Code : 051516

B.Tech 5th Semester Exam., 2019

COMPILER DESIGN

Time : 3 hours *Full Marks* : 70

Instructions :

(i) The marks are indicated in the right-hand margin.

*(ii) There are **NINE** questions in this paper.*

*(iii) Attempt **FIVE** questions in all.*

*(iv) Question No. **1** is compulsory.*

1. Choose the correct answer of the following (any *seven*) : 2×7=14

(a) When is the type checking usually done?

(i) During syntax directed translation

(ii) During lexical analysis

(iii) During code optimization

(iv) During syntax analysis

(b) Which one of the following statements is true?

(i) Canonical LR parser is more powerful than LALR parser.

(ii) SLR parser is more powerful than LALR.

(iii) LALR parser is more powerful than canonical LR parser.

(iv) SLR parser, canonical LR parser and LALR parser all have the same power.

(c) In a compiler, ____ checks every character of the source text.

(i) the lexical analyzer

(ii) the syntax analyzer

(iii) the code generator

(iv) the code optimizer

(d) For a grammar G, shift reduce (S-R) conflicts are present in LALR(1) parser, if and only if

(i) the LR(1) parser for G has S-R conflicts

(ii) the LR(0) parser for G has S-R conflicts

(iii) the SLR(1) parser for G has S-R conflicts

(iv) the SLR(0) parser for G has S-R conflicts

(e) In an absolute loading scheme, which loader function is accomplished by programmer?

(i) Allocation

(ii) Linking

(iii) Reallocation

(iv) Both (i) and (ii)

(f) _____ is a top-down parser.

(i) Operator precedence parser

(ii) An LALR (k) parser

(iii) An LR (k) parser

(iv) Recursive descent parser

(g) The languages that need heap allocation in the runtime environment are those that

(i) use global variables

(ii) use dynamic scoping

(iii) support recursion

(iv) allow dynamic data structure

(h) In compilers, generation of intermediate code based on an abstract machine model is useful because

(i) syntax-directed translations can be written for intermediate code generation

(ii) to generate code for real machines directly from high-level language program is not possible

(iii) portability of the front end of the compiler is enhanced

(iv) implementation of lexical and syntax analyses is easier

(i) To convert an arbitrary CFG to an LL(1) grammar

(i) factor the grammar alone

(ii) remove left recursion alone

(iii) remove left recursion and factor the grammar

(iv) None of the above

(j) The method which merges the bodies of two loops is

(i) loop rolling

(ii) loop jamming

(iii) constant folding

(iv) None of the above

2. (a) What is an activation record? Explain how they are used to access various local and global variables.

(b) What is bottom-up parsing? Discuss shift reduce parsing technique in brief. What is a handle? 7+7=14

3. (a) What is left recursion? Eliminate the left recursion from the following grammar :

$$E \rightarrow E+T \mid T$$
$$T \rightarrow T*F \mid F$$
$$F \rightarrow (E) \mid id$$

(b) What is the use of a symbol table? How are the identifiers stored in the symbol table? 7+7=14

4. (a) What is the pass of a compiler? Explain how the single- and multi-pass compilers work.

(b) Explain architecture and algorithm for the non-recursive predictive parser. 7+7=14

5. (a) Check whether the following grammar is CLR or not :

$$S \rightarrow Aa \mid bBa \mid Ba \mid bAc$$
$$A \rightarrow c$$
$$B \rightarrow d$$

(b) What is the syntax directed translation and why are they important? 7+7=14

6. (a) Explain different phases of compiler.

(b) Explain how type checking and error reporting are performed in compiler. Draw syntax tree and DAG for the statement.

$$a = (a * b + c) \wedge (b + c) * b + c$$ 7+7=14

7. (a) Explain various targets for code optimization with examples.

(b) How are CPU registers allocated while creating machine code? 7+7=14

8. (a) Check whether the following grammar is LL(1) grammar or not :

$$S \rightarrow iEtSA \mid a$$
$$A \rightarrow eS \mid \varepsilon$$
$$E \rightarrow b$$

(b) Design the FIRST SET and FOLLOW SET for the grammar 7+7=14

9. *(a)* Differentiate between S-attribute SDT and L-attribute SDT with suitable examples.

(b) Explain any *two* of the following :

(i) Lexical phase error

(ii) Code generation using dynamic programming

(iii) Syntax tree 7+7=14

* * *