USN | | | | | | | | | | |

06IS662

Sixth Semester B.E. Degree Examination, June/July 2013
Compiler Design

Time: 3 hrs. Max. Marks:100

Note: ***Answer FIVE full questions, selecting at least TWO questions from each part.***

PART – A

1 a. Explain in detail, the different phases of a compiler with a neat block diagram. **(10 Marks)**

b. Discuss the different compiler construction tools. **(05 Marks)**

c. Define Tokens, patterns and lexeme with examples. **(05 Marks)**

2 a. Consider the grammar

E → E + T/T

T → id/id[ ]/id[x]

X → E, E/E

i) Eliminate left recursion.

ii) Perform left factoring for the solution obtained in (i). **(06 Marks)**

b. Obtain the FIRST and follow for the grammar

Exp → Expaddopterm/term

addop → +/-

term → term mulop factor/factor

mulop → *

factor → (Exp)

factor → number. **(06 Marks)**

c. For the following grammar

E → TE′

E′ → + TE′/∈

T → FT′

T′ → *FT′/∈

F → (E)/id

i) Construct the LL(1) predictive parsing table.

ii) Show the moves of the parser for the input string id + id * id. **(08 Marks)**

3 a. What is handle and handle pruning? For the grammar

E → E + T/T

T → T * F/F

F → (E)/id

Indicate the handle for the following right sentential form $id_1 * id_2$. **(04 Marks)**

b. Consider the grammar

E → E + E

E → E * E

E → (E)

E → id

Perform shift reduce parsing for the string $id_1 * id_2 * id_3$. **(06 Marks)**

c. Consider the grammar E → E + n/n

i) Find LR(0) items

ii) Construct SLR parsing table and parse the input string n + n + n. **(10 Marks)**

4 a. Consider the grammar
$S \rightarrow CC$
$C \rightarrow aC$
$C \rightarrow d$
Construct set of LR(1) items and also obtain the parsing table using LALR technique. **(10 Marks)**

b. Consider the dangling else ambigious grammar $S \rightarrow iSeS/iS/a$
i) Obtain LR(0) items and construct SLR parsing table.
ii) Resolve the conflict obtained in solution of (i) for the input iiaea and construct the new parsing table. **(10 Marks)**

PART – B

5 a. Construct the grammar that is used for simple desk calculator:
$L \rightarrow En$
$E \rightarrow E_1 + T$
$E \rightarrow T$
$T \rightarrow T_1 * F$
$T \rightarrow F$
$F \rightarrow (E)$
$F \rightarrow$ digit
i) Obtain the semantic rules for the above grammar.
ii) Obtain the annotated parse tree for the string $(3 + 4) * (5 + 6)$ n. **(10 Marks)**

b. Obtain the syntax-directed definition for simple type declarations
$D \rightarrow TL$
$T \rightarrow$ int
$T \rightarrow$ float
$L \rightarrow L_1$, id
$L \rightarrow$ id
Also obtain the dependency graph for a declaration float id_1, id_2, id_3. **(10 Marks)**

6 a. What are the benefits of intermediate code generation? **(04 Marks)**

b. Explain quadruples, triples and indirect triples, also obtain for the expression
a: = b* - c + b * c. **(10 Marks)**

c. For the following two type expressions, obtain the equivalence classes using unification:
$((\alpha_1 \rightarrow \alpha_2) \times \text{list}(\alpha_3)) \rightarrow \text{list}(\alpha_2)$
$((\alpha_3 \rightarrow \alpha_4) \times \text{list}(\alpha_3)) \rightarrow \alpha_5$. **(06 Marks)**

7 a. Explain in detail the storage organization. **(04 Marks)**

b. Explain on activation tree and activation record. **(06 Marks)**

c. Explain how to access nonlocal data on the stack. **(10 Marks)**

8 a. Explain the different issues of a code generation. **(10 Marks)**

b. Mention the different types of optimization of basic blocks. Explain any two with examples. **(10 Marks)**

* * * * *