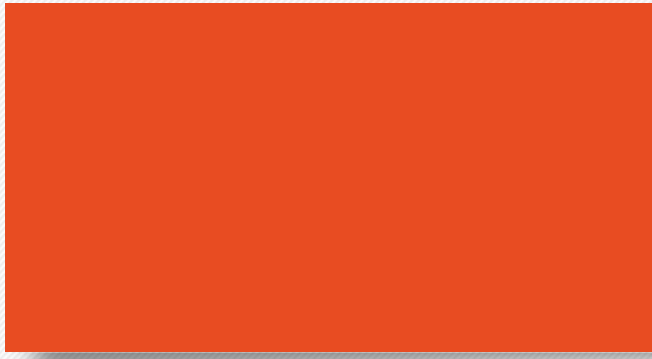
# Compilers - Introduction



# About the Course

- Subject Name: Compiler design
- Credits :  3
- Sub code : CSE3222
- Text book:
  - Alfred V. Aho, Monica S. Lam, Ravi Sethi, Jeffrey D. Ullman, “Compilers Principles, Techniques and Tools”, Pearson Education, 2nd edition. 2010.

# Course Outcomes (COs)

1. Ability to familiarize on different phases of a compiler and recognize steps involved in lexical analyzer generators.
2. Ability to understand the top down and bottom up parsing techniques.
3. Ability to identify ambiguous grammars and analyse syntax directed translation techniques.
4. Ability to translate expressions into three address code and elaborate code generation phase.
5. Ability to understand the automated tools like FLEX and YACC.

# Introduction

- What is Compiler?
- Why study compiler?
- How is interpreter different from compiler?

# Difference b/w Compiler and Intrepreter

| Basis for Comparision | Compiler | Intrepreter |
|---|---|---|
| Input | It takes an entire program at a time | It takes a single line of code or instruction at a time |
| Output | It generates intermediate object code | It does not produce any intermediate object code |
| Working Mechanism | The compilation is done before execution | Compilation and execution take place simultaneously |
| | | |
| Memory | Memory requriement is more due to the creation of object code | It requires less memory as it does not create intermediate object code |
| Errors | Display all errors after compilation all at the same time | Display errors of each line one by one |
| Error Detection | Difficult | Easier |
| Programming Languages | C, C++, C#, Scala, Typescript uses compiler | PHP, Perl, Python, Ruby uses an interpreter |

# Hybrid C

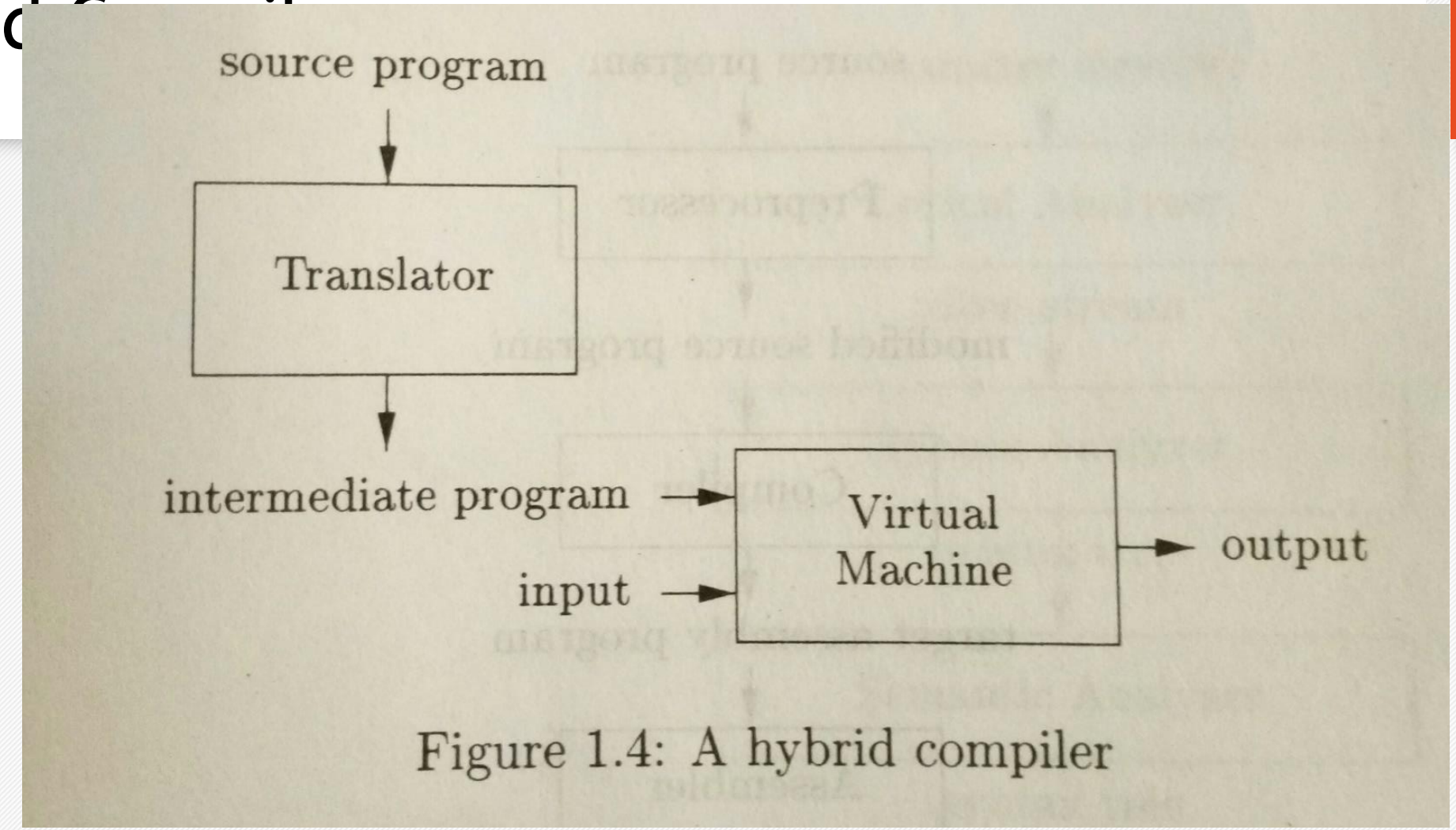



Figure 1.4: A hybrid compiler

# Language Processing System

Before knowing about the concept of compilers, let us first understand the processing system which work with compilers.

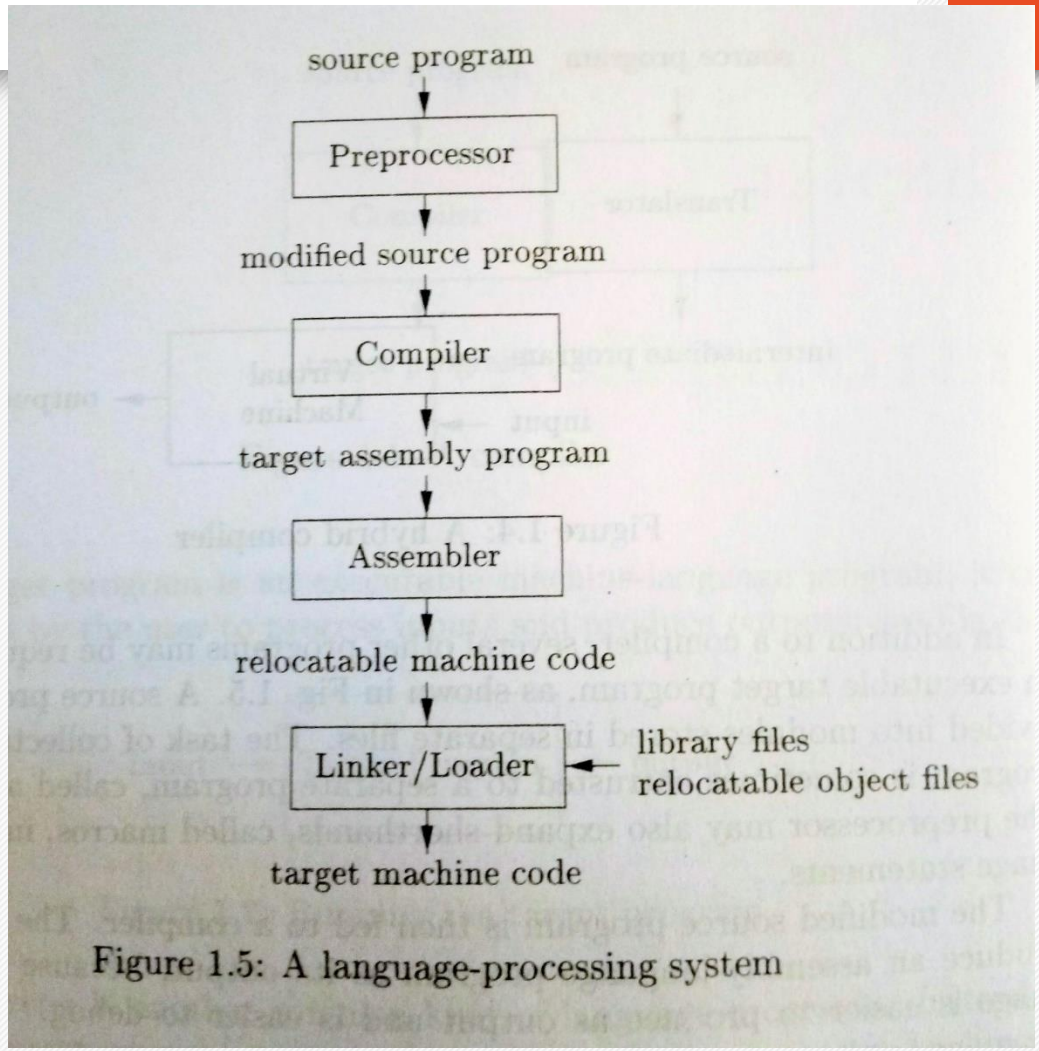
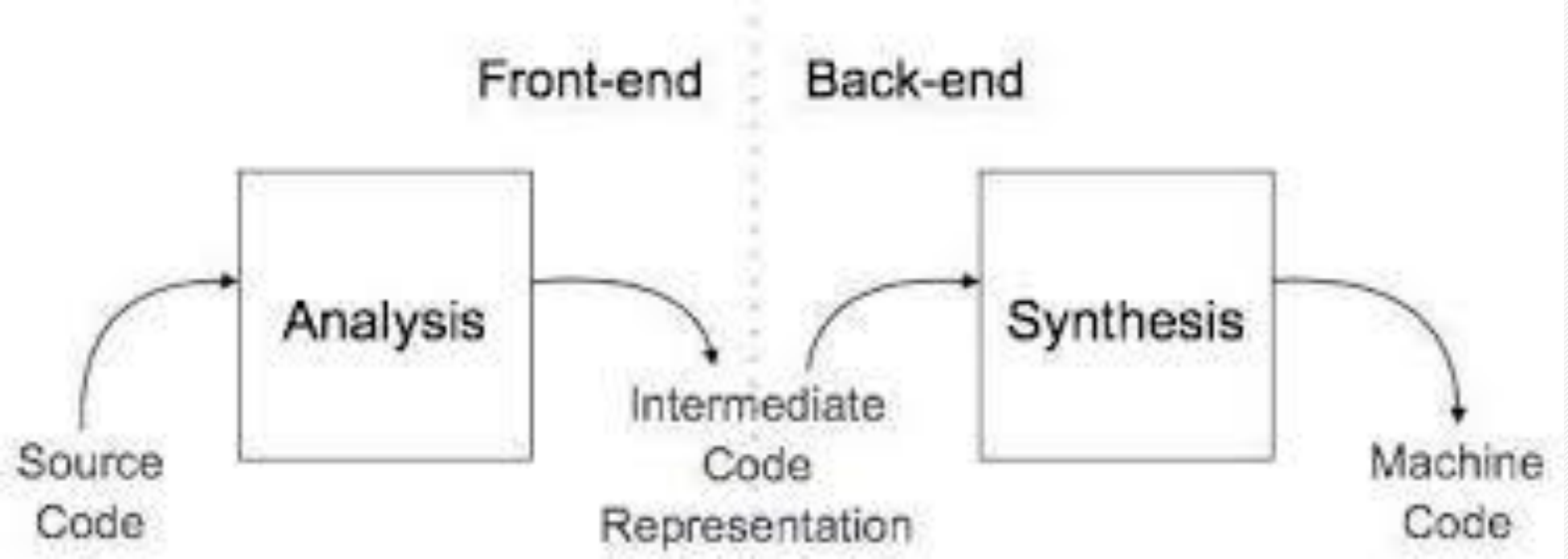
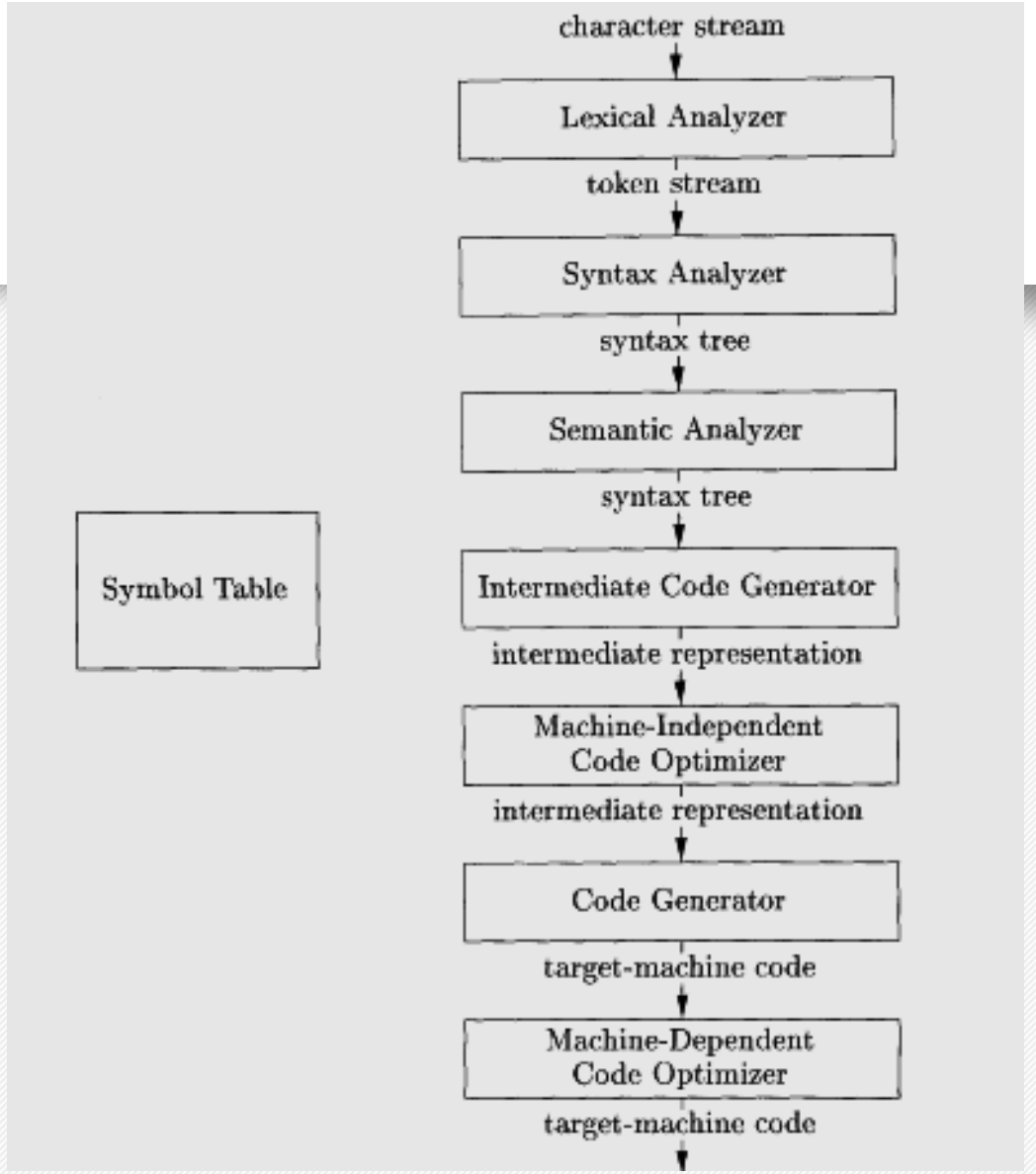


Figure 1.5: A language-processing system

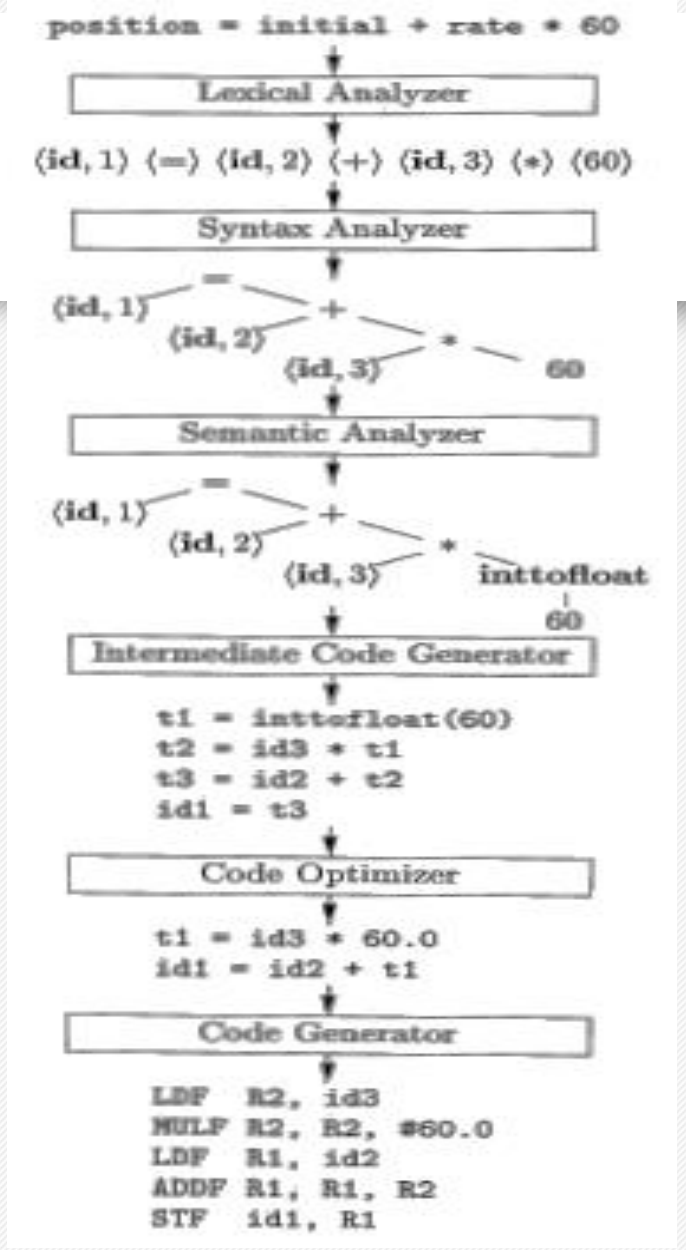
# Structure of the compiler



# Phases of compiler



# Example


position = initial + rate * 60
Lexical Analyzer
⟨id, 1⟩ ⟨=⟩ ⟨id, 2⟩ ⟨+⟩ ⟨id, 3⟩ ⟨*⟩ ⟨60⟩
Syntax Analyzer
=
⟨id, 1⟩
+
⟨id, 2⟩
*
⟨id, 3⟩
60
Semantic Analyzer
=
⟨id, 1⟩
+
⟨id, 2⟩
*
⟨id, 3⟩
inttofloat
60
Intermediate Code Generator
t1 = inttofloat(60)
t2 = id3 * t1
t3 = id2 + t2
id1 = t3
Code Optimizer
t1 = id3 * 60.0
id1 = id2 + t1
Code Generator
LDF R2, id3
MULF R2, R2, #60.0
LDF R1, id2
ADDF R1, R1, R2
STF id1, R1

# Problem #2

Fahrenheit = centigrade *1.8 +32