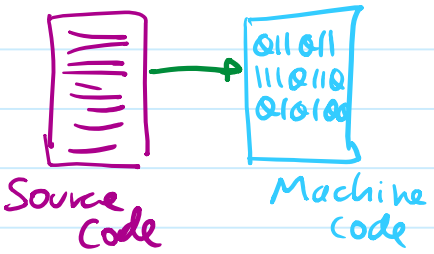
2 Programming Language Implementation

Monday, January 26, 2026 3:45 PM

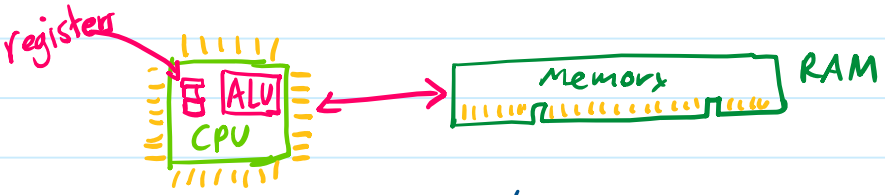
¿How are programming Languages implemented?

Objective



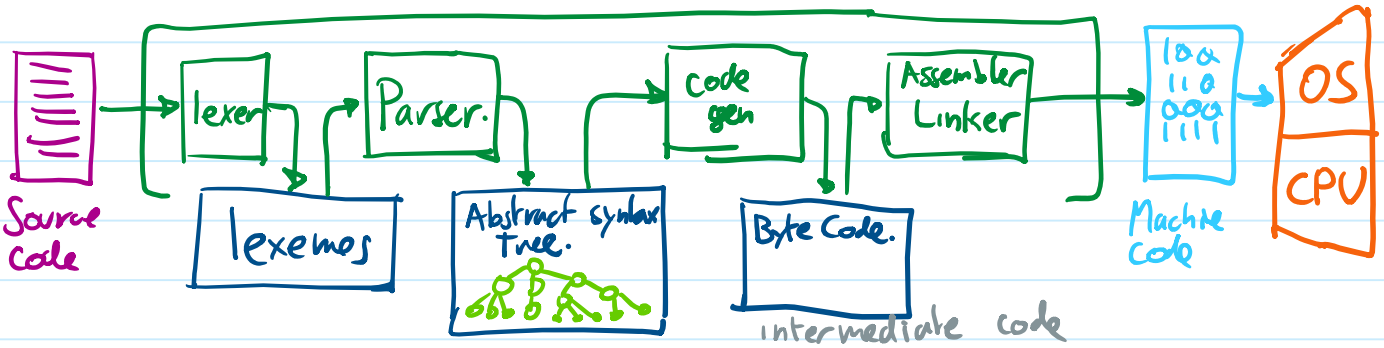
3 basic types
- Compilers
- interpreters
- hybrid (combination of both)

Note:
- Machine Code is specific CPU + OS
- Machine Code is "Relatively" Simple



1.- Perform operations in the ALU
2.- Fetch/Store data in memory
3.- Manipulate "instruction pointer" to implement loops/conditionals.

1) Compilers



Lexer:- split text into atomic components.

Eg apple|+=|banana|*|(|orange|+|7.35|);|

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

apple␣=␣3dfx␣[␣**␣orange␣*␣+␣(␣]␣7␣}␣;

Eg. Compilers.

Fortran, C, C++, Pascal, D, Rust, Go, zig

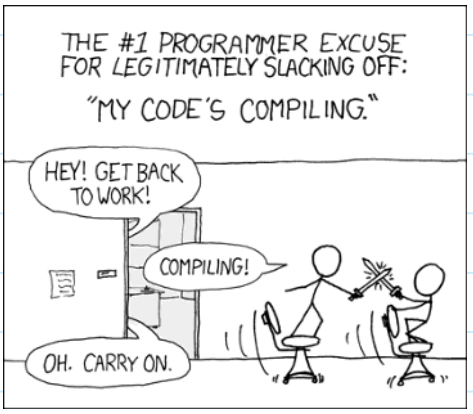
Pro ⊕ speed of final program

⊕ some i.p. protection

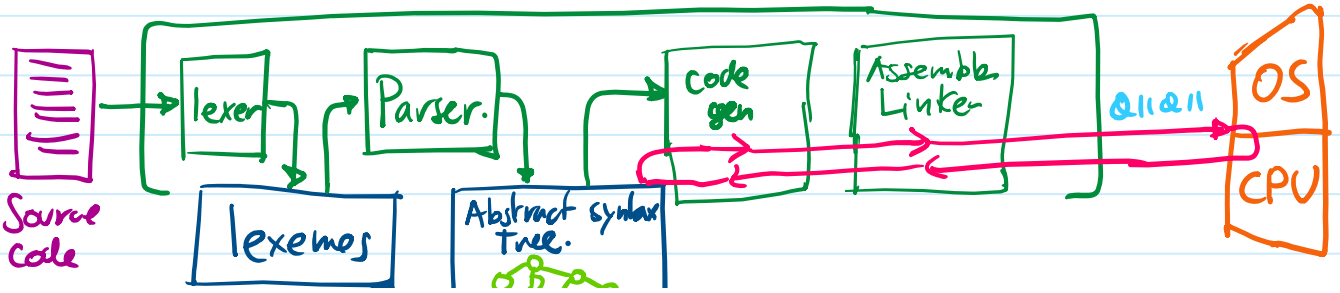
Cons ⊖ Portability

⊖ Development Flexibility

- Compilation can be slow.



2) Interpreters



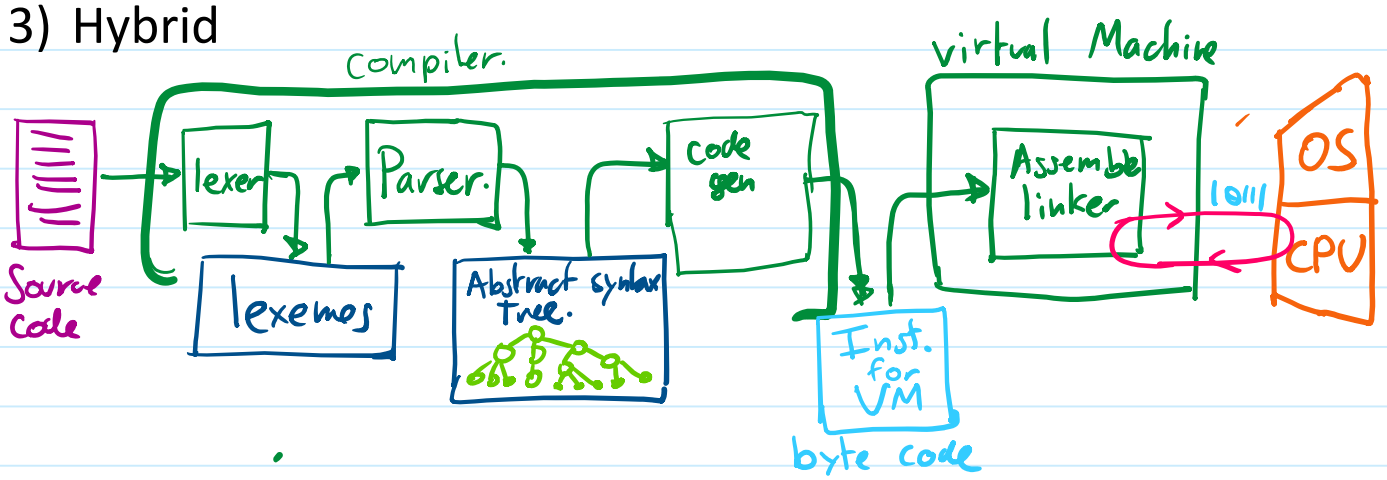
- REPL - Read → Evaluate → Print → Loop

E.g. LISP, Python, JavaScript, Lua, Ruby, ML, BASIC

Pro ⊕ interactivity via REPL
⊕ speedy program development
⊕ portability

Cons ⊖ no IP protection (Source Code is not hidden)
⊖ Execution Speed

3) Hybrid



Eg. Java, Scala, Kotlin, Pascal, C#

Pros ⊕ Speed > Interpreters.
⊕ Portability
⊕ IP Protection
⊕ inter-language operability

Cons ⊖ Speed < compilers
⊖ Memory use of virtual Machines
⊖ No REPL

- The "Web Browser"
  - A JavaScript interpreter.
  - A virtual Machine. Web ASM

- This course: