Introduction

CS 5300

Kenneth Sundberg

Grading Weights

- Exams - 30%
- Compiler Project - 70%

ADA

- Course is ADA compliant
- If necessary get me paperwork from the DRC

Cheating

- Don't do it
- Group Work

Compilers

- gcc 7.3
- flex 2.6
- bison 3.0.4
- cmake 3.9
- Mars 4.5

Newer versions should be fine, but avoid fancy new features.

Compiler Project Source Language

- We are building a simple compiler
- Language is PASCAL-like
- Only integral types and string constants
- MIPS is the target archetecture
- Beware - sometimes the dragon wins.

GitHub

- https://github.com/ksundberg/coursematerials/cs5300
- Documentation
  - Also available on Canvas as a .pdf
  - Further sections and clarifications may be added as course progresses
- Test Files
  - Correct processing of these files is the goal of compiler project
- Feel free to make pull requests, especially for more test files

Language Processors

- Compilers convert from source languages to target languages
- An important side job is error detection

Processing Chain

- Preprocessor
- Compiler
- Assembler
- Linker
- Loader

Symbol Table

- All parts access the Symbol Table
- Stores type and location information

Front End

- Responsible for language dependent portions
- Lexical, syntax, and semantic analysis
- Intermediate code generation

Optimizer

- Most work done here
- Intermediate code to better intermediate code
- Independent of both source and target language

Back End

- Responsible for target dependent portions
- Code generator and machine dependent optimization

Compiler Collection

- Mix and Match front ends and back ends
- New languages only require a front end
- New machines only require a back end

Scanning

- Lexical Analysis
- Regular Language
- Transform character stream into tokens

Parsing

- Syntax Analysis
- Context Free Grammar
- Transform token stream to syntax tree

Code Generation

- May include optimization
- Transform syntax tree to executable code

Generations

- First generation - machine code
- Second generation - assembly code
- Third generation - general purpose languages
- Fourth generation - special purpose languages

Types of Languages

- Imperative vs. Declarative
- High Level vs. Low Level