investigation of key features in modern programming languages such as memory management, built-in data types, static typing or dynamic typing (type checking), control flow, subroutine call and parameter passing, and garbage collection, templates and generic programming, object-oriented programming (inheritance and sub-type polymorphism), functional programming, and so forth.
Assessment: The submitted work done for the presentation and the language study assignments will be examined to determine whether a satisfactory level of study has been done based on the breadth, depth, and clarity of the submitted work.

Task 4: Exams (Quizzes, midterm and final exams)

Due Date: Once each month for the quizzes, Exams in the midterm and final exam week
Weighting: 20%
Possible Points: Up to 50 points each.
Description: The exams have both the written component, which mainly tests the conceptual understanding of data structures, and the programming component, which tests skills in object-oriented programming.

Assessment: The written component will be graded based on the answers provided while the programming component will be graded based on the same rubric provided for the weekly programming assignments.

CLASS INFORMATION

1. Class Attendance and Attendance Policy:

Attendance: You are expected to attend the class regularly since we will examine programming projects and various concepts of programming languages in the class. Missing the class may seriously hamper your understanding of many key concepts and programming skills critically needed in your programming assignments. Class attendance is counted toward points for the weekly progress report.

2. Turning in Assignments:

Assignments are expected to be electronically submitted under the Canvas system. Due dates are all on Tuesdays. The submission link under Canvas may remain open for 2 more days after the due date.

3. Late Policy:

1 point will be deducted for late submission within 2 days of the due date while the submission link is still open. **You will receive no points after the submission on canvas is closed** unless it is something like a serious health issue with statements from the doctor as proof.

4. Computation of Final Grade:

Weekly Progress Report	10 %
Weekly Programming Assignments	40%
Language Study Assignments	30%
Exams	20 %
Total	**100%**

5. Final grades will be awarded on the following point system:

A	93%
A-	90%
B+	87%
B	84%
B-	80%
C+	77%
C	74%

C-	70%
D+	67%
D	64%
D-	60% to pass class

GENERAL INFORMATION

1. The GPA System used by the University Registrar's Office is:

A = 4.0	B = 3.0	C = 2.0	D = 1.0
A- = 3.66	B- = 2.66	C- = 1.66	D- = 0.66
B+ = 3.33	C+ = 2.33	D+ = 1.33	F = 0.0

2. Method of Instruction:

The following methods of instruction will be included in this course:

1. Lecture
2. Written Reports
3. Programming Assignments
4. Use of the Internet
5. Reading

3. Posting of Grades:

Grades for individual assignments will be posted under Biola's Blackboard system. To access the records online, log on to http://canvas.biola.edu/ to make sure the records are accurate.

Tentative Schedule

Weeks 1~2	Language translation: Syntax and Semantics
Weeks 3~4	Inheritance and polymorphism
Week 5	Names, scoping, and binding
Week 6	Memory management
Week 7	Type checking
Week 8	Function calls
Week 9	Review and midterm

Weeks 10~12	Templates and generic programming
Weeks 13~14	Function programming and logic programming
Week 15	Review