Construção de Compiladores
Daniel Lucrédio
Especificação e Critérios de notas do Trabalho 5
(Última revisão: dez/2021)

O trabalho 5 (T5) da disciplina consiste em implementar um gerador de código para a linguagem LA (Linguagem Algorítmica) desenvolvida pelo prof. Jander, no âmbito do DC/UFSCar. O gerador de código deverá produzir código executável em C equivalente ao programa de entrada. Exemplo:

Entrada:

```
algoritmo
  declare
    x: literal
  leia(x)
  escreva(x)
fim_algoritmo
```

Saída produzida:

```
#include <stdio.h>
#include <stdlib.h>
int main() {
	char x[80];
	gets(x);
	printf("%s",x);
	return 0;
}
```

IMPORTANTE: o código gerado não precisa ser idêntico ao fornecido nos casos de teste, mas ele deve ser compilável usando GCC, e sua EXECUÇÃO deve ser a mesma que a dos casos de teste. O corretor automático irá compilar o código gerado (usando GCC), executá-lo e comparar a entrada/saída com o que é esperado.

Juntando o analisador semântico com o gerador de código em um único executável

A saída do executável final, que irá combinar analisador léxico+sintático+semântico+gerador, depende da entrada:

- Se a entrada tiver algum erro (léxico ou sintático ou semântico) a saída deve conter a descrição dos erros;
- Se a entrada não tiver nenhum erro, a saída deve conter o código gerado.

O compilador deve poder ser executado em linha de comando (windows, mac ou linux), com DOIS ARGUMENTOS OBRIGATORIAMENTE:

Argumento 1: arquivo de entrada (caminho completo)
Argumento 2: arquivo de saída (caminho completo)

Exemplo de como seu compilador deve rodar:

```
c:\java -jar c:\compilador\meu-compilador.jar c:\casos-de-teste\arquivo1.txt c:\temp\saida.txt
```

Como resultado, seu compilador deve ler a entrada de `c:\casos-de-teste\arquivo1.txt` e salvar a saída no arquivo `c:\temp\saida.txt`

NÃO SERÃO ACEITOS programas que imprimem a saída no terminal. É obrigatório salvar no arquivo.

Critérios de avaliação do Trabalho 5 e descontos nas notas

O trabalho 5 deve ser desenvolvido em **grupos de até 3 estudantes** (máximo). A nota do trabalho 5 será composta de 3 parcelas, cada uma valendo de 0 a 10, e com os pesos assim distribuídos:

DE - Documentação externa: 10%
DI - Documentação interna: 10%
CT5 - Casos de teste sem erros (geração de código): 80%

DE - Documentação externa: deve ser fornecido um arquivo de ajuda, para possibilitar que qualquer pessoa consiga compilar e executar seu trabalho. Deve incluir programas que precisam ser instalados, suas respectivas versões, configurações necessárias, e os passos de execução.

Exemplos de erros comuns na documentação externa e que causarão desconto na nota:

- Só diz como executar, mas não como compilar o programa, ou vice-versa
- Nada foi dito sobre como compilar/interpretar o programa e executá-lo
- Ausência da documentação externa

DI - Documentação interna: o código-fonte (gramática + demais arquivos) devem ser documentados a ponto de possibilitar seu entendimento por parte de outros programadores olhando seu código. Insira comentários explicativos em todos os pontos relevantes do seu código. Nomes de variáveis e funções também fazem parte da documentação interna.

Exemplos de erros comuns na documentação interna e que causarão desconto na nota:

- Pouco documentado (sem explicação do propósito das regras léxicas, sintáticas, semânticas e geração de código, entrada, saída, descrição das variáveis, etc.)
- Descuido nos comentários ou nomes de funções/variáveis pouco indicativos
- Ausência de comentários sobre o processo de compilação
- Nenhuma linha de documentação relevante

CT5 - Casos de teste sem erros (geração de código): o compilador deve produzir código C compilável. O código gerado não precisa ser idêntico ao fornecido nos casos de teste, mas sua execução (mapeamento entrada-saída) deve ser idêntica à fornecida nos casos de teste. São **20 casos de teste** nesta categoria.