CMSC430: Introduction to Compilers

Michael Li

Contents

1 Introduction to Compilers

`Compiler: Source Program -> Target Program`

Compiler takes in **Source Program** and outputs a **Target Program**

- **Concrete Syntax**: Defines the "shape of the source program code"
- **Parsers**: Takes concrete syntax and produces an **Abstract Syntax Tree**
- **Grammar**: Defines what things you can express with concrete syntax
- **Note**: This does not necessarily have to be from higher-level program to a lower-level program (e.g. compilers exist that take C to javascript)

Upshot: Compilers breakdown interpretation into 2 phases, each with access to different resources

1. **Compile Time**: Translating original source language to another target language (has access to source code)
2. **Run Time**: Running the program (no access to source code)

1.1 Importance of Source-Target Relationship

When we write a program, what do you expect the implementation program language to do (or not do)?

CMSC430 studies this relationship, specifically looking into how we bridge the gap between high-level languages and low-level languages

Semantics: Defines program meaning

- Do all features in one language need to be present in the target language? – No they shouldn't