اصول طراحی کامپایلر
PRINCIPLES OF COMPILER DESIGN

تمرین‌های مرتبط با درس‌نامه ۴
گرامرهای مستقل از متن

◊ تمرین‌ها

۱. گرامر مستقل از متن زیر را در نظر بگیرید: [2.1]

$$S \rightarrow SS+ \mid SS* \mid a$$

آ) نشان دهید که رشتهٔ $aa+a*$ را می‌توان توسط این گرامر تولید نمود.
ب) درخت تجزیه (parse tree) و درخت نحو (syntax tree) را برای این رشته بسازید.
پ) این گرامر چه زبانی را تولید می‌کند؟

۲. کدام زبانها توسط گرامرهای زیر تولید می‌شوند؟ [2.2]

آ) $S \rightarrow 0S1 \mid 01$
ب) $S \rightarrow +SS \mid -SS \mid a$
پ) $S \rightarrow S(S)S \mid \varepsilon$
ت) $S \rightarrow aSbS \mid bSaS \mid \varepsilon$
ث) $S \rightarrow a \mid S+S \mid SS \mid S* \mid (S)$

۳. کدام یک از گرامرهای تمرین ۲ مبهم است؟ [2.3]

۴. برای هر یک از زبانهای زیر، یک گرامر مستقل از متن بسازید. [2.4]
آ) عبارات ریاضی به شکل پسوندی
ب) لیستهای "شرکت پذیر از چپ" (left associative) از شناسه‌ها که با کاما جدا شده‌اند.
پ) لیستهای "شرکت پذیر از راست" (right associative) از شناسه‌ها که با کاما جدا شده‌اند.
ت) عبارات ریاضی از اعداد و شناسه‌ها با چهار عملگر دو عملوندی $+ - * /$
ث) به عملگرهای ریاضی (ت) جمع و تفریق تک عملوندی (unary) اضافه نمایید.

۵. یک گرامر مستقل از متن برای اعداد رومی بنویسید. [2.6]

۹. گرامر زیر را برای دستورات **if-then** و **if-then-else** در نظر بگیرید:

$$\begin{aligned} stmt \rightarrow\ & \textbf{if}\ expr\ \textbf{then}\ stmt \\ \mid\ & \textbf{if}\ expr\ \textbf{then}\ stmt\ \textbf{else}\ stmt \\ \mid\ & \textbf{other} \end{aligned}$$

که در آن **other** مشخص‌کنندهٔ دستورات دیگر در زبان است.
آ) نشان دهید که این گرامر مبهم است.
ب) گرامر غیرمبهم متناظر با آن را بسازید که در آن هر **else** با نزدیکترین **then** انطباق‌نیافتهٔ قبلی متناظر است.
پ) یک رویهٔ ترجمهٔ نحوی بر مبنای این گرامر برای ترجمهٔ عبارات شرطی به کد ماشین پشته‌ای بنویسید. [2.16]

۱)

$S \to SS+$ (1)

$S \to SS*$ (2)

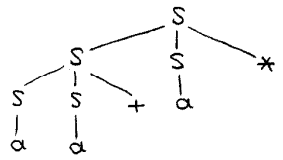
$S \to a$ (3)

$G = \langle \{S\}, \{a, +, *\}, S, P \rangle$

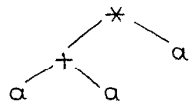
آ) $x = aa+a*$

$\underline{S} \overset{(2)}{\Rightarrow} \underline{S}S* \overset{(1)}{\Rightarrow} \underline{S}S+S* \overset{(3)}{\Rightarrow} a\underline{S}+S* \overset{(3)}{\Rightarrow} aa+\underline{S}* \overset{(3)}{\Rightarrow} aa+a*$

ب) parse tree :



syntax tree :



پ) $L(G) = \{ w : w \text{ is a postfix expression using } a \text{ as operand and } +, * \text{ as operators} \}$

۲)

$G = \langle \{S\}, T, S, P \rangle$ $\quad L(G) = ?$

آ) $S \to 0S1$ (1)

$S \to 01$ (2)

$\underline{S} \overset{(1)}{\Rightarrow} 0\underline{S}1 \overset{(1)}{\Rightarrow} 00\underline{S}11 \overset{(1)}{\Longrightarrow}{}^{n-3} a^{n-1}\underline{S}b^{n-1} \overset{(2)}{\Rightarrow} a^n b^n \quad , n \geq 1$

$L(G) = \{ a^n b^n : n \geq 1 \}$

ب) $S \to +SS$ (1)

$S \to -SS$ (2)

$S \to a$ (3)

$L(G) = \{ w : w \text{ is a prefix expression using operand } a \text{ and } +, - \}$

پ) $S \to S(S)S \mid \varepsilon$ $\quad L(G) = \{ w \in \{(,)\}^* : w \text{ is a matched and balanced brackets} \}$

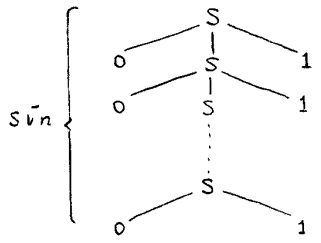
ت) $S \to aSbS \mid bSaS \mid \epsilon$

$L(G) = \{ w \in \{a,b\}^* : n_a(w) = n_b(w) \}$

ث) $S \to a \mid S+S \mid SS \mid S* \mid (S)$

$L(G) = \{ w : w \text{ is a regular expression using operand } a \}$

۳) آ) صحیح نیست. برای $x = 0^n 1^n$ $\quad n \geq 1$ تنها یک درخت اشتقاق ممکن است.



ب) مبهم نیست : زیرا هر عبارت پسوندی را می‌توان بدون نیاز به وارد کردن پرانتزها بدون ابهام تفسیر کرد.

پ) مبهم است. رشتهٔ $x = ( )( )$ دارای دو اشتقاق چپ است.

I) $S \Rightarrow \underline{S}(S)S \overset{(2)}{\Rightarrow} (\underline{S})S \Rightarrow ( )\underline{S} \Rightarrow ( )S(S)S \overset{(2)}{\Rightarrow}{}^{+} ( )( )$

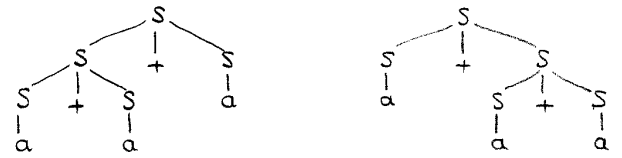
II) $S \Rightarrow \underline{S}(S)S \overset{(1)}{\Rightarrow} S(S)S(S)S \Rightarrow^{+} ( )( )$

ت) مبهم است. رشتهٔ $x = abab$ دارای دو اشتقاق چپ است.

I) $\underline{S} \Rightarrow_L a\underline{S}bS \Rightarrow_L ab\underline{S} \Rightarrow_L abaSbS \Rightarrow_L^{+} abab$

II) $\underline{S} \Rightarrow_L aaSbSbS \Rightarrow_L^{+} abab$

ث) مبهم است. رشتهٔ $x = a+a+a$ دارای دو درخت اشتقاق متفاوت است.



گراف چپ گرامر :

برای گرامر $G = \langle N, T, S, P \rangle$ ، گراف چپ آن را به صورت زیر تعریف می‌کنیم :

$g_G = (Nodes, Edges)$

$Nodes = \{\alpha : S \overset{+}{\underset{G_L}{\Rightarrow}} \alpha \ , \ \alpha \in (N \cup T)^*\}$ گره‌های گراف فرم‌های جمله‌ای گرامر G است.

$Edges = \{\langle \alpha, r, \beta \rangle : \alpha \overset{(r)}{\underset{G_L}{\Longrightarrow}} \beta \ , \ \alpha, \beta \in Nodes\}$ یال‌های گراف بیانگر یک اشتقاق چپ است.

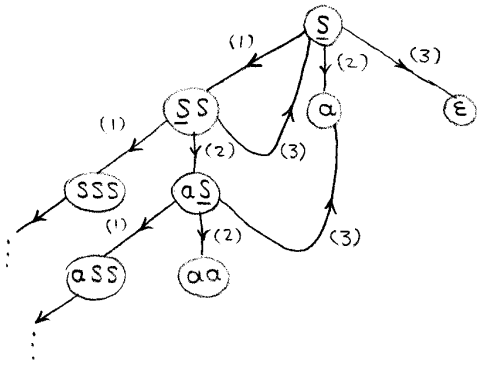
قضیه : فرض می‌کنیم که G گرامری باشد که تمام ناپایانه‌های آن فعال باشد (یعنی $\forall A \in N \ \exists x \in T^* \ A \Rightarrow^{*} x$) در این صورت G یک گرامر مبهم است، اگر و فقط اگر گراف چپ آن دارای چرخه (cycle) باشد.

مثال :

$S \longrightarrow SS$ (1)

$S \longrightarrow a$ (2)

$S \longrightarrow \varepsilon$ (3)



۴)

آ) عبارات ریاضی به شکل پسوندی postfix

فرض می‌کنیم عملوندها اعداد صحیح باشند.

S → SS+ | SS- | SS* | SS/ | INT | ID

INT → Ø | SIGN NON_ZERO_DIGIT INT1

INT1 → INT1 DIGIT | ε

SIGN → + | - | ε

NON_ZERO_DIGIT → 1|2|3|4|5|6|7|8|9

DIGIT → Ø | NON_ZERO_DIGIT

ID → ID LETTER | ID DIGIT | LETTER

LETTER → a|b|c|...|z

ب) لیست‌های شرکت پذیر از چپ

LIST → LIST , ID | ID

پ) لیست‌های شرکت پذیر از راست

LIST → ID , LIST | ID

ت) عبارات ریاضی از اعداد و شناسه‌ها با عملگرهای باینری + - * /

EXPR → EXPR + TERM | EXPR - TERM | TERM

TERM → TERM * FACTOR | TERM / FACTOR | FACTOR

FACTOR → INT | ID | (EXPR)

ث) جمع و تفریق unary

FACTOR → - FACTOR | + FACTOR

۵) گرامر مستقل از متن برای اعداد رومی roman numerals

ROMAN_NUMERAL → THOUSANDS HUNDREDS TENS ONES

ONES → LOWONES | IV | V LOWONES | IX

LOWONES → ε | I | II | III

TENS → LOWTENS | XL | L LOWTENS | XC

LOWTENS → ε | X | XX | XXX

HUNDREDS → LOWHUNDREDS | CD | D LOWHUNDREDS | CM

LOWHUNDREDS → ε | C | CC | CCC

THOUSANDS → M THOUSANDS | ε

1	2	3	4	5	6	7	8	9
I	II	III	IV	V	VI	VII	VIII	IX
10	20	30	40	50	60	70	80	90
X	XX	XXX	XL	L	LX	LXX	LXXX	XC
100	200	300	400	500	600	700	800	900
C	CC	CCC	CCC	D	DC	DCC	DCCC	CM

1000	2000	3000	4000	5000	...	m x 1000
M	MM	MMM	MMMM	MMMMM		MMM...M (m's)

۹)

```
stmt → if expr then stmt
     | if expr then stmt else stmt
     | other
```

آ) برای اینکه نشان دهیم این گرامر مبهم است، دو اشتقاق چپ برای رشتهٔ زیر می‌نویسیم که توسط این گرامر تولید می‌شود :

w = if expr then if expr then other else other

I) $stmt \Rightarrow_{lm}$ if expr then stmt $\Rightarrow_{lm}$ if expr then if expr then stmt else stmt $\Rightarrow_{lm}^{*}$ w

II) $stmt \Rightarrow_{lm}$ if expr then stmt else stmt $\Rightarrow_{lm}$ if expr then if expr then stmt else stmt $\Rightarrow_{lm}^{*}$ w

ب) گرامر غیر مبهم معادل : (هر else شاظر با نزدیکترین then تطبیق نیافته)

```
stmt → matched stmt                                              (1)
     | unmatchedstmt                                             (2)
matchedstmt → if expr then matchedstmt1 else matchedstmt2        (3)
            | other                                              (4)
unmatchedstmt → if expr then stmt                                (5)
              | if expr then matchedstmt1 else unmatched stmt2   (6)
```