UNIVERSIDADE ESTADUAL DO RIO GRANDE DO SUL

UNIDADE GUAÍBA

Programação de Sistemas

Introdução

Celso Maciel da Costa

Guaíba, outubro 2013.

I. Execução de Programas

Programa em Linguagem de Máquina

- Uma linguagem de programação é um conjunto de ferramentas, regras de sintaxe e símbolos ou códigos que nos permitem escrever programas de computador.
- A primeira e mais primitiva linguagem de computador é a própria linguagem máquina (0's e 1's).
- Um programa era difícil, longo e principalmente caro para construir.
- Era também difícil de ser entendido por outros programadores.
- Essa complexidade levou è necessidade de desenvolver novas técnicas e ferramentas.

I. Execução de Programas

Linguagem de Montagem

- A resolução do problema passou pela criação de uma linguagem em que os códigos numéricos foram substituídos por mnemônicos.
- O nome dessa linguagem é ASSEMBLY LANGUAGE.
- Então será necessário um outro programa que leia o programa escrito nessa linguagem alternativa e o traduza para a linguagem nativa do computador!!!
- O processo de tradução da linguagem de montagem para a linguagem de máquina é realizada por um programa chamado ASSEMBLER.

I. Execução de Programas

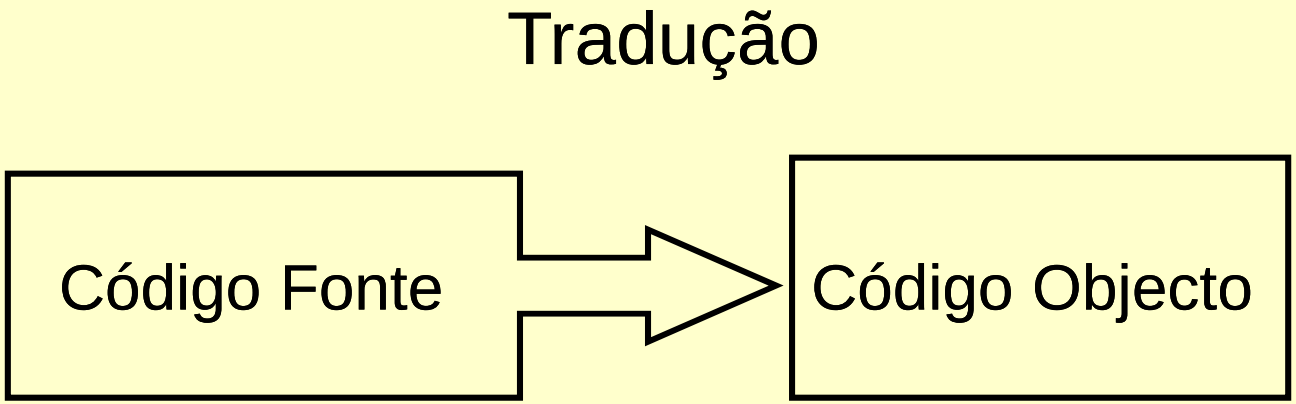
Linguagem de Programação

- Foram desenvolvidas diversas linguagens de programação:
  - **FORTRAN (1957)**
  - **ALGOL (1958)**
  - **COBOL (1959)**
  - **PASCAL (1963)**
  - **BASIC (1965)**
  - **ADA (1968)**
  - **C (1982) e mais tarde o C++ (1986)**
  - **Etc….**
- Estas novas linguagens foram afastando cada vez mais o programador do nível de máquina.

I. Execução de Programas

Tradução

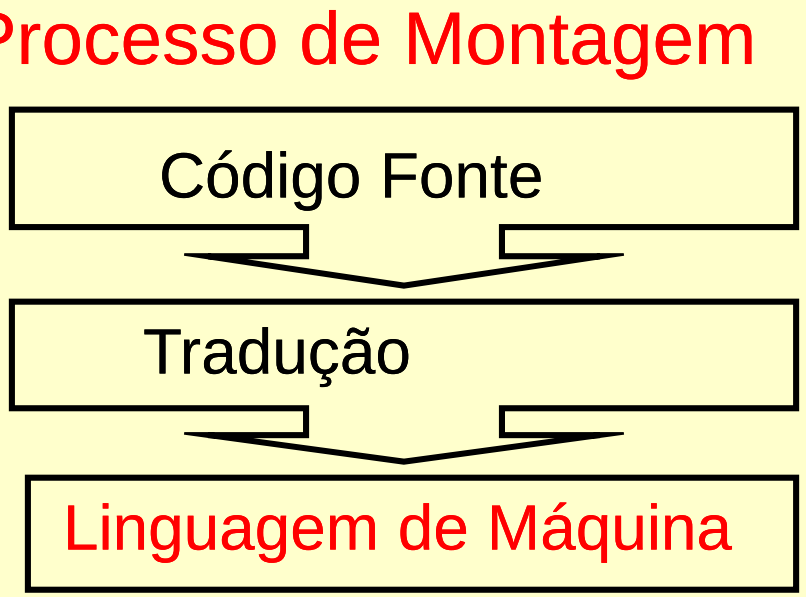
- Os programas em linguagem de alto nível também precisam ser traduzidos para linguagem de máquina.



I. Execução de Programas

Montagem

- O processo de montagem traduz um programa escrito numa LP num programa equivalente em linguagem de máquina.



II . Compiladores

O que é um compilador

- Um **compilador** tem a finalidade de converter uma linguagem – Linguagem Fonte – de fácil escrita e leitura para os programadores, numa linguagem – Linguagem alvo ou objeto – que pode ser executada pelas máquinas.
- O código executável gerado pelo compilador é dependente do sistema operacional e da linguagem de máquina para o qual o código fonte foi traduzido.

II . Compiladores

O que é um compilador

- Os compiladores são por vezes classificados como uni-passo, multi-passo, optimizador, ou corretor de erros, dependendo da forma como foram construídos ou da funcionalidade desejada.
- Começaram a aparecer no início da década de 50.
- Muito do trabalho inicial dos compiladores resumia-se na tradução de fórmulas aritméticas para código máquina.

II . Compiladores

O que é um compilador

- O primeiro compilador de FORTRAN, por exemplo, demorou 18 meses para ser implementado.

- Boas linguagens de implementação, ambientes de programação, e ferramentas de software foram desenvolvidas e facilitaram o desenvolvimento de compiladores.

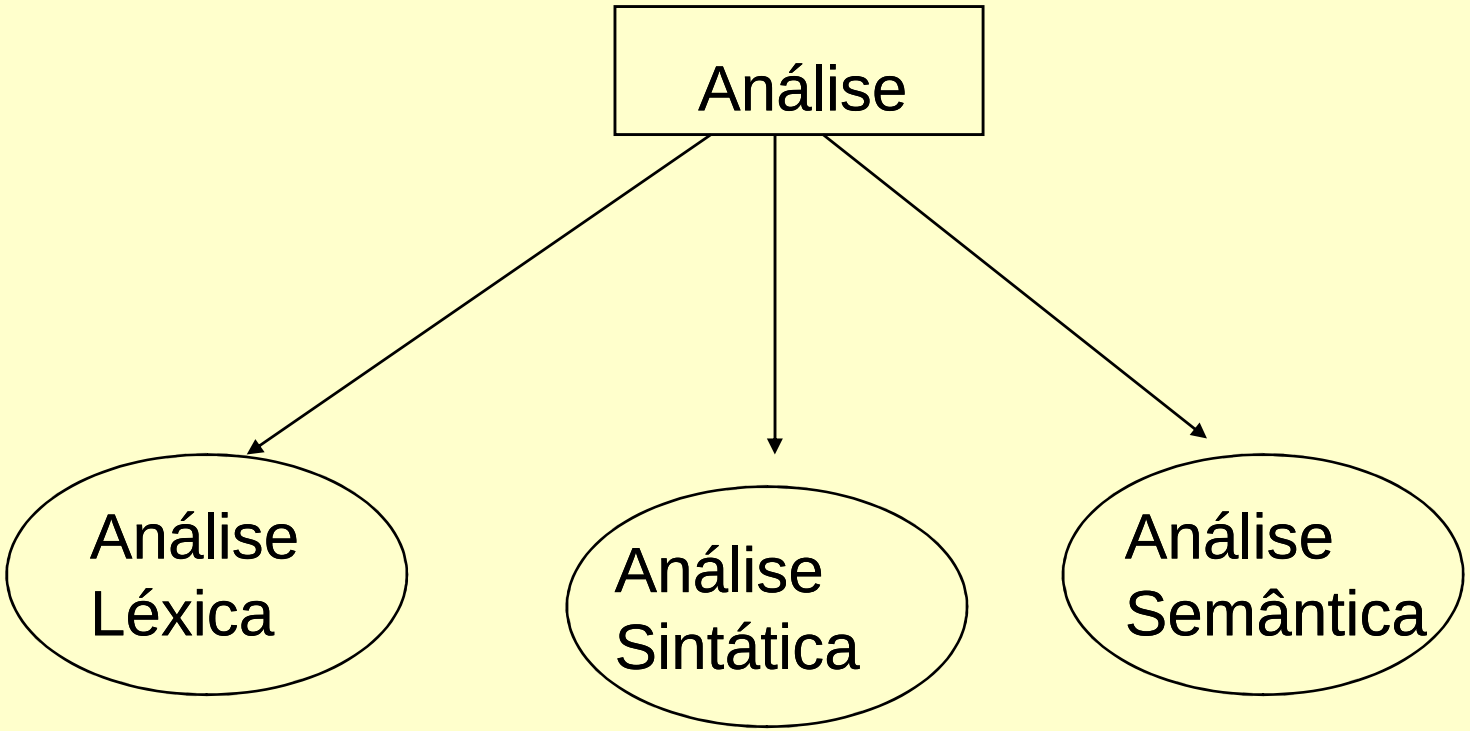
II . Compiladores

Modelo Análise- síntese da compilação

- Podemos dividir o processo de compilação em duas fases:
  - **Análise : parte o programa fonte em peças constituintes e cria uma representação intermédia do programa fonte.**
  - **Síntese : Constrói o desejado programa alvo (código de máquina) a partir da representação intermédia.**
- A parte da síntese é a que requer técnicas mais especializadas.

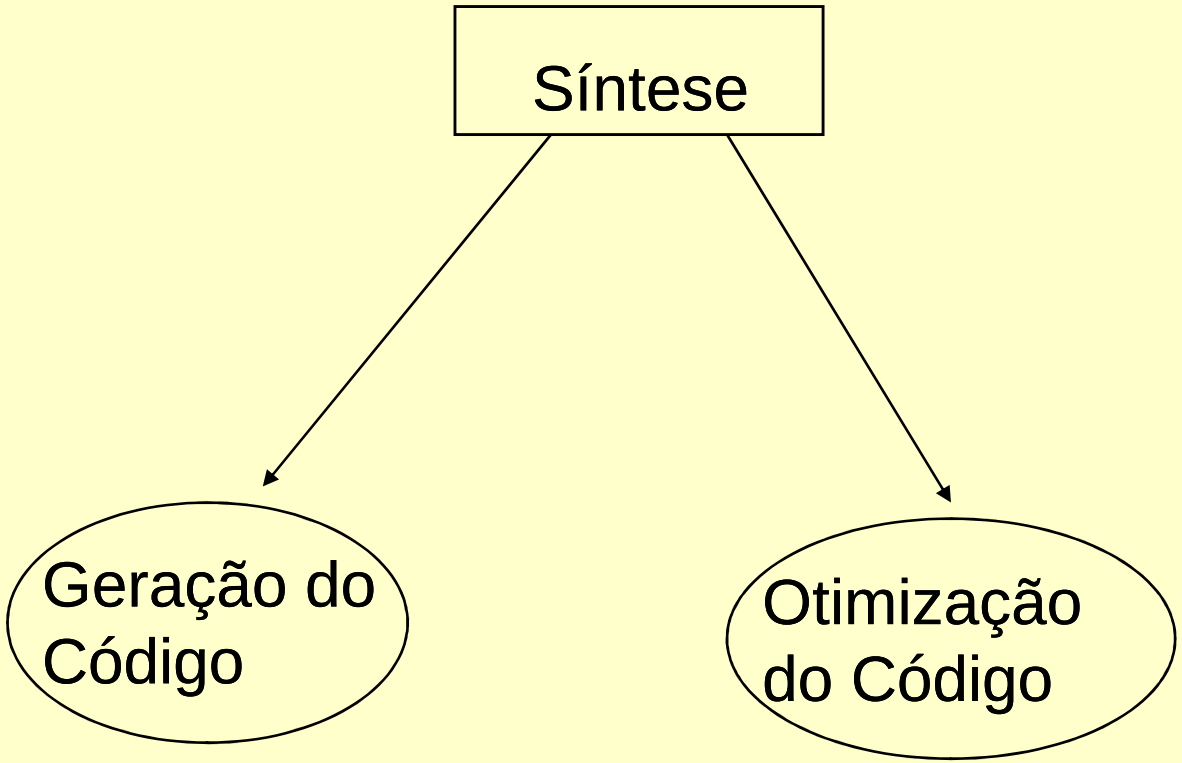
II . Compiladores

Modelo Análise- síntese da compilação



II . Compiladores

Modelo Análise- síntese da compilação



II . Compiladores

Contexto de um compilador

Muitos outros programas podem ser necessários para criar um programa alvo executável.

II . Compiladores

Componentes de um compilador

- Pré-processadores: produzem a entrada para os compiladores.
- Montadores: Alguns compiladores produzem código Assembler que é passado para um montador para posterior processamento.
- Alguns compiladores produzem o trabalho dos montadores.

II . Compiladores

Componentes de um compilador

- **Montagens bi-passo:**
  - **I Passo - todos os identificadores que denotam localizações de armazenamento, são encontrados e armazenados numa tabela de símbolos**
  - **II Passo - traduz cada código de operação para sequências de bits representando essa operação na linguagem de máquina**
- **Carregadores e editores de união (Linker):**
  - **Carregar consiste em colocar na memória o código de máquina, nas localizações convenientes.**
  - **O ligador permite fazer um único programa dos vários arquivos de código de máquina relocável.**

II . Compiladores

- **Bibliotecas:**
  - O desenvolvimento de um programa certamente utilizará diversas operações que são comuns a muitos outros programas.
  - Um programa de alto nível possivelmente conterá diversas chamadas de biblioteca.
  - Essas funções não devem ser confundidas com as instruções da linguagem – na realidade, são pequenos programas externos.

II . Compiladores

Análise do programa fonte

- Na compilação a análise consiste em 3 partes:
  - **Análise Léxica ou Linear:**
    - **Em que a cadeia de caracteres que forma a estrutura do programa fonte é lido da esquerda para a direita e agrupado em *tokens* que são sequências de caracteres.**
    - **A sua função básica é o reconhecimento e a classificação das estruturas elementares ou classes sintáticas das linguagens.**

II . Compiladores

Análise do programa fonte

- **Análise sintática ou hierárquica:**
  - **Na qual caracteres ou tokens são agrupados hierarquicamente em coleções aninhadas, com sentido coletivo.**
  - **Verifica se a estrutura geral do texto ou programa fonte está correta.**

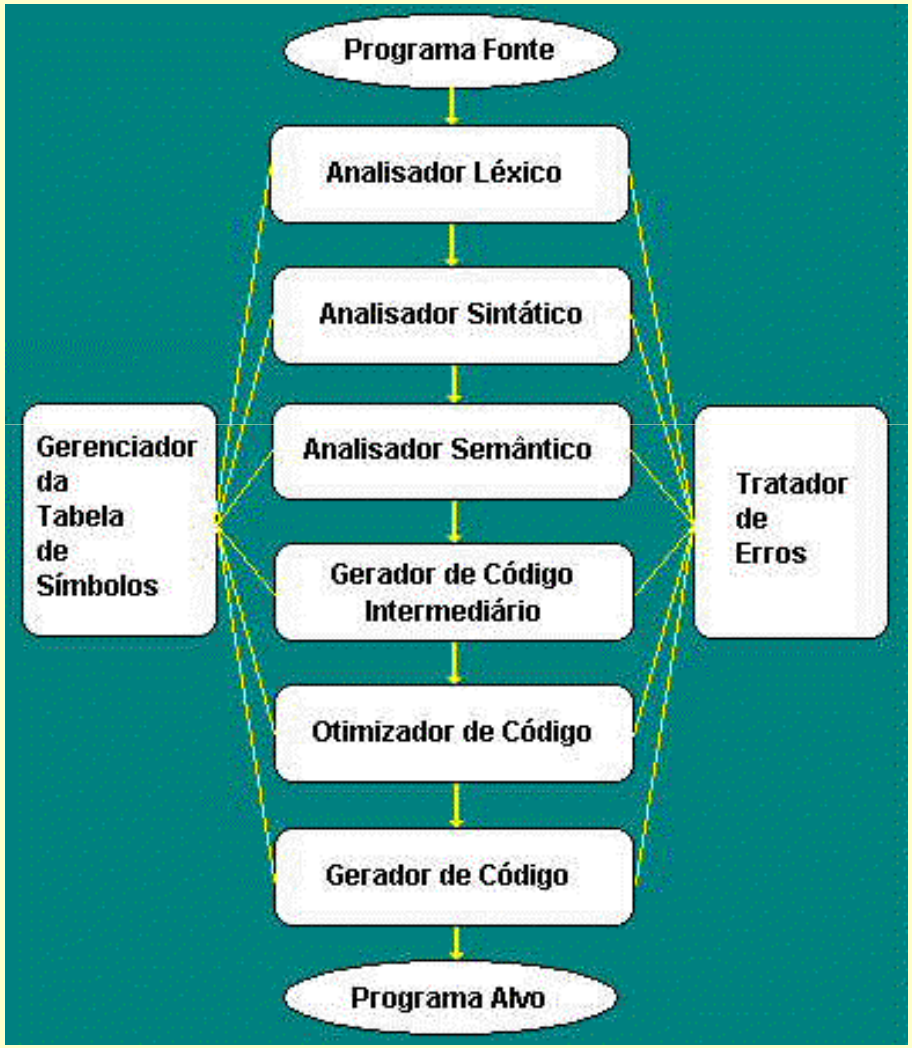
II . Compiladores

Análise do programa fonte

- **Análise semântica:**
  - **Verifica se o programa fonte tem erros semânticos e reúne a informação dos tipos para a fase de gerador de código subsequente.**
  - **Um componente importante da análise semântica é a verificação do tipo.**

II . Compiladores

Fases de um compilador



II . Compiladores

Fases de um compilador

- Gerenciador da tabela de símbolos:
  - Uma função essencial de um compilador é registar os identificadores usados no programa fonte e colecionar informação sobre vários atributos de cada identificador.
  - Uma tabela de símbolos é uma estrutura de dados contendo cada identificador, com campos para os atributos do identificador.

II . Compiladores

Fases de um compilador

- Tabela de códigos:
  - É uma estrutura criada pela análise semântica de um compilador, que mantém por algum tempo as linhas do código intermediário geradas.
  - Em geral as linhas de código geradas permanecem nesta tabela enquanto não estão totalmente analisadas.

II . Compiladores

Fases de um compilador

- Detecção de erros e aviso do erro:
  - Cada fase pode encontrar erros. Porém, depois de descobrir um erro, a fase tem de ocupar-se de alguma maneira com aquele erro, para que a compilação possa prosseguir.
  - As fases de análise sintática e semântica normalmente tratam de uma grande fração dos erros detectáveis pelo compilador.

II . Compiladores

Fases de um compilador

- Geração de código intermediário:
  - Depois da análise sintática e semântica, alguns compiladores geram uma representação intermediária do programa fonte.
  - Pode-se pensar nesta representação intermediária como um programa para uma máquina abstrata.

II . Compiladores

Fases de um compilador

- Otimização do código:
  - Esta fase tenta melhorar o código intermediário, de forma a que resulte num código de máquina mais rápido de ser executado.
- Geração do código:
  - A fase final do compilador é a geração de código alvo (código de máquina).
  - Neste ponto, após o programa fonte ter sido analisado e aprovado, segundo a sua sintaxe, e livre de erros semânticos, o compilador tem condições de gerar um programa equivalente na linguagem alvo.

II . Compiladores

Facores condicionantes da organização física dos compiladores

- Dividir o processo de compilação em diversas fases "lógicas" permite um melhor entendimento do processo como um todo e leva a uma implementação mais estruturada.

- A eficiência e os recursos disponíveis na máquina hospedeira do compilador influenciam de maneira decisiva um item importantíssimo na implementação de um compilador: o número de passos de compilação, para poder otimizar o tempo de compilação.

III . Interpretadores

Como funcionam os interpretadores

- O funcionamento dos interpretadores é muito parecido com o dos compiladores.
- O interpretador traduz o código linha a linha.
- O código fonte não é totalmente traduzido antes de ser executado.
- Não existem fases distintas nem se produz código intermediário.
- Passa o tempo todo lendo e traduzindo código.

III . Interpretadores

Os Exemplos de interpretadores

- **Excel, Access, ... ;**
- **SmallTalk;**
- **AutoLisp;**
- **Lisp.**

IV . Comparação

	Vantagens	Desvantagens
Compiladores	Execução mais rápida	Várias etapas de tradução
	Permite estruturas de programação mais completas	Programação final é maior, necessitando mais memória para a sua execução
	Permite a otimização do código fonte	Processo de correção de erros e depuração é mais demorado
Interpretadores	Depuração do programa é mais simples	Execução do programa é mais lenta
	Consome menos memória	Estruturas de dados simples
	Resultado imediato do programa ou rotina desenvolvida	Necessário fornecer o programa fonte do usuário