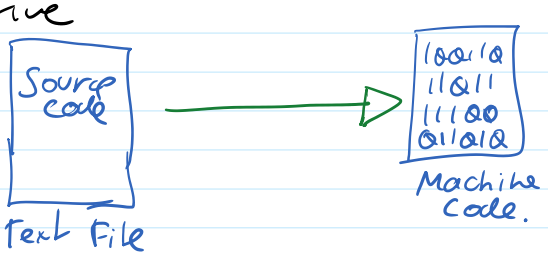
2 Programming Language Implementation

Wednesday, January 24, 2024 12:00 PM

¿ How are programming languages implemented?

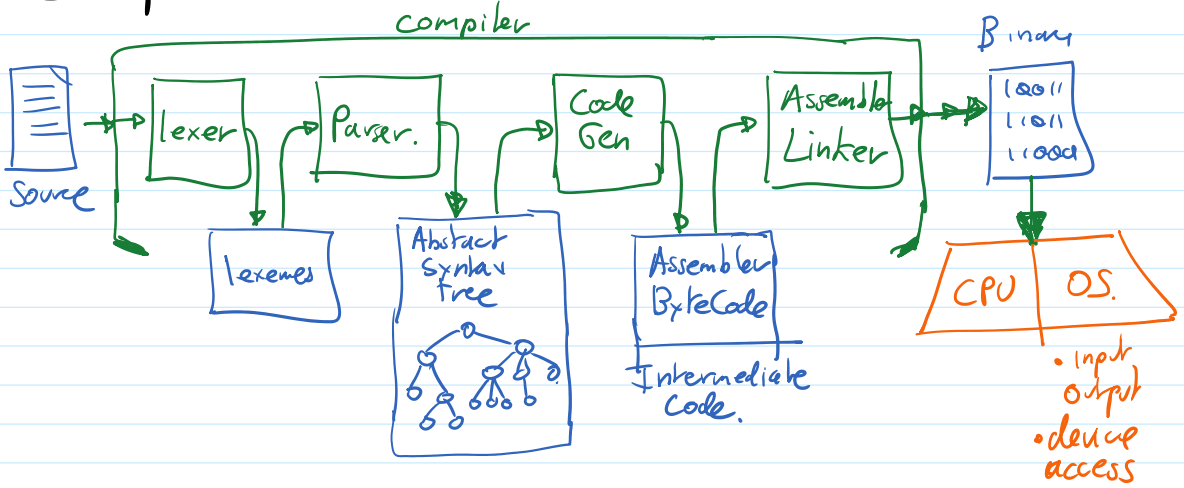
- Compilers
- Interpreters
- Hybrid .- combination of both.

- Objective



- Machine code is "relatively" simple.
  - fetch data from memory into cpu
  - Store data from cpu into memory
  - perform operations on two values on CPU (Add, substract, multiply, shift ...)
  - Move, change or modify the "instruction pointer" to skip or repeat instructions.

- Compilers:



Angel at 1/24/2024 12:43 PM

- lexer.- spit input into atomic components.

e.g. apple = banana * (orange + 7.3) *=

Lexemes.

- Parser :- recognize whether lexemes form a valid program under the rules of the language.

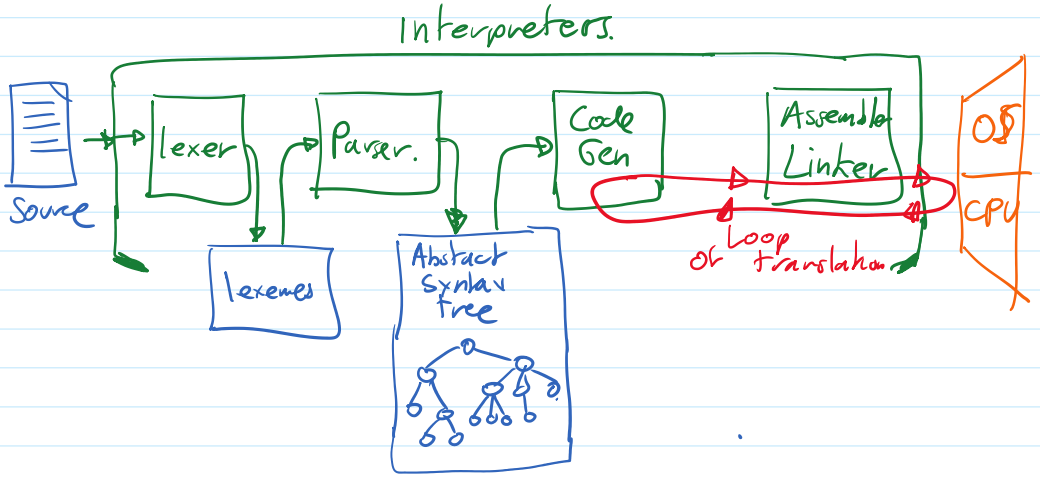
apple / = / 3dfx / [/ + / orange / + / [/] / 7 /]

E.G. FORTRAN, C, C++, Pascal, D, Go, Rust ...

Pro : - Speed of the final program
- Some I.P. protection.

Cons : - Portability.
- Speed of development
  ↳ compilation can be slow.

- INTERPETERS.



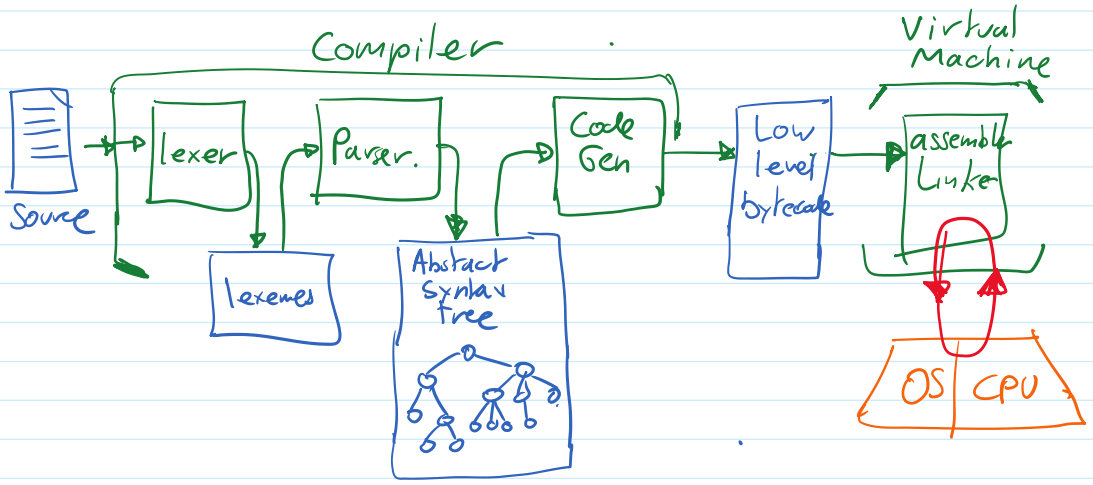
REPL = Read - Evaluate - Print Loop

E.G. Lisp, Python, JavaScript, Ruby, ML, BASIC

Pro : + Interactivity via REPL.
+ Speedy program development
+ portability.

Cons - speed of execution
- code is not hidden. - No IP protection.

HYBRID



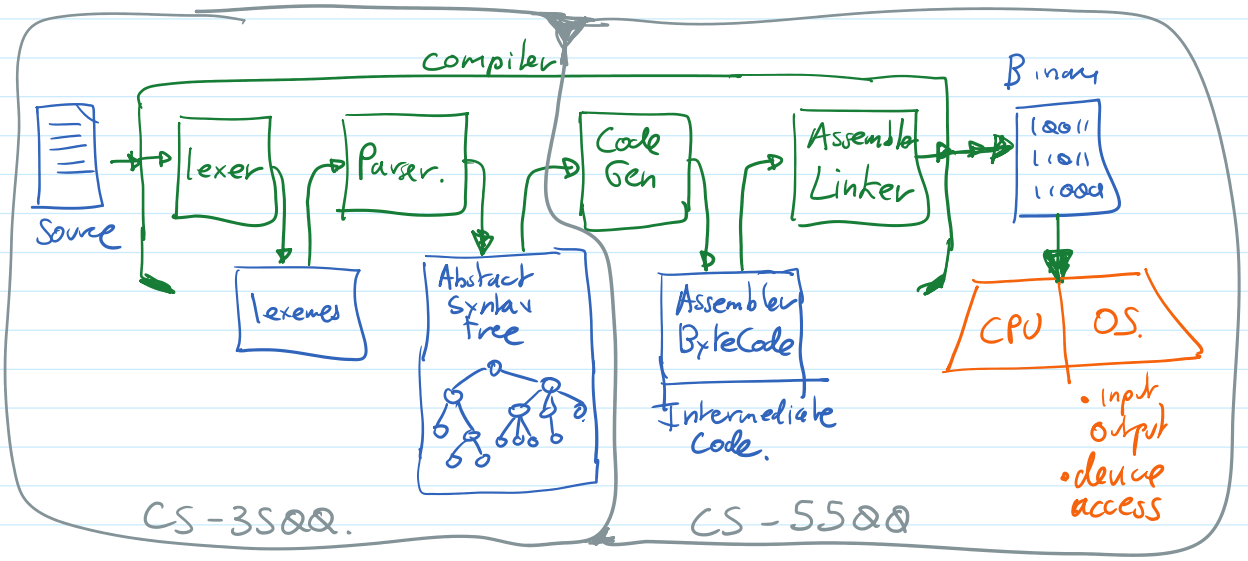
E.G. Java., C#, Scala, Kotlin, Pascal

Pros: + Speed > Interpreters
+ portability.
+ I.P. protection
+ inter-Language operability.

Cons: - Less speed than binary
- More Memory use by Virtual Machine.
- No R.E.P.L.

E.G.
idea: turn web browser into virtual machine. "WebASM"

This Course:



EOF