COMPILERS

School of Health Sciences, Technology and Trades

Course Number: CST8152	**Contribution to Program:** Core	**Educator(s):** Ian Allen
Applicable Program(s): Computer Science Technology Computer Engineering Technology	**AAL:** 05 05	**Approval Date:** 1997 Summer Semester **Approved By:** David Fisher, Chairperson Computer Studies/Math Dep't.
Course Hours: Delivered: 64 Normative: 64	**Prerequisites:** CST8130, CST8134 **Corequisites:** None	**Approved for Academic Year:** 1997 - 1998

COURSE DESCRIPTION

This is a medium level applied course. The theory of compilers provides the foundation to develop a rather complex programming application. The C programming language is used as a tool to create a series of increasingly powerful text processors / interpreters. No new C constructs or C language elements are taught, though some time is spent learning how to structure and test a large programming project. Students will use their knowledge of data structures such as stacks, tables, and dynamic memory allocation to implement their projects. Each assignment builds on the previous, so the timely and thoughtful completion of each and every assignment is essential.

RELATIONSHIP TO PROGRAM LEARNING OUTCOMES	
This is a vocational course that supports the following vocational program standards:	**This course contributes to your program by helping you to achieve the following provincial generic skills standards:**

COURSE CURRICULUM

I. Course Learning Requirements/Embedded Knowledge and Skills

Course Learning Requirements	Knowledge and Skills
When you have earned credit for this course you will have demonstrated an ability to:	
Understand the process of compiling and the parts of a compiler.	Program debugging. Coordinating the compilation and linking of multiple files containing self-contained data structures and functions. Application of the C programming language in a complex program. Writing understandable large programs. Building code for modification and re-use.
Understand the process of and tools for lexical analysis, parsing, error handling, and interpretation.	
Describe the different aspects of code generation.	
Understand basic grammars and regular expressions.	
Describe the nature of a compiled and assembled program executable.	

II. Learning Resources

The course consists of three hours of lectures and one hour of laboratory per week.

Assignments derive from the material covered in the lectures. Due to their complexity, most assignments require work outside assigned lab hours. Students will have access to the lab after hours for this purpose. Lab assignments may be developed using any C environment, but they must compile and run in the Algonquin laboratory environment using Borland or Turbo C. Assignments written in languages other than ANSI C are not accepted; assignments that work at home or elsewhere, but do not work in the labs, are not accepted.

Textbook:

Aho, et al; *Compilers - Principles, Techniques & Tools*; Addison-Wesley

References:

Class Notes: **`http://www.algonquinc.on.ca/cst/8152/`**

III. Teaching/Learning Methods

During this course you are likely to experience:

Lecture sessions will present the theoretical material of the course, aided by the use of support media such as overhead transparencies, computer demonstrations and/or additional lecture notes. Students are expected to read assigned material and be prepared to answer oral or written questions in following lectures and laboratories. Laboratory sessions will require students to apply the readings and lecture material to a series of assignments. The

assignments build upon each other and become increasingly complex as the course progresses. Students are encouraged to ask questions during lectures and laboratories.

IV. Learning Activities and Assessment

Samples of learning activities include:

Lectures and readings will touch on the following topics. Students are expected to observe carefully and to ask for clarifications or further examples:

1. The compilation process (2 weeks)

   Relationship between languages and machines, aspects of the compilation process, assembly and linking, parts of a compiler. Introduction to interpreters.

2. Lexical analysis and language definition (6 weeks)

   Recognition of symbols, output from the lexical analyzer, state transition diagrams, finite state automata, regular expressions. Syntax and semantics, grammars, formal definition of programming languages, parse trees, the parsing problem.

3. Language tools, syntax analysis, parsing and semantics (5 weeks)

   Context-free grammars, recursive descent parsing, bottom-up parsing, error recovery, symbol tables, type checking.

4. Interpreters and compilers; code generation (2 weeks)

   Interpreters vs. compilers. Production of quadruples, intermediate languages, target language.

5. Other aspects of compiling (as time permits)

   The run-time stack, object code optimization, assemblers, parser generators.

Laboratories will provide opportunities for hands-on use of the computer to write, test and debug computer programs, with the professor in attendance and on call for assistance. Laboratories will also be used for individual demonstration and evaluation of completed work.

Students are expected to work on their own and to ask for assistance from the professor when necessary. Students may be required to show completed pre-lab portions of assignments before being admitted to the laboratory.

V. Evaluation/Earning Credit

The following will provide evidence of your learning achievement:

Assessment of student learning will be done by means of class quizzes, mid term tests, lab assignments,

lab demonstrations, and a written final examination.

Lab attendance is mandatory, and absence from three or more laboratory sessions without a medical certificate or the consent of the instructor will result in a final grade of **F**.

Assignments will be penalized for lateness: late less than 1 week, -20%; late 1 week or more, -100%. All laboratory assignments *must be successfully completed* to receive credit for the course, even if the assignments are late.

Plagiarism (work submitted by the student that is substantially the work of other persons) will not be tolerated. Students who knowingly allow their work to be copied *will receive the same sanctions* as the plagiarizer. The first occurrence of plagiarism will result in a mark of zero for the assignment and all the students involved will be required to redo it. A second occurrence will result in an overall course grade of **F** for all concerned. ***Share your ideas, not your source code.***

The factors in the final grade are:

1.	Lab assignments	20%
2.	Mid term tests	40%
3.	Final exam	40%

The final grade will be calculated as the total of the above factors. In addition, the student must achieve at least 40 out of 80 in the total of the mid terms and final exam to receive a passing grade.

VI. Prior Learning Assessment

Evidence of learning achievement for PLA candidates will include:

A portfolio of related work completed by the student, completion of a challenge test with a breadth of coverage and level of difficulty equivalent to the final examination in the course, with a grade of at least C and at the assessor's discretion, successful completion of a special assignment relating to the course content.

RELATED INFORMATION

If you are a student with a disability please identify your needs to the professor and/or the Centre for Students with Disabilities (CSD) so that support services can be arranged for you. You can do this by making an appointment at the CSD, Room C142, Ext. 7683 or arranging a personal interview with the professor to discuss your needs.

Students, it is your responsibility to retain course outlines for possible future use to support applications for transfer of credit to other educational institutions.