Syllabus: CmpSci *4280*

Program Translation Techniques

Department of Mathematics and Computer Science

Semester

SP/SS/FS 20xx

Instructor Details

Name: *Put name*
Office Hours: *Put hours, 1.5 hrs/wk per course for FT faculty, 3 hours a week for adjunct*
Office Location: *Put location, can be in office, in class, online, etc.*

Submission and Communication

Your specific policies and procedures regarding submissions, late submissions, communication means, etc.

Scoring

List any additions/changes you want to make to the course details below, such as using quizzes, attendance requirements, etc. Keep in mind you cannot change the course details, you can only work within what it says. So if you want to add quizzes and they are not listed, scoring for quizzes has to be added to homework, tests, or another part. If the course detail gives ranges for grading, you have to provide specific values within these ranges.

Incremental Grading

Provide information if used.

Schedule

If the course detail does not state detailed timing or sections/topics, you may put them here. Keep in mind you cannot remove topics and if the course is coordinated you may have to follow topic allocations.

Course Details

General Policies

We follow the university policies regarding excused EX and EX-F drops.
Students are given and are expected to sustain positive learning environment in class. This means positive conduct in class, no late walk-ins or early walk outs without a good explanation or a prior arrangement, and if on-line access is available in class - not using it for anything not class related. Students not meeting these standards may be asked to leave the classroom.
Sample tests will be provided.
All in and out of class work for grade should be done independently. Homework can be discussed with others, but the final work (code, answer, etc.) should be independent. Programs may be discussed up to design, but no code is allowed to be shared except for what is presented in class. Help can always be sought and received. However, help

to assignments should be generic on the subject matter or very narrowly focused on specific problem not being the central point in the assignment.

Course Description

Prerequisites: CMPSCI2700, CMPSCI 2750, CMP SCI 3130, and CMP SCI 4250, or graduate standing. This course focuses on methods, techniques, and mechanisms used to bridge the abstraction from high level programming to machine level execution, and it also requires an individual semester long project.

Text and Other Materials

No textbook is required, materials will be provided. Suggested books:

- D. Watt. "Programming Language Processors". Prentice Hall, 1993 (reserve and ESH 316)
- C. Fisher and R. LeBlanc, Jr. "Crafting a Compiler with C" (reserve and ESH 316)
- R. Sabesta. “Programming Languages” (current cs4250 text)

Course Schedule

No fixed schedule, but the course schedule will be organized around topics relevant to program translation and building a working compiler as a semester project. Overall, the following topics will be covered, with notes distributed to everyone.

• *Programming standards: source and architecture*	*1 week*
• *Introduction to languages*	*1 week*
• *Translation models*	*1.5 week*
• *Lexical analysis*	*1.5 week*
• *Introduction to context free grammars*	*1 week*
• *Top down parsing*	*2 weeks*
• *Semantics processing and code generation*	*2 weeks*
• *Testing, project discussions (various times)*	*5 weeks*

Course Objectives and Learning Outcome

The course will have a semester long project to build a working compiler, translating from some subset of a modern programming language into a simple assembly or machine language with available virtual machine for execution. One of the main objectives is to build a working product from independently tested modules.

Topics and objectives not related to translation

- Programming standards
- Program architecture and modularization, internal vs. external linkage, header files
- Building a larger project from components
- Problem decomposition
- Incremental development, testing, integration

Topics and objectives related to languages and translation

- Generation vs. interpretation
- BNF notation, standard and extended

- Ambiguity, precedence and associativity
- Top-down vs. bottom-up compilers
- Modular and reconfigurable compilers, front end vs. back end
- Interpretive compilers (Java or Pascal)
- Cross vs. host translation
- Bootstrapping
- Improving compiler and target properties
- Static semantics vs. execution semantics
- Global vs. local storage allocation
- Process space elements
- Tombstone (T, mushroom) notation
- The need for program translation, constraints, tradeoffs
- Chomsky hierarchy of languages
- Algorithms and properties for regular and context free languages

Upon completion, a student should be able to

- Develop software using proper programming and architectural standards
- Develop software incrementally
- Assist with translation needs

Course Grading

We will use the standard 10% grading scale: 90% and above gives A, 80% and above B, 70% and above C, 60% and above D, else F. Graduate students may be required to do additional work to be incorporated into the grade scale.

Tests	40-50%
Semester project, other projects	40-50%
Homework, quizzes, etc.	10-20%

University Policies and Information

http://www.umsl.edu/~webdev/mathematics/files/pdfs/cs_umsl_syllabus_university.pdf