اصول طراحی کامپایلر - تمرینات سری سوم - خلیلیان

۱. گرامر عبارات را در نظر بگیرید. مراحل اشتقاق از سمت چپ را برای عبارت (a*b-c)/(d-e) را مشخص نمایید.

E→E+T | E-T | E OR T | T
T→T*F | T/F | T AND F | F
F→ID | NO | (E) | NOT F

۲. گرامر زیر را در نظر بگیرید. نشان دهید که این گرامر مبهم است. این گرامر تمام عبارات منظم روی a و b را تولید می‌کند.

R→R 'ا' R | RR | R^* | (R) | a | b

۳. گرامر ساختار لیست را در نظر بگیرید. درخت تجزیه را برای جملات زیر ترسیم نمایید.

S→'('L')' | a
L→L','S | S

الف) (a,a)

ب) (a,(a,a))

ج) (a,((a,a),(a,a)))

۴. نشان دهید که گرامرهای زیر مبهم است. (جمله‌ای پیدا کنید که برای آن دو درخت اشتقاق متمایز وجود داشته باشد و آنها را رسم کنید)

S→aSbS | bSaS | ε

S→AB | aaB
A→a | Aa
B→b

۵. گرامر زیر را در نظر بگیرید.

bexpr→bexpr or bterm | bterm
bterm→bterm and bfactor | bfactor
bfactor→not bfactor | (bexpr) | true | false

الف) برای عبارت not (true or false) درخت تجزیه ایجاد کنید. ب) آیا این گرامر مبهم است؟

۶. برای هر یک از زبان‌های زیر یک گرامر طراحی کنید. کدام‌یک از این زبان‌ها منظم هستند؟

الف) مجموعه تمام رشته‌هایی از ۰ و ۱ ها که بعد از هر صفر دست‌کم یک ۱ قرار داشته باشد.

ب) رشته‌هایی از ۰ و ۱ ها که در آنها تعداد ۰ و ۱ ها برابر باشد. ج) رشته‌هایی از ۰ و ۱ ها که در آنها تعداد ۰ و ۱ ها برابر نباشد.

۷. در یک زبان بدون محدودیت که الفبای آن از دو حرف a و b تشکیل شده است، چند رشته با طول ۶ وجود دارد که تعداد a و b های آنها مساوی باشند؟

۸. برای ساده سازی گرامرهای نوع دوم، باید به‌ترتیب قوائد ε، سپس قوائد یکه (منفرد) و نهایتاً قوائد بی‌فایده را از گرامر حذف کرد. یک گرامر فاقد ε است اگر قانون تولید ε نداشته باشد یا تنها یک قانون تولید ε به‌صورت S→ε داشته باشد و نماد شروع S در سمت راست هیچ قانون تولیدی ظاهر نشود. برای حذف قوائد ε و به‌دست آوردن گرامر فاقد ε، باید ابتدا متغیرهای (ناپایانه‌ها) میرا را به‌دست آوریم و سپس قوائد ε را طبق الگوریتمی حذف کنیم. یک قانون تولید منفرد است اگر سمت راست آن دارای یک غیرپایانی منفرد باشد. یک قائده بی‌فایده است اگر در آن حداقل یک ناپایانه بدون استفاده ظاهر شود. یک ناپایانه بدون استفاده است اگر یکی یا هر دو شرط زیر را نداشته باشد.

الف) اگر از آن شروع کنیم بتوانیم به یک رشته نهایی برسیم: $A\rightarrow^* w,\ w\epsilon T^*$

ب) $S\rightarrow^* \alpha A\beta \qquad \alpha,\beta\in(V\cup T)^*$

اولاً: تمام قوانین ε و بی‌فایده را از گرامر زیر حذف کنید.

S→AaB | aaB
A→ε
B→bbA | ε

ثانیاً: تمام قوانین منفرد را از گرامر زیر حذف کنید.

S→aA | a | B | C
A→aB | ε
B→Aa
C→cCD
D→ddd

۹. (پروژه برنامه نویسی) فصل دوم کتاب کامپایلر، چند روش دستورگرا برای ساختن بخش جلوبندی کامپایلر ارائه کرده است. در انتهای این فصل، برنامه‌ای به زبان سی نوشته شده است که به‌صورت مترجم یک عبارت میانوندی به پسوندی عمل می‌کند. این برنامه پارسر را با روش تجزیه بازگشتی نزولی پیاده‌سازی کرده است.

- این برنامه را در محیط کامپایلر سی تایپ کرده و آنرا روی چند نمونه ورودی اجرا کنید.
- خروجی اجرای برنامه روی چند ورودی نوشته و تفسیر عملکرد هر یک از بخش‌های برنامه مترجم را با جزییات شرح دهید.
- برنامه را طوری توسعه دهید که عبارت پسوندی نهایی را نیز ارزیابی کرده و مقدار آنرا محاسبه کند.

۱۰. (پروژه برنامه نویسی) برنامه‌ای مشابه تمرین ۸ بنویسید که با یک روش ترجمه دستورگرا، اعداد صحیح را به معادل رومی تبدیل کند.

۱۱. گرامر مستقل از متن زیر را در نظر بگیرید و مراحل اشتقاق از سمت چپ را برای رشته aa+a* نشان دهید. سپس درخت تجزیه را برای ان رسم کنید.

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

۱۲. گرامرهای زیر چه زبان‌هایی تولید می‌کنند؟ کدام‌یک مبهم هستند؟

$S \rightarrow 0S1 \mid 01$
$S \rightarrow +SS \mid -SS \mid a$
$S \rightarrow S(S)S \mid \varepsilon$
$S \rightarrow a \mid S+S \mid SS \mid S* \mid (S)$

۱۳. در تمام زبان‌های برنامه نویسی با دستور شرطی به شکل else سرگردان، از قانون کلی زیر استفاده می‌شود:
هر else باید با نزدیکترین then جفت نشده قبلی جور شود. این قانون رفع ابهام را می‌توان مستقیماً در گرامر قرار داد. بر این اساس، گرامر غیر مبهم برای گرامر مبهم دستورات شرطی زیر را بنویسید.

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