# CSE 440 Compiler Construction

Brett Hansen

## Contents

# 1 Week of August 14th, 2016

## 1.1 Compiler Components

1. Syntax
   (a) Lexical Analysis
   (b) Parsing
2. Semantics
   (a) Type Checking *(Project 1)*
   (b) Code Generation
      i. Intermediate Representation *(Project 2)*
      ii. Analysis/Optimization *(Project 2)*
      iii. Low Level Code Generation & Optimization *(Project 3 & 4)*

## 1.2 Language Components

1. Character Set
2. Tokens: sequence of characters
   (a) identifiers
   (b) keywords
   (c) separators
   (d) operators
   (e) built-in types
   (f) modifiers

   Tokens are defined using regular expressions.

## 1.3 Regular Expressions

| **Regular Expression** | **String Representation** |
|---|---:|
| $\emptyset$ | *empty set* |
| $\epsilon$ | *empty string* |
| a | {a} |
| $R_1 \vert R_2$ | $L(R_1) \cup L(R_2)$ |
| $R_1 \cdot R_2$ | $L(R_1) \cdot L(R_2)$ |
| $R^*$ | $\{\epsilon\} \cup L(R) \cup L(R) \cdot L(R) \cup ...$ |
| $(R)$ | *grouping* |

$L(R)$ The set of strings that R represents, also known as the *language of R.*

Precedence $* \quad \rightarrow \quad \cdot \quad \rightarrow \quad |$

**Regular Expression Example:**

ID *letter followed by 0 or more letters or digits*

letter $a \mid b \mid \cdots \mid z \mid A \mid B \mid \cdots \mid Z$

digit $0 \mid 1 \mid \cdots \mid 9$

ID $letter \cdot ( \; letter \mid digit \; )^*$

From regular expressions ⟶ non-deterministic finite state automata

↪ deterministic finite state automata

↪ program