CS 536

Practice Midterm Exam

Fall 2018

1. (a) Is the set of strings that contain no duplicate characters regular? Why?

   (b) Write a regular expression for comments that begin with << and end with >>. The body of the comment may contain any character sequence except >> (so that >> always marks the end of the comment).

2. Let $S = \{ [^i \;]^j \mid i \neq j \}$. S is the set of all *unbalanced* brackets; that is, a number of left brackets followed by a *different* number of right brackets.

   Is S a regular set? If it is, give a regular expression or finite automaton that defines it. If S isn't a regular set, explain carefully why.

3. Give JLex regular expression definitions that match the following strings

   (a) The four characters: `"\n"`

   (b) Any odd number of backslash characters (e.g., \ or \\\ or \\\\\, etc.).

   (c) A CSX multi-line comment, delimited by `{` and `}`, that is allowed to contain *no more* than two new-line characters. That is, the comment may appear entirely on one line, or it may span two or three lines, but no more than three lines.

4. Below is a context-free grammar for a language of assignments that includes arrays:

1. stmtList → stmt stmtList
2. | λ

3. stmt → ID = exp ;

4. array → [rowList]

5. rowList → nonEmpty
6. | λ

7. nonEmpty → row moreRows

8. moreRows → ; nonEmpty
9. | λ

10. row → exp more

11. more → , row
12. | λ

13. exp → term tail

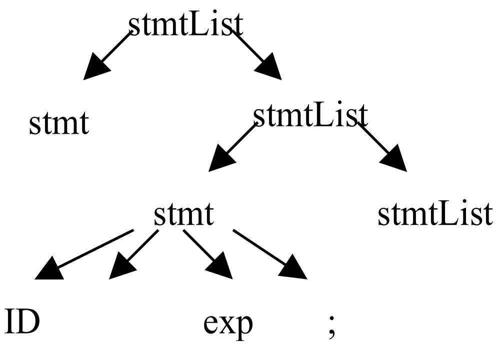
14. tail → + term tail
15. | λ

16. term → ID
17. | INTLIT
18. | array

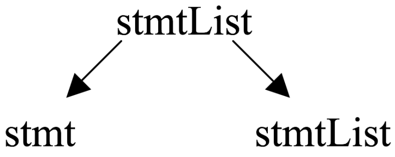
Here are the *FIRST* and *FOLLOW* sets for all of the non-terminals:

Non-terminal *X*	***FIRST*(*X*)**	***FOLLOW*(*X*)**
stmtList	ID	EOF
stmt	ID	ID EOF
array	[	+ , ;]
rowList	ID INTLIT [	]
nonEmpty	ID INTLIT [	]
moreRows	;	]
row	ID INTLIT [	;]
more	,	;]
exp	ID INTLIT [	, ;]
tail	+	, ;]
term	ID INTLIT [	+ , ;]

(a) Recall that terminal *t* is in *FOLLOW*(*X*) if in some partial parse tree with the start non-terminal at the root, *X* is one leaf of the tree and *t* is the next non-lambda leaf immediately to the right. For example, the following partial parse tree justifies the fact that for the CFG given above, terminal ID is in *FOLLOW*(stmt):



Complete the partial parse tree below to justify the fact that terminal ; is in *FOLLOW*(term).



(b) Fill in the parse table below using the numbers of the grammar rules rather than the rules themselves. Is the grammar LL(1)?

	ID	INTLIT	=	+	;	,	[	]	EOF
stmtList									
stmt									
array									
rowList									
nonEmpty									
moreRows									
row									
more									
exp									
tail									
term									

5. Consider the following grammar

$$\begin{array}{rcl}
File & \rightarrow & Record \\
 & | & Record\ File \\
Record & \rightarrow & \textbf{name idnum}\ OptGrades \\
OptGrades & \rightarrow & Grades \\
 & | & \lambda \\
Grades & \rightarrow & OneGrade \\
 & | & OneGrade\ \textbf{comma}\ Grades \\
OneGrade & \rightarrow & \textbf{intlit}\ OptLate \\
OptLate & \rightarrow & Stars \\
 & | & \lambda \\
Stars & \rightarrow & \textbf{star} \\
 & | & Stars\ \textbf{star}
\end{array}$$

where *File* is the start non-terminal, and symbols in **bold** are terminals.

(a) Apply the transformations learned in class to *left factor* the grammar above and write the results below. Give the entire grammar, not the just the transformed rules.

(b) If the grammar you wrote above has any immediate left recursion, apply the transformation learned in class to remove it and write the result below. You do not need to give the entire grammar; you can just give the transformed rules.