ĐẠI HỌC ĐÀ NẴNG
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG VIỆT - HÀN
Vietnam - Korea University of Information and Communication Technology

Lecture 1: Introduction

What is this course about?

In computing, a compiler is a computer program that translates computer code written in one programming language (the source language) into another language (the target language) *<wikipedia.org>*

What is this course about?

The name "compiler" is primarily used for programs that translate source code from a high-level programming language to a lower level language (e.g. assembly language, object code, or machine code) to create an executable program.*<wikipedia.org>*

What is this course about?

- How are programs written in a high-level language translated into machine code?
- How can we ensure a program is somewhat meaningful?
- How is the program's memory managed?
- How programming languages are designed?
- Why there are so many different kinds of programming languages?

Why study compilers?

Textbooks and Resources

1. **Compliers Principles technique and Tools (Second Edition)**

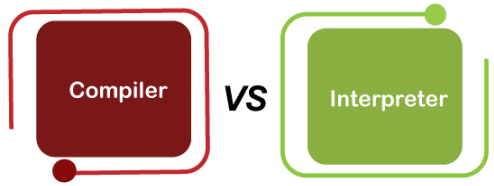
   Alfred V.Aho, Ravi Sethi, Jeffrey D.Ullman

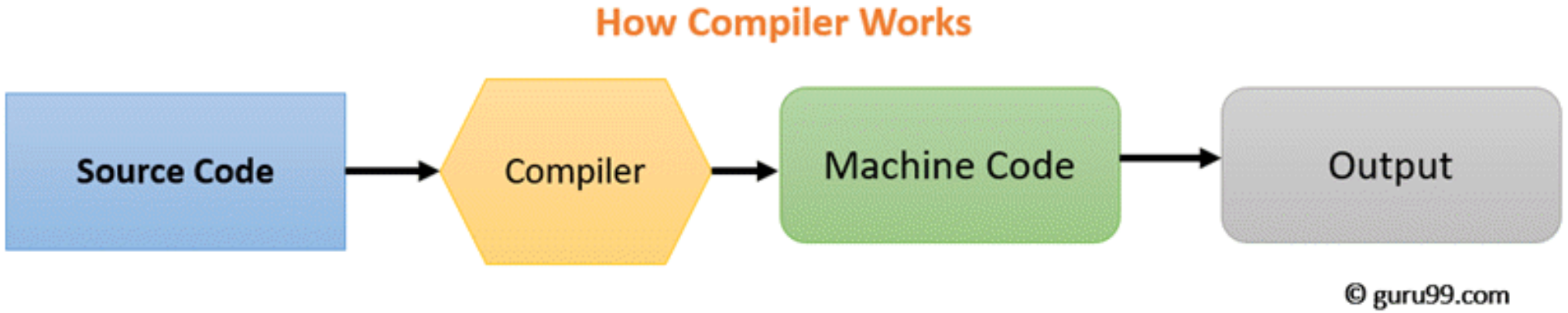
2. **Trình biên dịch**

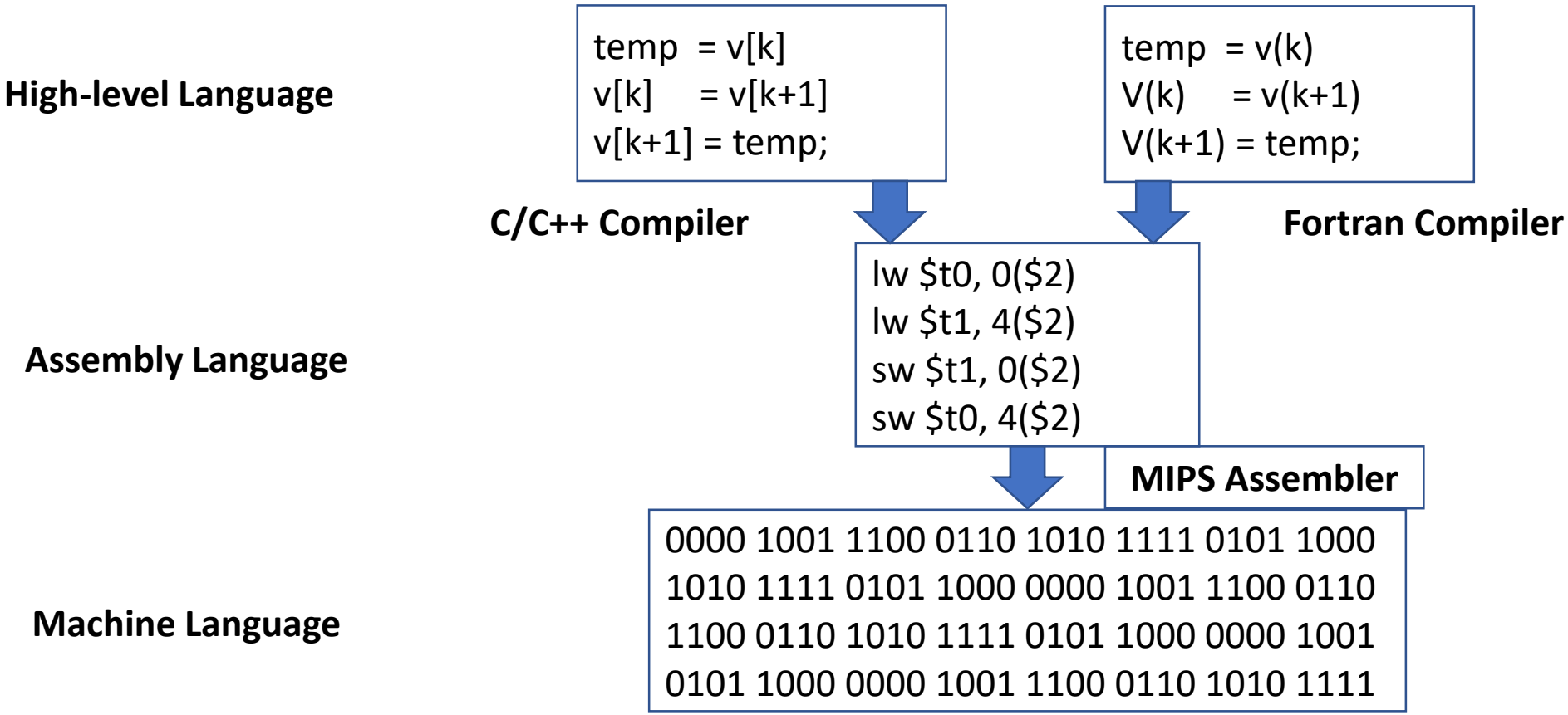
   Phan Thị Tươi

3. Introduction to Programming Languages/Grammars
4. Write your own compiler
5. The lox language

Compiler vs Interpreter






High-level Language
temp = v[k]
v[k] = v[k+1]
v[k+1] = temp;
temp = v(k)
V(k) = v(k+1)
V(k+1) = temp;
C/C++ Compiler
Fortran Compiler
Assembly Language
lw $t0, 0($2)
lw $t1, 4($2)
sw $t1, 0($2)
sw $t0, 4($2)
MIPS Assembler
Machine Language
0000 1001 1100 0110 1010 1111 0101 1000
1010 1111 0101 1000 0000 1001 1100 0110
1100 0110 1010 1111 0101 1000 0000 1001
0101 1000 0000 1001 1100 0110 1010 1111

Structure of a compiler

- Lexical Analysis
- Parsing
- Semantic Analysis
- Optimization
- Code Generation

Lexical Analysis

- First steps: recognize words
  - Smallest unit above letters

This is a sentence

ist his asen tence

Lexical Analysis

- Lexical analysis devides program text into “words” or “tokens”

```
if num1 == num2 then result = 1; else result = 2;
```

Keywords: if, then, else

Variable names (identifier): num1, num2, result

Constants: 1, 2

Operators: double equals, assignment

Other tokens: semi colons, the punctuation, the separators (blank)

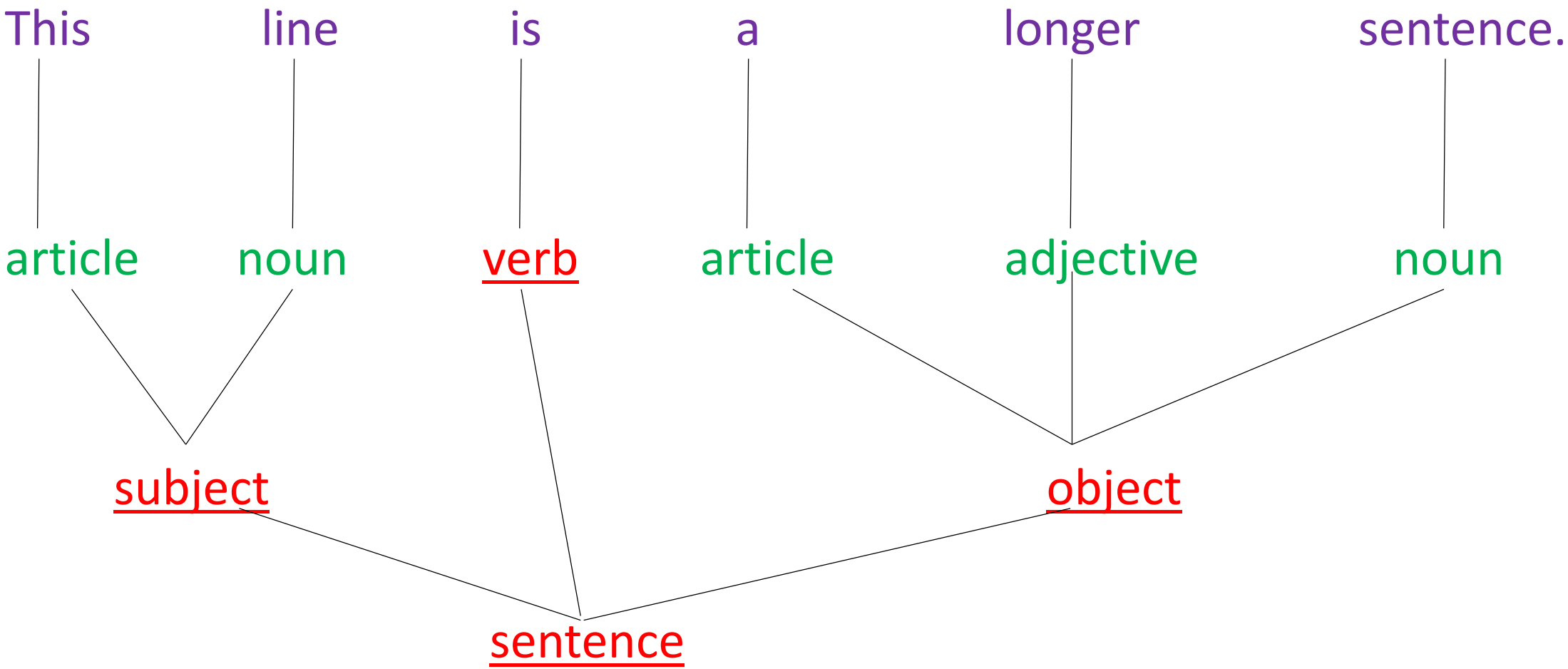
⇒How do we know that is double equals, not two individual equals signs?

⇒How do we know that we want this? A double equal, not two single euqals?

Parsing

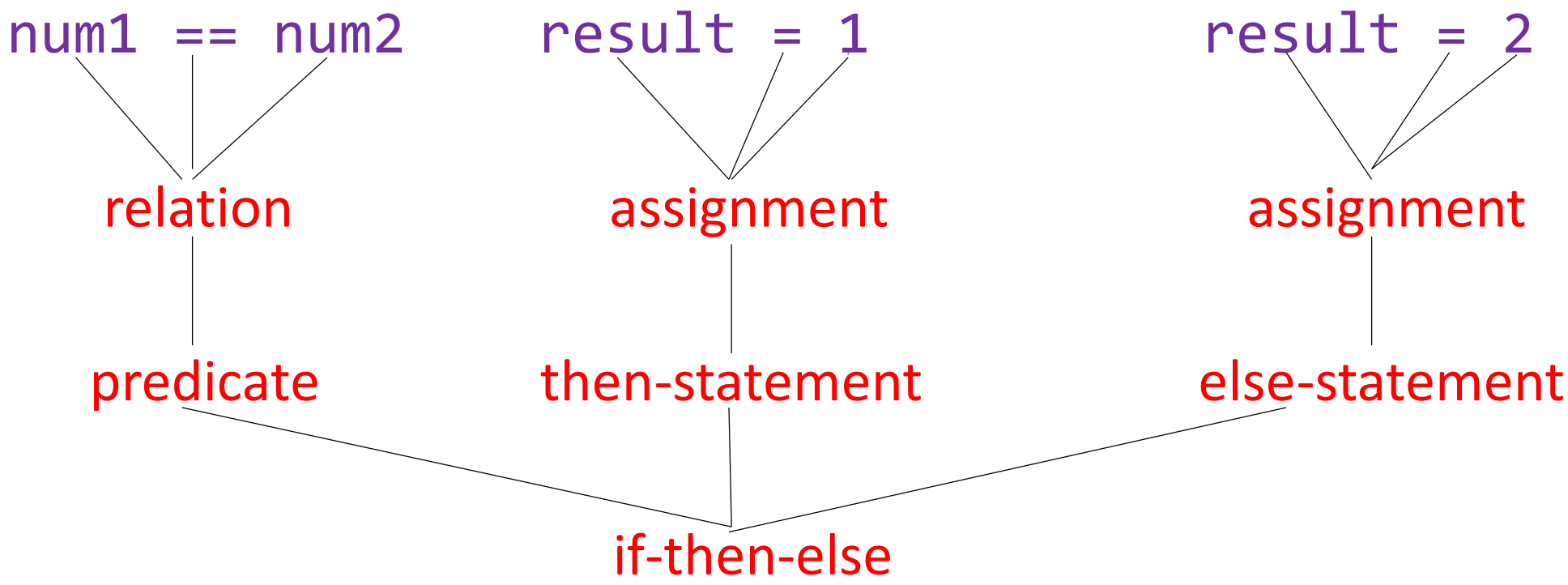
- Once words are understood, the next step is to understand sentence structure.
- Parsing = Diagramming Sentences
  - The diagram is a tree

Parsing


This
line
is
a
longer
sentence.
article
noun
verb
article
adjective
noun
subject
object
sentence

Parsing

```
if num1 == num2 then result = 1; else result = 2;
```



Semantic Analysis

- Once sentence structure is understood, we can try to understand “meaning”.

- Compilers perform limited semantic analysis to catch inconsistencies.

Semantic Analysis

Tom said Jerry left his key at home.

Tom said Tom left his key at home?

Semantic Analysis

- Programming languages define strict rules to avoid such ambiguities

```
{
        int Tom= 20;
        {
                int Tom= 50;
                cout << Tom;
        }
}
```

Semantic Analysis

- Compilers perform many semantic checks besides variable bindings
- Example: Tom left her homework at home.
- A ‘type mismatch’ between Tom and her, we know they are different people.

Optimization

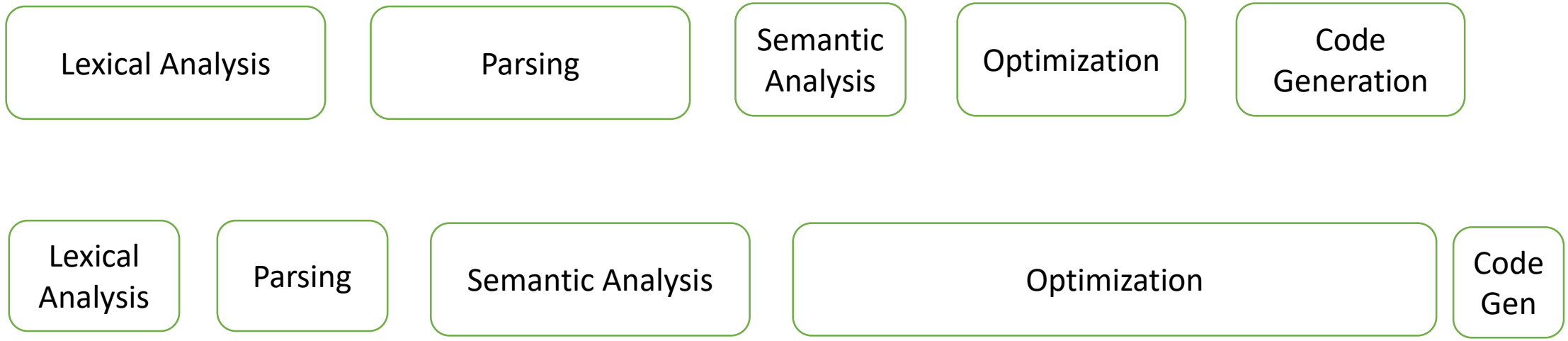
- Automatically modify programs so that they
  - Run faster
  - Use less memory
  - Reduce the amount of power
  - Reduce the number of network messages
  - Reduce the number of database accesses.

Optimization

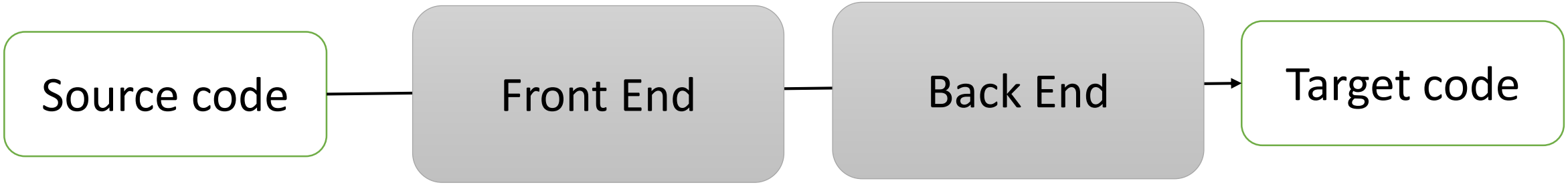
A = B * 0 is the same as A = 0

Code Generation

- Produces assembly code (usually)
- A translation into another language (analogous to human translation)



- Compiler translates from one language to another



- Front End: Analysis
  - Pulls apart program, understand structure and meaning
- Back End: Synthesis
  - Puts it back together in a different way

- Why are there so many programming languages?
- Why are there new programming languages?
- What is a good programming language?

Why are there so many programming languages?

Application domains have distinctive

Scientific computing

- need good floating point support,
- need good for arrays and operations on arrays
- need parallelism

Business applications

- need persistence
- report generation
- data analysis

Systems programming

- control of resources
- real time constraints

Why are there new programming languages?

What is a good programming language?

There is no universally accepted metric for language design.