15.5.14

BE 6th semester Examination 2014
Subject : Compiler Design Code: CS 603
Full Marks : 70 Time: 3 hours

Answer question no. 1 and any four from the remaining questions.

1.(a) Justify the statement:
' Interpreter is more suitable than compiler in development environment whereas the reverse is true in production environment.' [2]
(b) Define leftmost canonical derivation and rightmost canonical derivation. Show with an example that top-down parsing method is associated with leftmost canonical derivations whereas bottom-up parsing method is associated with reverse of rightmost canonical derivations. [2+3]
(c) Distinguish among SLR(1), LR(1) and LALR(1) parsers with reference to number of states, economy and efficiency. [3]
(d) Why is design of a multi-pass compiler needed instead of a single-pass one? [2]
(e) Justify the presence of a look-ahead buffer within a lexical analyzer. [2]

2.(a) What are the basic features of assembly language statement? What are the advantages of assembly language compared to the machine language? Describe the method for designing a single-pass assembler using a suitable data structure. [2+2+4]
(b) What is a MACRO? What are the data structures needed for designing a macro preprocessor. Draw the flowchart for a macro pre-processor. [1+1+4]

3. (a) Differentiate between static storage allocation and dynamic storage allocation. What are the criteria that need to be satisfied for static storage allocation? Define activation record. What are the roles of activation base pointer and display in activation record? How the display will be constructed when a block at level *j* is entered from a block at level i? [2+2+1+2+3]
(b) Define basic block. Explain with example how DAG can be used for variable propagation and common sub-expression elimination. [1+ 3]

4.Given the grammar with the set of terminal symbols {#,=,+,*}, set of non-terminal symbols {S,G,E,T} and the start symbol S, where production rules are as follows:

S→G#
G→E=E| f
E→T| E+T
T→f | T*f

(a)Compute FIRST and FOLLOW for all non-terminal symbols excluding S of the grammar. [3]
(b)Construct LR(0) machine for this grammar. [5]
(c)Mark all ambiguous states of the grammar. Show their item sets and describe which type of conflict is associated with each of those states. [4]
(d)Is the grammar SLR(1)? [2]

5.Define LL(1) grammar.
For the given grammar with set of terminal symbols {a,b,c}, set of non-terminal symbols {S, A, B, C, D } and start symbol S, find FIRST and FOLLOW for all non-terminal symbols, construct the parsing table and tell whether the following grammar is LL(1) or not.

S→ aABC
A→a
A→ bbD
B→a
B→ Ɛ
C→b
C→ Ɛ
D→c
D→ Ɛ [2+5+6+1]

6.(a) Consider the following code fragment. Generate the three-address code for it.

if a < b then while c>d
{c = c + b}
else c = c + d ; [2]

(b) Write the syntax directed translation schemes for the grammar rules involving boolean expressions, if else and while statement and generate annotated parse tree for the code fragment. [5 + 2 + 2 + 3]

7.(a) Define the following operator precedence relations [3]
< ,> , =
(b) Define operator precedence grammar. [1]
(c) Compute Leading and Trailing of the non-terminal symbols for the grammar consisting of the following rules with the set of terminal symbols {+, -, *, /}
E→T | E+T | E – T
T→ F |T*F | T/F [4]
F→ P |F | P
P→ i | (E)
(d) Derive the operator precedence relations existing between pair of terminal symbols. [6]