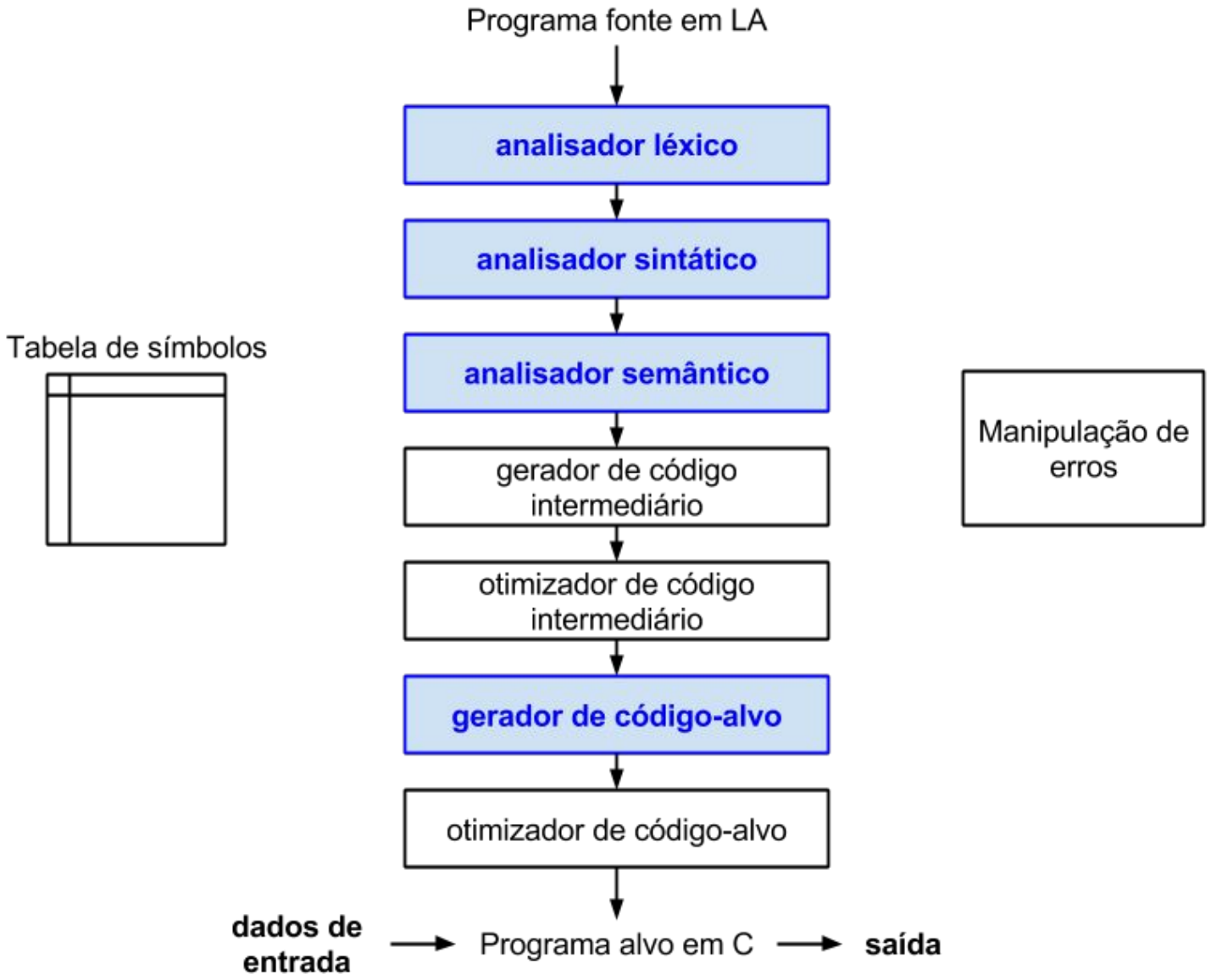
Construção de Compiladores 2

Descrição do Trabalho 1

Prof. Dr. Daniel Lucrédio
Profa. Dra. Helena de Medeiros Caseli
Departamento de Computação
UFSCar

Trabalho 1 - Compilador para LA



Trabalho 1 - Compilador para LA

- Especificação
  - Usando um gerador automático (p. ex. ANTLR), gerar um compilador para a linguagem LA
    - Análise Léxica + Análise Sintática
      - De acordo com o que está especificado na Gramática da LA
    - Análise Semântica
      - Capaz de acusar 6 erros semânticos listados no próximo slide
    - Geração de Código
      - Geração de código em C correspondente à LA
      - Utilize a apostila do Prof. José Guimarães como referência (disponível no Moodle, veja Compilador 10)

Trabalho 1 - Compilador para LA

Erros semânticos a serem relatados:

1) Identificador (variável, constante, procedimento, função, tipo) já declarado anteriormente no escopo
   a) O mesmo identificador não pode ser usado novamente no mesmo escopo mesmo que para categorias diferentes
2) Tipo não declarado
3) Identificador (variável, constante, procedimento, função) não declarado
4) Incompatibilidade entre argumentos e parâmetros formais (número, ordem e tipo) na chamada de um procedimento ou uma função
5) Atribuição não compatível com o tipo declarado
6) Uso do comando 'retorne' em um escopo não permitido

Trabalho 1 - Compilador para LA

4. Incompatibilidade entre argumentos e parâmetros formais (número, ordem e tipo) na chamada de um procedimento ou uma função

A quantidade e tipo dos argumentos deve ser exata

```
endereço → ponteiro
real → real
inteiro → inteiro
literal → literal
logico → logico
registro → registro (com mesmo nome de tipo)
```

Trabalho 1 - Compilador para LA

5. Atribuição não compatível com o tipo declarado

Atribuições possíveis

```
ponteiro ← endereço
(real | inteiro) ← (real | inteiro)
literal ← literal
logico ← logico
registro ← registro (com mesmo nome de tipo)
```

As mesmas restrições são válidas para expressões, por exemplo, ao tentar combinar um literal com um logico (como em literal + logico) deve dar tipo_indefinido e inviabilizar a atribuição

Trabalho 1 - Compilador para LA

- **Se entrada CORRETA, Saída = Código C**
  - Se o arquivo de entrada estiver léxica, sintática e semanticamente correto então o arquivo de saída irá conter o código em C gerado para o arquivo de entrada
- **Se entrada ERRADA, Saída = Relato dos erros**
  - Se forem encontrados erros léxicos, sintáticos ou semânticos então o arquivo de saída irá conter o relato dos erros encontrados no arquivo de entrada
  - NENHUM código C deve ser gerado!

Trabalho 1 - Compilador para LA

Exemplo de programa COM erros

```
algoritmo
declare
   nome: literal
declare
   idade: inteir

{ leitura de nome e idade }
   leia(nome)
   leia(idades)

{ saída da mensagem na tela }
   escreva(nome, " tem ", idade, " anos.")
fim_algoritmo
```

Programa LA errado - entrada.txt

Saída produzida - saida.txt

```
Linha 5: tipo inter nao declarado
Linha 9: identificador idades nao declarado
Fim da compilacao
```

Trabalho 1 - Compilador para LA

Exemplo de programa COM erros

```
algoritmo
declare
   nome: literal
declare
   idade: inteir

{ leitura de nome e idade }
   leia(nome)
   leia(idades)

{ saída da mensagem na tela }
   escreva(nome, " tem ", idade, " anos.")
fim_algoritmo
```

Programa LA errado - entrada.txt

Saída produzida - saida.txt

```
Linha 5: tipo inter nao declarado
Linha 9: identificador idades nao declarado
Fim da compilacao
```

Nesse caso, veja que 'idade' foi inserido na Tabela de Símbolos, mesmo sua declaração estando errada semanticamente, pois ela está correta sintaticamente. Por isso, o compilador não acusa erro que ident não existe como poderia ocorrer na linha 12.

Trabalho 1 - Compilador para LA

Exemplo de programa SEM erros

```
algoritmo
declare
   nome: literal
declare
   idade: inteiro
{ leitura de nome e idade }
   escreva("Nome: ")
   leia(nome)
   escreva("Idade: ")
   leia(idade)
{ saída da mensagem na tela }
   escreva(nome, " tem ",
           idade, " anos.")
fim_algoritmo
```

Programa LA correto - entrada.txt

```
#include<stdio.h>

int main(){
  char nome[80];
  int idade;
  printf("Nome: ");
  gets(nome);
  printf("Idade: ");
  scanf("%d", &idade);
  printf("%s tem %d anos.\n",
          nome, idade);
  return 0;
}
```

Saída em C - saida.txt

Trabalho 1 - Compilador para LA

Exemplo de programa SEM erros

```
algoritmo
declare
   nome: literal
declare
   idade: inteiro
{ leitura de nome }
   escreva("Nome: ")
   leia(nome)
   escreva("Idade: ")
   leia(idade)
{ saída da mensagem }
   escreva(nome,
           idade)
fim_algoritmo
```

Programa LA correto - entrada.txt

```
#include<stdio.h>

int main(){
  char nome[80];
  int idade;
  printf("Nome: ");
  gets(nome);
  printf("Idade: ");
  scanf("%d", &idade);
  printf("%s tem %d anos.\n",
          nome, idade);
  return 0;
}
```

Saída em C - saida.txt

tamanho arbitrário

para literais

para números

Trabalho 1 - Compilador para LA

Comportamento dos comandos leia e escreva

```
algoritmo
declare
   nome: literal
declare
   idade: inteiro
{ leitura de nome e idade }
   escreva("Nome: ")
   leia(nome)
   escreva("Idade: ")
   leia(idade)
{ saída da mensagem na tela }
   escreva(nome, " tem ",
           idade, " anos.")
fim_algoritmo
```

Programa LA correto - entrada.txt

```
Brad Pitt
50
```

Possível entrada na execução

Leia os valores separados por espaço/enter

Trabalho 1 - Compilador para LA

- Utilize **SEMPRE** os casos de teste como referência
- Serão fornecidos todos os casos de teste usados para correção
- Eles servem como base para praticamente todas as dúvidas de como fazer
  - Análise léxica / sintática - mensagens mostradas
  - Análise semântica - como detectar erros
  - Geração de código - como gerar código e como o código gerado deve executar
- Será também fornecido um corretor automático
  - Consulte as instruções sobre como utilizá-lo, no ambiente da disciplina

Observações importantes

- Grupos de **4 (quatro)** alunos
- A nota do trabalho levará em conta boas técnicas e bons costumes de programação
  - Modularização, Documentação interna e externa
- Casos de teste **(saída exata!)**
- Mais detalhes sobre os critérios encontram-se disponíveis no ambiente

Observações importantes

Se cópia **então** **zero para todos** os grupos envolvidos!

Entrega do trabalho

- **Data/horário da tarefa no Moodle**
- Via Moodle - 1 único arquivo compactado com:
  - Código-fonte e executável (quando possível)
  - Arquivo texto LEIA_ME.txt contendo
    - Nomes dos integrantes do grupo
    - Passo-a-passo para **baixar** e **instalar** o gerador
    - Passo-a-passo para **compilar/interpretar** o compilador e **executá-lo**
- Se Moodle fora do ar, então enviar até o prazo final por e-mail ao professor
- Apenas 1 submissão por grupo
- O Moodle NÃO aceitará a submissão de trabalhos após o dia e hora especificados acima!
- O arquivo submetido pode ser substituído até a data de entrega