Dr. B R Ambedkar National Institute of Technology Jalandhar, Punjab, India

System Programming and Compiler Design (CSPC-401)

Assignment

- **Question 01:** Find first and follow for the given grammar:-
  E → PQa / bRP
  P → cQRS / ∈
  Q → RdP / ad
  R → eR / ∈
  S → bsf / a

- **Question 02:** A student wrote two context-free grammars G1 and G2 for generating a single C-like array declaration. For example: " **float f [10] [3];** "
  In the grammars D use as a start symbol and six different terminal symbols {float, ;, id, [,], num } use as a non-terminal symbol.

Grammar G1	**Grammar G2**
D → int L;	**D → int L;**
L → id [E	**L → id E**
E → num]	**E → E [num]**
E → num] [E	**E → [num]**

  Create parse tree using each grammar to generate the declaration statement as mentioned above.

- **Question 03:** Consider the following grammar G_1 and G_2, eliminate left recursion from G_1 and eliminate left-factoring from grammar G_2.

Grammar G_1	**Grammar** G_2
A → Ba / Aa / c	**S → a / ab / abc / abcd**
B → Bb / Ab / d	

- **Question 04:** Write an algorithm for Recursive Decent Parser. Explain with a suitable example.

- **Question 05:** Consider the grammar with non-terminals N={S,C,S_1}, terminals T={a,b,i,t,e}, where S is the start symbol, and the set of rules are as follows:

  $$S \rightarrow iCtSS_1 \;/\; a$$
  $$S_1 \rightarrow eS \;/\; \in$$
  $$C \rightarrow b$$

  Construct the LL(1) parsing table and check whether given grammar is LL(1) or not?

- **Question 06:** Consider the grammar with non-terminals N={S,A}, terminals T={a,b}, with S as the start symbol, and the following set of rules:
  S → AA
  A → aA / b
  Construct the LR (0) parsing table and check whether given grammar is LR (0) or not? Also implement the stack for the input string: **aabb**

- **Question 07:** Consider the grammar with non-terminals N={E,T,F}, terminals T={ (,) , +,*, id}, with E as the start symbol, and the following set of rules:
  E → E+T / T
  T → T*F / F
  F → (E) / id
  Construct the SLR(1) parsing table and check whether given grammar is SLR(1) or not? Also generate the input string "**id*id+id**"using stack implementation.

- **Question 08:** Consider the grammar with non-terminals N=S,A, terminals T=a,b, with S as the start symbol, and the following set of rules:
  S → AA
  A → aA / b
  Construct the CLR parsing table and check whether given grammar is CLR or LALR or both?

- **Question 09:** Consider the following SDT,
  S → M {PRINT "2";} A
  M → 1 {PRINT " "; }
  A → D {PRINT "1";} E
  D → 2 {PRINT " "; }
  E → E {PRINT " "; } A
  E → 3 {PRINT " "; }
  A → S {PRINT "4";} Y
  S → 4 {PRINT " "; }
  Y → ∈ {PRINT " ";}
  If the bottom up parsing is used to parse the input string "**1234**" then find the output number produced (without any spaces).

- **Question 10:** What is Intermediate Code Generation? Write the three-address code for the following conditional statement:
  x = (a + b * c) / (a – b * c)
  Also draw the Abstract Syntax Tree (AST) for the above statement.