

O QUE É UM COMPILADOR?

- Não é um interpretador...
- É um *tradutor*.
- Mas preserva a equivalência semântica.

COMO SE CONSTRÓI UM COMPILADOR?

- Pipeline de processos independentes:
 1. Analisador lexical
 2. Analisador sintático
 3. Analisador semântico
 4. Optimizador
 5. Gerador de código
- Uso de “compiler compiler” tools:
 - flex/bison
 - JFlex/Cup
 - ANTLR

GERAÇÃO DE CÓDIGO

- Normalmente, código “low-level”
 - Assembly (MIPS, x86, x64, etc)
 - Bytecode (JVM, PHP/Zend, etc)

IDENTIFICAÇÃO DE SÍMBOLOS

- Palavras (e sua classificação lexical)
- Numerais (literais numéricos)
- Pontuação
- Comentários (basicamente ignorados)

EM LINGUAGENS DE PROGRAMAÇÃO

Tipos de tokens:

- Palavras reservadas (if, return, for, int, char)
- Identificadores/nomes (a, printf)
- Operadores e pontuação (+ - == > ; { })
- Literais (valores numéricos, strings)

EXEMPLO (PORTUGUÊS)

o gato comeu o rato

(artigo nome verbo artigo nome)

EXEMPLO (LINGUAGEM DE PROGRAMAÇÃO)

if a==b then x=1; else x=2;

(kw var op var kw var op lit sep kw var op lit sep)

ANÁLISE SINTÁCTICA

- Uso de gramática regular (BNF, EBNF, ...)

FRASE := SUJEITO verbo PREDICADO

SUJEITO := artigo nome | nome

PREDICADO := artigo nome | nome | <vazio>

ANÁLISE SEMÂNTICA

Análise do significado/sentido da frase:

O rato comeu o gato

a:string; if a==1 then print 'a is 1';

Também chamada *análise de nomes e tipos*.

OPTIMIZAÇÃO

- Tornar mais rápido
- Usar menos memória
 - Uso de registos em vez de memória
 - Substituir variáveis por literais (quando são constantes)