Syllabus: CmpSci *4280*

Program Translation Techniques

Department of Mathematics and Computer Science

Semester

SP/*SS/FS 2019*

Instructor Details

Name: *Cezary Z Janikow*
Office Hours: *Monday 1:15-2:30pm, Wednesday 4:00pm-5:15pm*
Office Location: *ESH 308*

Scoring and Grading

Grading:

- *Tests 40%*
- *Projects 45%*
- *Homework, attendance, quizzes 15%.*

Incremental grades (+/-) can be used.
No delayed grade will be given unless really special circumstances are proven.

Submission, Participation and Communication

Communicate through Canvas or from official UMSL account. Other emails may be missed or not answered.
Submissions and grades postings will also be through Canvas. However, programming projects may also require separate script-based submission.
Students are expected to be in class. Absences beyond two may result in penalty.
No make-up will be given for missed tests. Special circumstances will be discussed individually.
All programming is expected to be done on time (always midnight), with a 3-day grace period with 10, 20, and 30% penalty, respectively. Selected students may be required to make in-person project presentation, which at such time would be required for receiving score.
Non-programming assignments will not be accepted after the deadline (always beginning of class). Solutions will be discussed in class, and these discussions replace most of the feedback for the actual submission.

Schedule

We will follow the schedule under course details below.
In the event of missed classroom meetings due to unexpected closings announced by the university or unexpected incapacity of the instructor, instructions for study at home or online session will be posted and expected to be carried out or participated in.

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 complex 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