# Computer Fundamental

Lecture 4

By
Nakib Aman Turzo
Lecturer, Department of CSE
Varendra University

# Introduction to Computer Programming

# Outline of Topics

- Computer Languages
- Compilation vs. Interpretation etc.

# Programs

- Programs are written in programming languages
  - PL = programming language
- A PL is
  - A set of rules and symbols used to construct a computer program
  - A language used to interact with the computer

# Computer Languages

- Machine Language
  - Uses binary code
  - Machine-dependent
- Assembly Language
  - Uses mnemonics
  - Machine-dependent
- High-Level Language (HLL)
  - Uses English-like language
  - Machine independent
  - Examples: Pascal, C, C++, Java, Fortran, Excel Programming. . .

# Machine Language

- The representation of a computer program which is actually read and understood by the computer.
- Instructions:
  - Machine instructions are in binary code

**Example:**

| Operation | Address |
|---|---|
| 0010 | 0000 0000 0100 |
| 0100 | 0000 0000 0101 |
| 0011 | 0000 0000 0110 |

# Assembly Language

- A symbolic representation of the machine language of a specific processor.
- Mnemonic representation of the instructions and data

- **Example:**

```
Load    Price
Add     Tax
Store   Cost
```

# High-level language

- A programming language which use statements consisting of English-like keywords such as "FOR", "PRINT" or “IF“, ... etc.
- Much easier to program than in assembly language.
- Example:

Cost := Price + Tax

# Compilers & Programs

- **Translation Software**
  - A program that converts another program from some source language (or high-level programming language / HLL) to machine language (object code).
  - Some translators output assembly language which is then converted to machine language by a separate assembler.

# Compilers & Programs

- **Source program**
  - The form in which a computer program, written in some formal programming language, is written by the programmer.

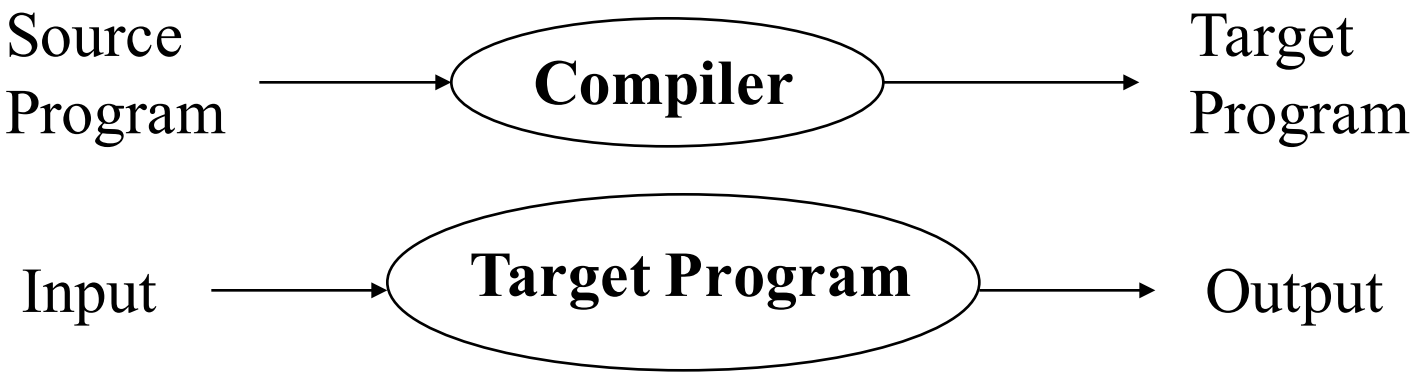
# Compilers & Programs

- **Object program**
  - Output from the translator software

# Types of Translator Software

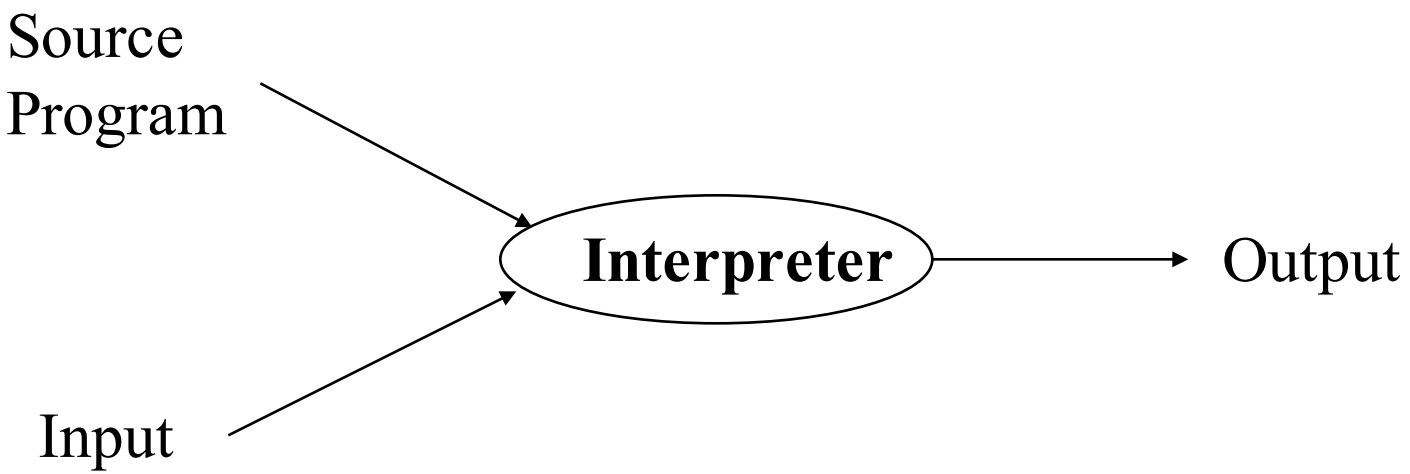
- Compiler
- Interpreter

# Compilation




- Compiler translates source into target (a machine language program)
- Compiler goes away at execution time

# Interpretation




- The interpreter stays around during execution
- It reads and executes statements one at a time

# Compilation vs. Interpretation

- Compilation:
  - Syntax errors caught before running the program
  - Better performance
  - Decisions made once, at compile time
- Interpretation:
  - Better diagnostics (error messages)
  - More flexibility
  - Supports creation/modification of program code on the fly (e.g. Lisp, Prolog)