CONSTRUÇÃO DE COMPILADORES I [BCC328]

Primeira prova (2017–2)
08 de novembro de 2017

Matrícula: _____

Nome: _____

Departamento de Computação
Universidade Federal de Ouro Preto
Prof. José Romildo Malaquias

Questão 1. Diferentes linguagens de programação usam diferentes notações para representar literais numéricos. Construa uma expressão regular para especificar os literais inteiros não negativos na base 16 da linguagem C. são Eles formados pelo prefixo `0x` ou `0X`, seguidos de uma sequência não vazia de dígitos hexadecimais, opcionalmente seguidos por um sufixo que é uma combinação de `U` e `L` em qualquer ordem, maiúsculo ou minúsculo, designando respectivamente inteiro sem sinal e inteiro longo.

Seguem alguns exemplos:

```
0x2378
0X19ABC
0xA00B5l
0x10bb7aU
0x95f04bcaLU
```

Questão 2. Considere a seguinte especificação léxica:

```
(00)*   { TokA }
01+     { TokB }
10+     { TokC }
```

Nesta especificação cada regra é formada por uma expressão regular e a classificação do símbolo léxico correspondente.

1. Construa **um** autômato finito estendido para classificar os símbolos léxicos de acordo com esta especificação léxica. O autômato é estendido porque cada estado final tem uma indicação da classificação do símbolo léxico correspondente. O autômato pode ser não determinístico.

2. Faça a análise léxica das cadeias seguintes usando esta especificação léxica. Indique os símbolos léxicos (palavras) encontrados e a classificação obtida para os mesmos.

   a) `01100110`

   b) `0001101`

   c) `0111110`

   d) `01100100`

Questão 3. A gramática livre de contexto a seguir define expressões regulares usando os símbolos *a* e *b*:

$$
\begin{array}{lcl}
E & \rightarrow & E \ \texttt{|} \ E \\
E & \rightarrow & E \ E \\
E & \rightarrow & E \ \texttt{*} \\
E & \rightarrow & \texttt{a} \\
E & \rightarrow & \texttt{b}
\end{array}
$$

1. Dê evidências de que a gramática é ambígua.

2. Escreva uma gramática livre de contexto não ambígua equivalente à gramática dada. Considere que a ordem crescente de precedência das operações é: *união*, *concatenação* e *repetição*, e que a *união* é associativa à esquerda, e a *concatenação* é associativa à direita.

Questão 4. Considere a seguinte gramática livre de contexto:

$$
\begin{aligned}
S &\rightarrow E \texttt{ ;} \\
S &\rightarrow \texttt{id := } E \texttt{ ;} \\
S &\rightarrow \texttt{if (} E \texttt{) } S \texttt{ else } S \\
S &\rightarrow \texttt{while (} E \texttt{) } S \\
S &\rightarrow \texttt{\{ } L \texttt{ \}}
\end{aligned}
$$

$$
\begin{aligned}
L &\rightarrow \\
L &\rightarrow S \ L
\end{aligned}
$$

$$
\begin{aligned}
E &\rightarrow A \texttt{ = } A \\
E &\rightarrow A
\end{aligned}
$$

$$
\begin{aligned}
A &\rightarrow T \texttt{ + } A \\
A &\rightarrow T
\end{aligned}
$$

$$
\begin{aligned}
T &\rightarrow F \texttt{ * } T \\
T &\rightarrow F
\end{aligned}
$$

$$
\begin{aligned}
F &\rightarrow \texttt{id} \\
F &\rightarrow \texttt{(} E \texttt{)}
\end{aligned}
$$

a) Estenda a gramática com uma nova regra de produção para um novo símbolo inicial *S'* que introduz o terminal `$` que representa o fim da entrada. Utilze a gramática estendida para resolver os próximos itens.

b) Determine quais símbolos não terminais são anuláveis.

c) Determine o conjunto *FIRST* de cada símbolo não terminal.

d) Determine o conjunto *FOLLOW* de cada símbolo não terminal.

NT	NULLABLE	FIRST	FOLLOW
S'			
S			
L			
E			
A			
T			
F			

Questão 5. Considere a seguinte gramática livre de contexto:

0 $S \rightarrow E$ `$`

1 $E \rightarrow$ `-` E
2 $E \rightarrow$ `(` E `)`
3 $E \rightarrow V\ E'$

4 $E' \rightarrow$ `-` E
5 $E' \rightarrow$

6 $V \rightarrow$ `id` V'

7 $V' \rightarrow$ `(` E `)`
8 $V' \rightarrow$

Os símbolos não terminais anuláveis, bem como os conjuntos *FIRST* e *FOLLOW* estão indicados na tabela a seguir.

NT	*NULLABLE*	*FIRST*	*FOLLOW*
S	F	`-` `(` `id`	
E	F	`-` `(` `id`	`)` `$`
E'	V	`-`	`)` `$`
V	F	`id`	`-` `)` `$`
V'	V	`(`	`-` `)` `$`

Construa a tabela de análise sintática LL(1) para a gramática. Esta gramática é LL(1)? Justifique sua resposta.
Observação: utilize os números das regras de produção para preencher a tabela.

NT	`id`	`-`	`(`	`)`	`$`
S					
E					
E'					
V					
V'					