DCC024 Linguagens de Programação
2020/01

Introdução ao Curso

Área de Linguagens de Programação DCC/UFMG

Princípios

- Conceitos que caracterizam linguagens de programação:
  - Sintaxe
  - Nomes
  - Tipos
  - Abstrações
  - Semântica
- Para qualquer linguagem:
  - seus criadores devem definir estes conceitos
  - seus programadores devem dominar estes conceitos

Sintaxe

- A *sintaxe* de uma linguagem de programação é a descrição precisa de *todos* os seus programas gramaticamente corretos

- Quando estudando sintaxe, nos perguntamos:
  - Qual é a gramática da linguagem?
  - Qual é o vocabulário?
  - Como erros de sintaxe são detectados?

Nomes

- Vários tipos de entidades em programas possuem *nomes*:
  - variáveis, tipos, funções, parâmetros, classes, objetos...

- Entidades nomeadas em programas são restritas de acordo com:
  - escopo
  - visibilidade
  - tipo
  - tempo de vida

Tipos

- Um *tipo* é uma coleção de valores e de operações sobre esses valores
  - Tipos simples
  - Tipos estruturados

- O *sistema de tipos* de uma linguagem pode ajudar a:
  - determinar que operações são permitidas
  - identificar erros de tipagem
  - otimizar certas operações

Abstrações

- Mecanismo para *generalização* de dados ou de computação:

Abstrações

- Mecanismo para *generalização* de dados ou de computação:
  - Procedimentos / funções
  - Módulos
  - Tipos de dados abstratos
  - Classes
  - Modelos de memória

Semântica

- O *significado* de um programa é definido pela *semântica* de sua linguagem.

- Ao estudas semântica, nos perguntamos:
  - Quando um programa é executado, o que acontece com os valores de suas variáveis?
  - O que cada elemento do programa faz?
  - Que modelo rege a execução, por exemplo com a chamada de uma função?
  - Como variáveis e objetos são alocadas na memória durante a execução?

Paradigmas de programação

- Um *paradigma de programação* é um padrão de construção de soluções que permeia um dado grupo de programas e linguagens

- Existem diversos paradigmas de programação:
  - Imperativo
  - Orientado a objeto (OO)
  - Funcional
  - Lógico

Paradigma imperativo

- Segue o clássico modelo von-Neumann:
  - Programa e dados são indistinguíveis na memória
  - Programa: sequência de comandos modificando um *estado* atual
  - Estado: valores de todas as variáveis quando o programa é executado
  - Programas maiores usam abstração através de procedimentos

- Exemplos de linguagens imperativas:

Paradigma imperativo

- Segue o clássico modelo von-Neumann:
  - Programa e dados são indistinguíveis na memória
  - Programa: sequência de comandos modificando um *estado* atual
  - Estado: valores de todas as variáveis quando o programa é executado
  - Programas maiores usam abstração através de procedimentos

- Exemplos de linguagens imperativas:

  C, C++...

Paradigma orientado a objeto (OO)

- Um programa OO é uma coleção de objetos que interagem trocando mensagens que modificam o estado atual

- Principais propriedades:
  - Encapsulamento de estado
  - Troca de mensagens
  - Herança
  - Subtipagem

- Exemplos de linguagens imperativas:

Paradigma orientado a objeto (OO)

- Um programa OO é uma coleção de objetos que interagem trocando mensagens que modificam o estado atual

- Principais propriedades:
  - Encapsulamento de estado
  - Troca de mensagens
  - Herança
  - Subtipagem

- Exemplos de linguagens imperativas:

  C++, Java, Python...

Paradigma funcional

- Programação funcional modela computação como uma coleção de funções (matemáticas)
  - Entrada : domínio
  - Saída : imagem
- Principais propriedades:
  - Composição
  - Recursão
  - Transparência referencial
- Exemplos de linguagens funcionais:

Paradigma funcional

- Programação funcional modela computação como uma coleção de funções (matemáticas)
  - Entrada : domínio
  - Saída : imagem
- Principais propriedades:
  - Composição
  - Recursão
  - Transparência referencial
- Exemplos de linguagens funcionais:

  Standard ML, Lisp, Haskell, F#, ...

Paradigma lógico

- Programação lógica declara que resultado o programa *deve* ter em vez de *como* obtê-lo

- Principais propriedades:
  - Programas como conjuntos de restrições a um problema
  - Computação de todas as soluções possíveis
  - Computação não-determinística

- Exemplos de linguagens imperativas:

Paradigma lógico

- Programação lógica declara que resultado o programa *deve* ter em vez de *como* obtê-lo

- Principais propriedades:
  - Programas como conjuntos de restrições a um problema
  - Computação de todas as soluções possíveis
  - Computação não-determinística

- Exemplos de linguagens imperativas:

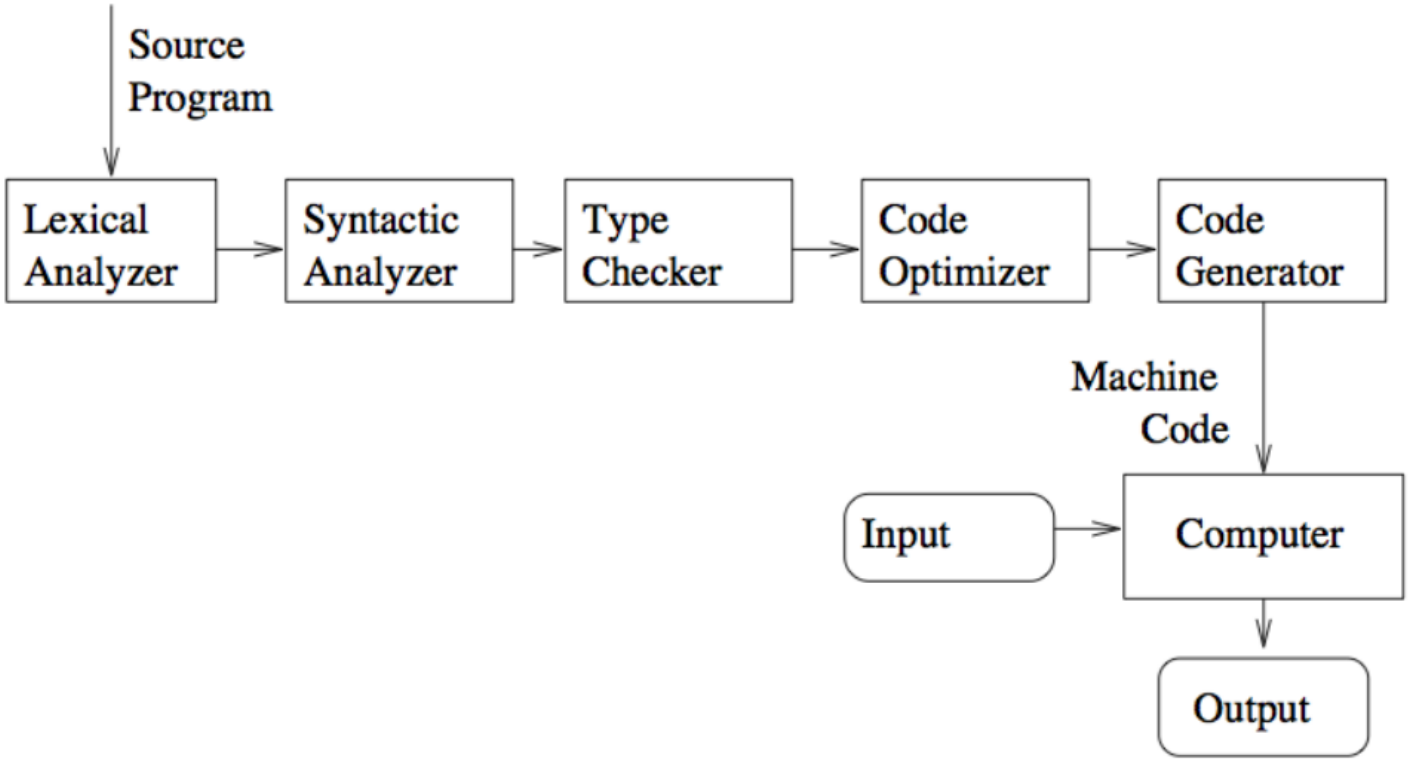
  Prolog, Datalog...

Compiladores e interpretadores

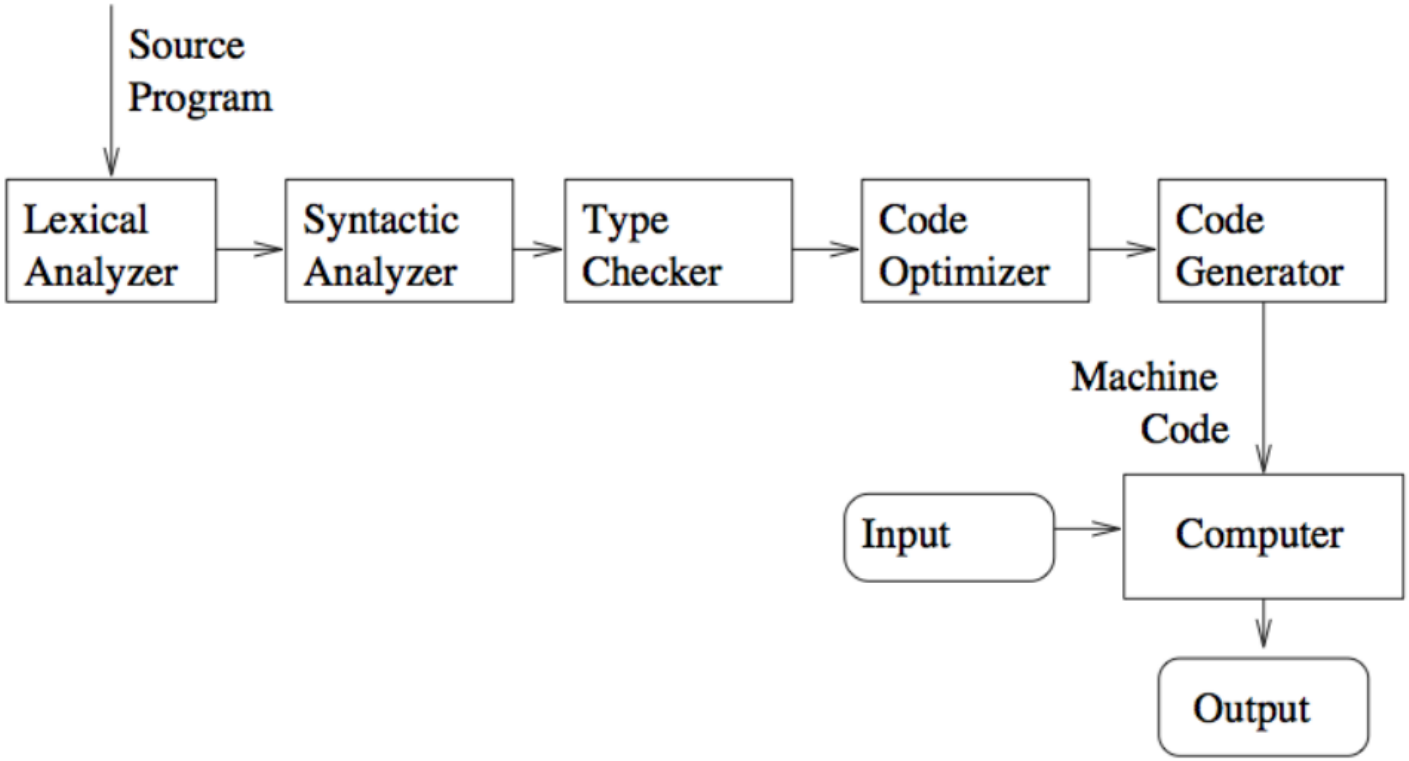
- Compilador: produz código de máquina

- Interpretador: executa instruções em uma máquina virtual

Compilação

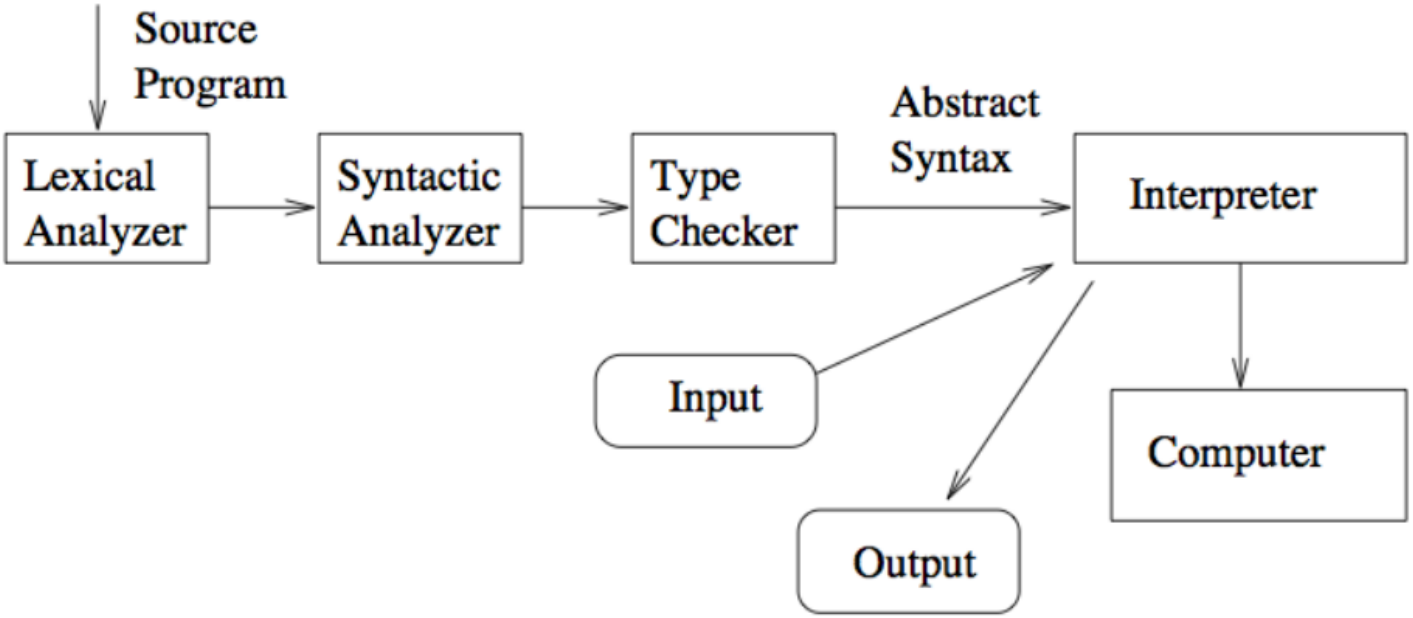

Source
Program
Lexical
Analyzer
Syntactic
Analyzer
Type
Checker
Code
Optimizer
Code
Generator
Machine
Code
Input
Computer
Output

Compilação

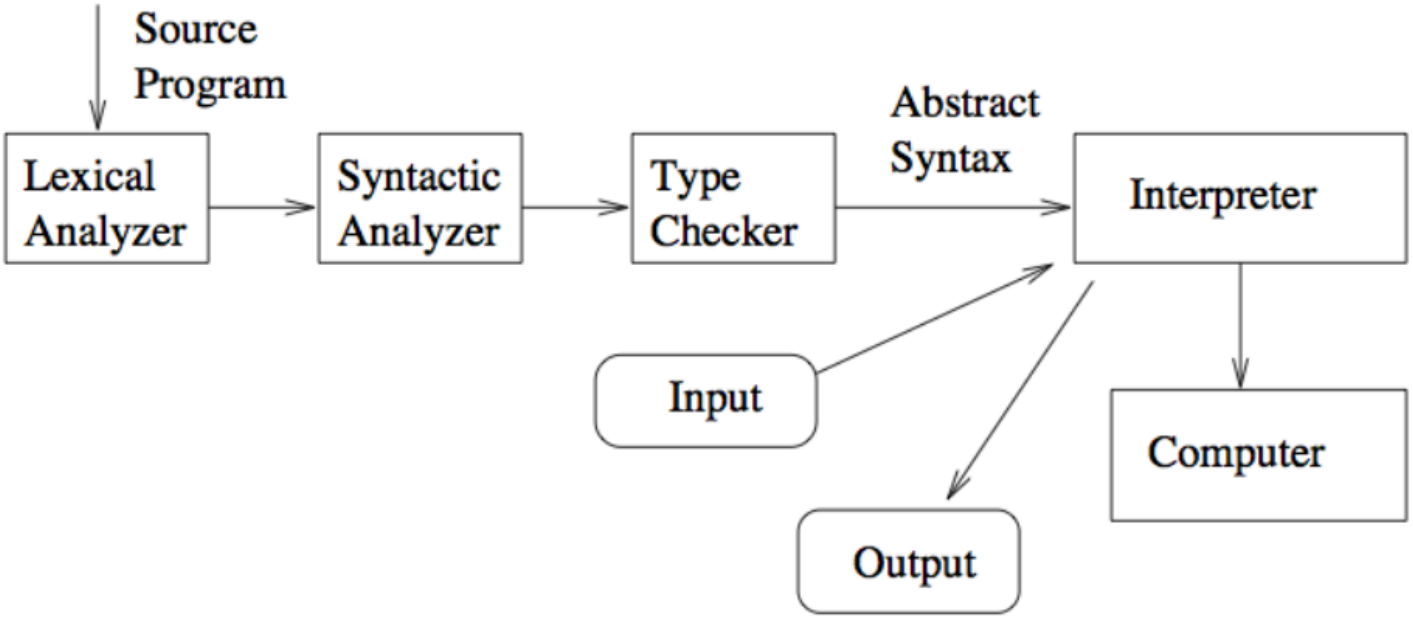


- Exemplos de linguagens compiladas:
  - C, C++, Rust, ...

Interpretação



Interpretação



- Exemplos de linguagens interpretadas:
  - Python, Javascript, ...

Conteúdos do curso

- O curso consistirá em cobrir conceitos de linguagens de programação no contexto dos paradigmas:
  - Funcional (utilizando ML)
  - Imperativo / OO (utilizando Python)
  - Lógico (utilizando Prolog)
- Introduções a estas linguagens serão feitas conforme cobrimos conceitos relevantes aos respectivos paradigmas
- Análises sintática e semântica serão pervasivos durante o curso