CS202: Software Tools and Techniques for CSE

Lecture 5

Shouvick Mondal

shouvick.mondal@iitgn.ac.in
August2025

Let us shift our focus on analysis of a single version only...

WIKIPEDIA
The Free Encyclopedia

Search Wikipedia | Search

Program analysis

9 languages

Article | Talk | Read | Edit | View history | Tools

From Wikipedia, the free encyclopedia

For other uses, see Program analysis (disambiguation).

This article **needs additional citations for verification**. Please help improve this article by adding citations to reliable sources. Unsourced material may be challenged and removed.

Find sources: "Program analysis" – news · newspapers · books · scholar · JSTOR *(February 2018) (Learn how and when to remove this message)*

Contents hide

In computer science, **program analysis**[1] is the process of analyzing the behavior of computer programs regarding a property such as correctness, robustness, safety and liveness. Program analysis focuses on two major areas: program optimization and program correctness. The first focuses on improving the program's performance while reducing the resource usage while the latter focuses on ensuring that the program does what it is supposed to do.

Program analysis can be performed without executing the program (static program analysis), during runtime (dynamic program analysis) or in a combination of both.

Part of a series on

Software development

- **Core activities** [show]
- **Paradigms and models** [show]
- **Methodologies and frameworks** [show]
- **Supporting disciplines** [show]
- **Practices** [show]
- **Tools** [show]

What is the role of a software tool here? Is there any…?

What is Program Analysis?

For an end-goal, identify “interesting aspects” of a program's representation.

What is Program Analysis?

For an end-goal,

identify “interesting aspects” of

a program's representation.

Checking security

Array index range

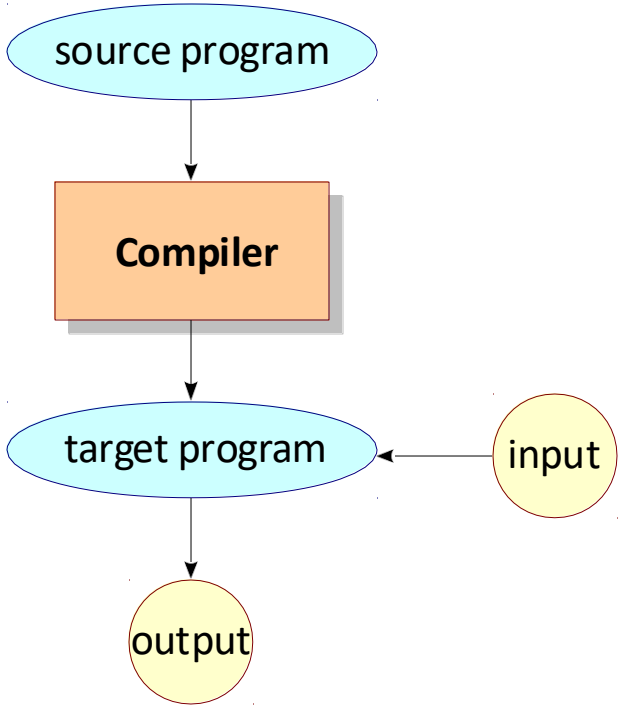
Source, AST, binary,
executed instruction

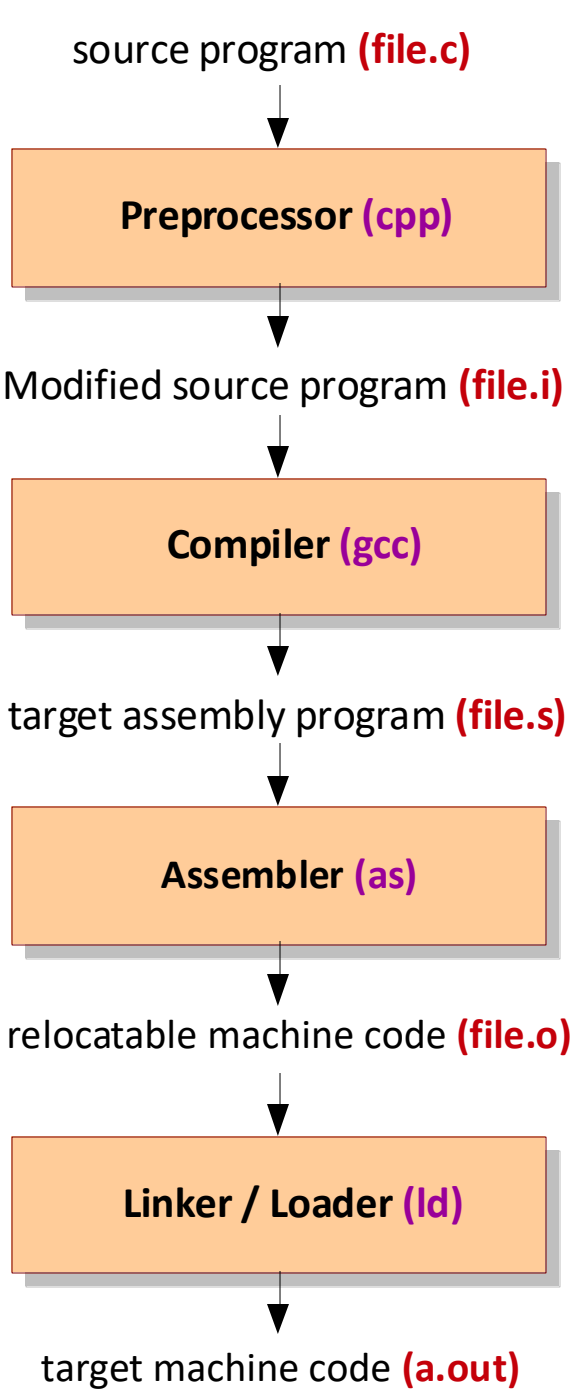
Examples

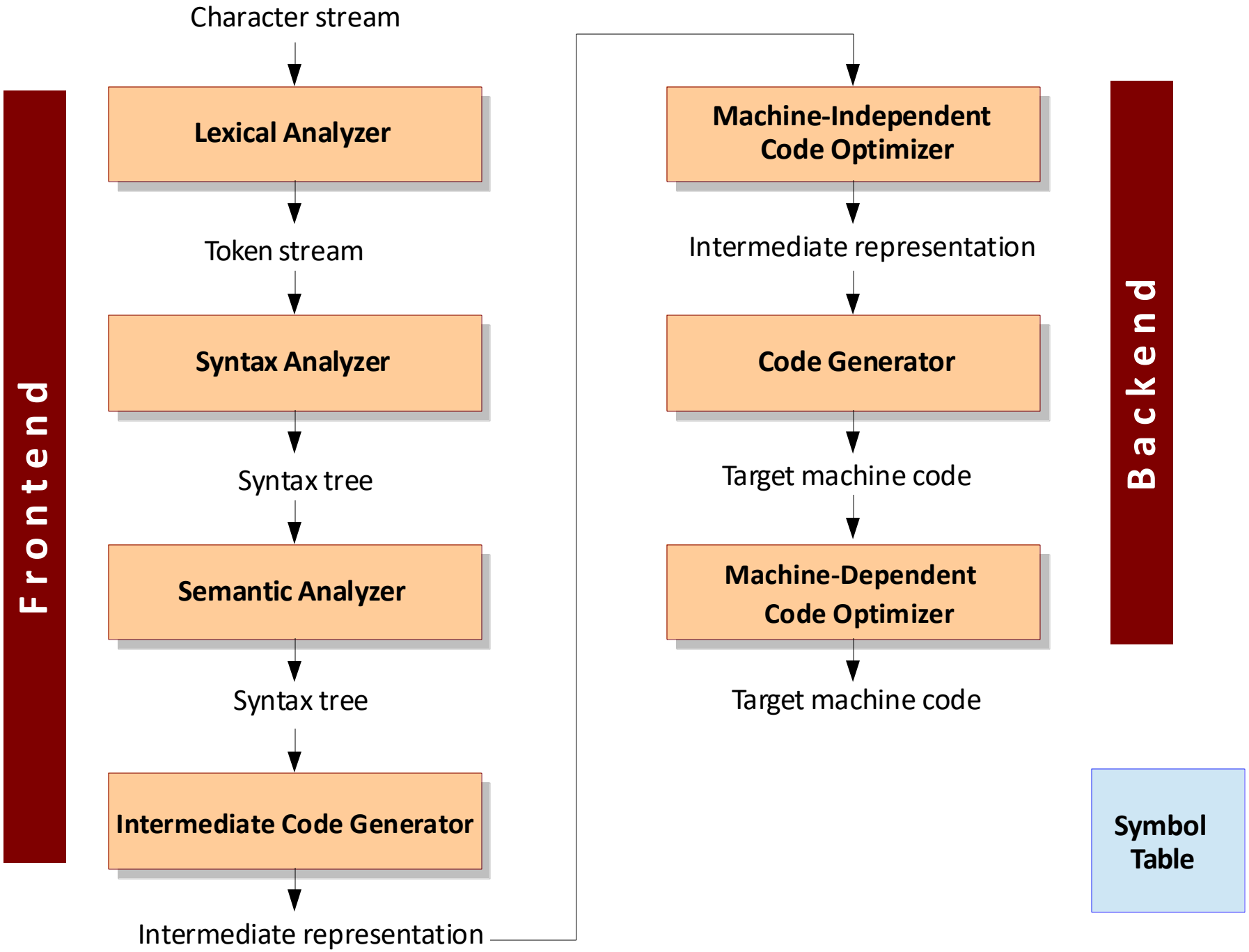
End goal	Interesting aspect
Dead code elimination	Reachability
Constant propagation	use-def
Security	Array index range, dangling pointers
Parallelization	Data dependence, SIMD opportunities
Debugging	Slice
Cache performance	Memory access pattern
Memory reduction	Live ranges
…	…

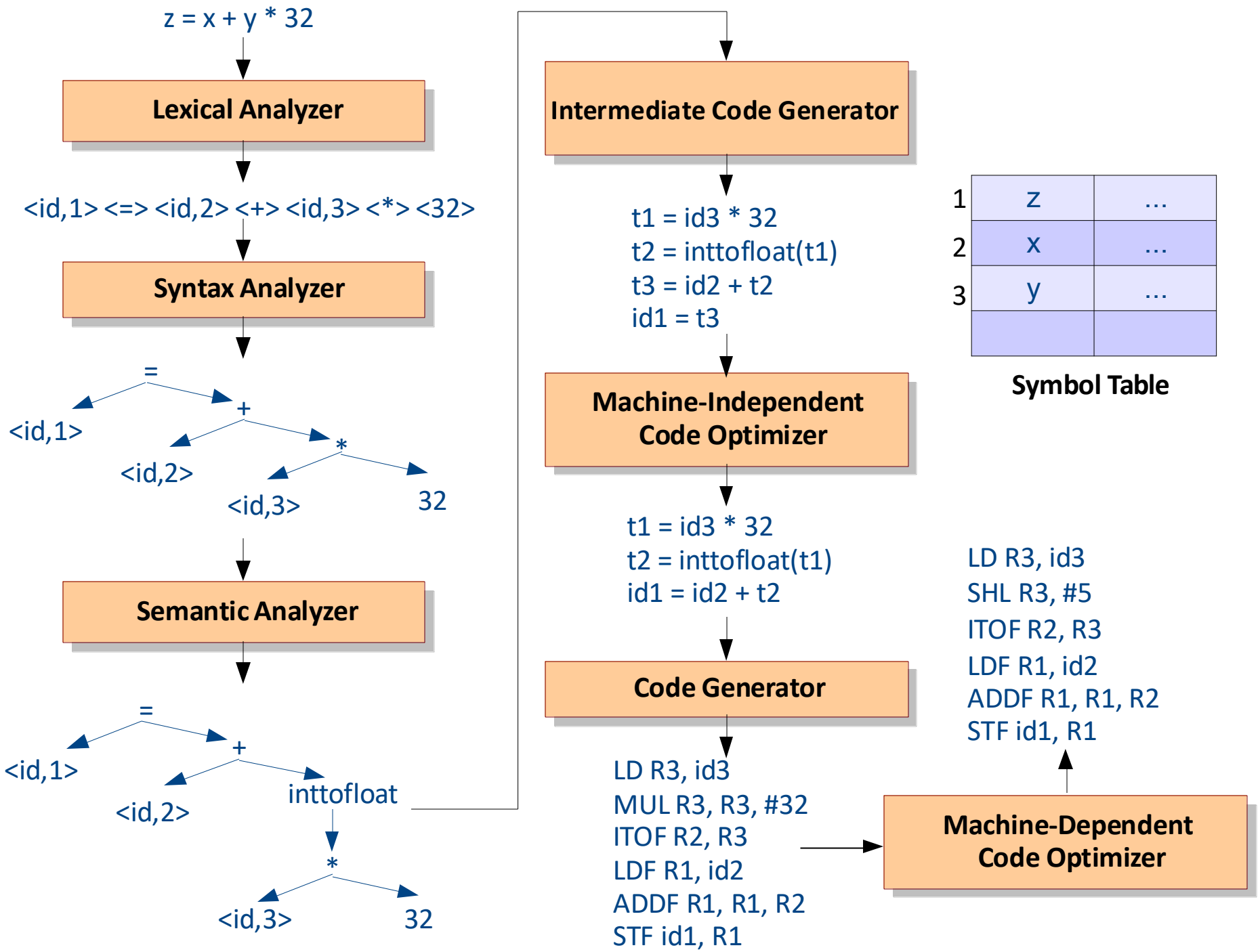
Program Analysis is often a pre-cursor to Optimization.

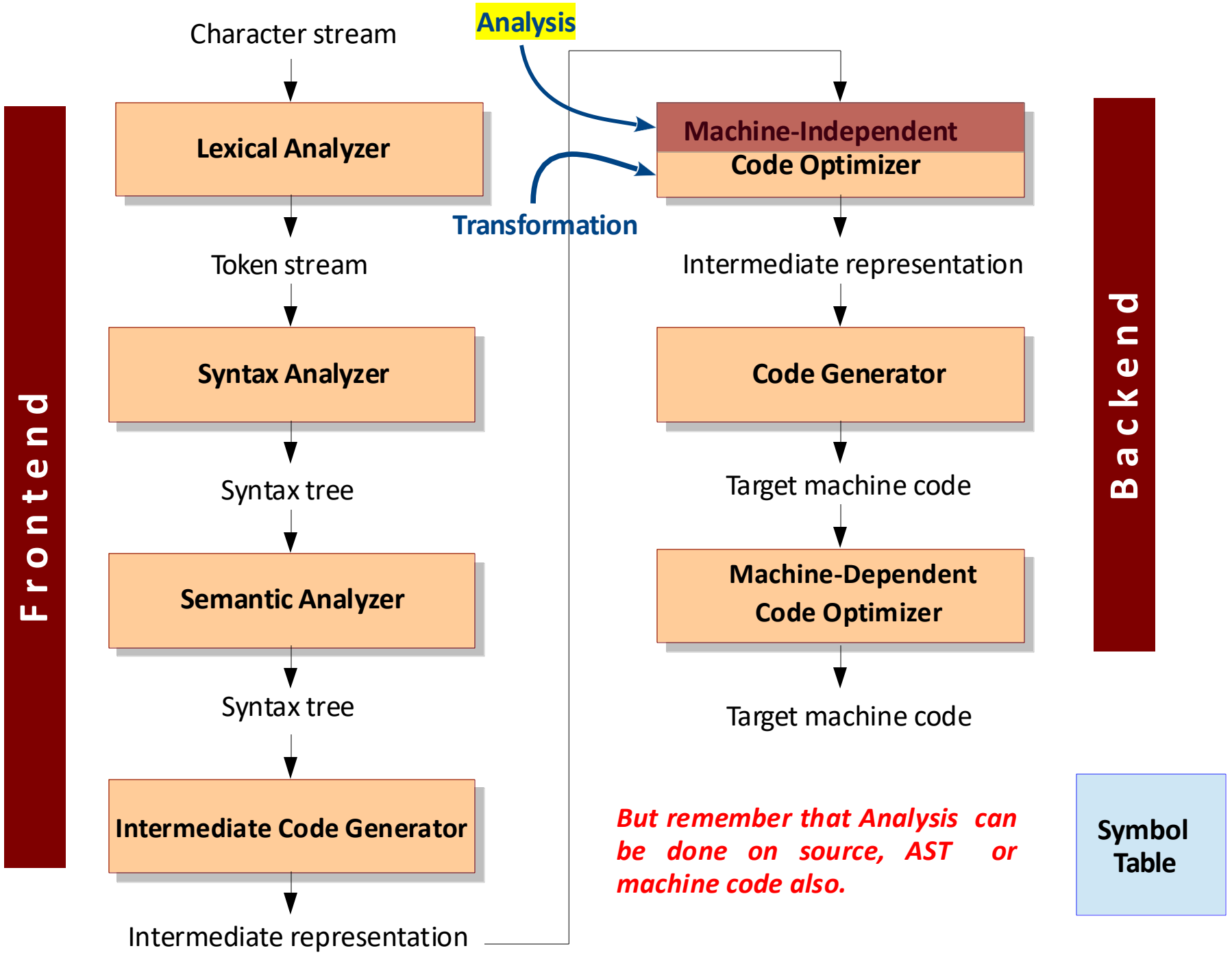
```
:~$ gcc file.c
:~$ ./a.out
```






Character stream
Lexical Analyzer
Token stream
Syntax Analyzer
Syntax tree
Semantic Analyzer
Syntax tree
Intermediate Code Generator
Intermediate representation
Frontend
Machine-Independent Code Optimizer
Intermediate representation
Code Generator
Target machine code
Machine-Dependent Code Optimizer
Target machine code
Backend
Symbol Table


z = x + y * 32
Lexical Analyzer
<id,1> <=> <id,2> <+> <id,3> <*> <32>
Syntax Analyzer
=
<id,1>
+
<id,2>
*
<id,3>
32
Semantic Analyzer
=
<id,1>
+
<id,2>
inttofloat
*
<id,3>
32
Intermediate Code Generator
t1 = id3 * 32
t2 = inttofloat(t1)
t3 = id2 + t2
id1 = t3
Machine-Independent Code Optimizer
t1 = id3 * 32
t2 = inttofloat(t1)
id1 = id2 + t2
Code Generator
LD R3, id3
MUL R3, R3, #32
ITOF R2, R3
LDF R1, id2
ADDF R1, R1, R2
STF id1, R1
Machine-Dependent Code Optimizer
LD R3, id3
SHL R3, #5
ITOF R2, R3
LDF R1, id2
ADDF R1, R1, R2
STF id1, R1
1 z ...
2 x ...
3 y ...
Symbol Table


Analysis
Character stream
Lexical Analyzer
Token stream
Syntax Analyzer
Syntax tree
Semantic Analyzer
Syntax tree
Intermediate Code Generator
Intermediate representation
Transformation
Machine-Independent
Code Optimizer
Intermediate representation
Code Generator
Target machine code
Machine-Dependent
Code Optimizer
Target machine code
Frontend
Backend
But remember that Analysis can be done on source, AST or machine code also.
Symbol Table

Texts, References, and Acknowledgements

Online:

- Continuous Integration and Delivery (**CircleCI**: https://circleci.com)
- http://www.cse.iitm.ac.in/~rupesh/teaching/pa/jan19

Textbook:

- Sharp, J. (2022). *Microsoft Visual C# Step by Step*, 10th edition, Microsoft Press.
- Watson, K., Nagel, C., Pedersen, J. H., Reid, J. D., & Skinner, M. (2008). *Beginning Microsoft Visual C# 2008*. John Wiley & Sons.
- Mark J. Price (2024). *C# 13 and .NET 9 – Modern Cross-Platform Development Fundamentals*, 9th edition, Packt Publishing Ltd.

Reference:

- Soni, M. (2016). *DevOps for Web Development*. Packt Publishing Ltd.
- Yusuf Sulistyo Nugroho, Hideaki Hata, and Kenichi Matsumoto. 2020. *How different are different diff algorithms in Git? Use* `--histogram` *for code changes*. Empirical Softw. Engg. 25, 1 (Jan 2020), 790–823.