Reg No.:_____ Name:_____

APJ ABDUL KALAM TECHNOLOGICAL UNIVERSITY

SIXTH SEMESTER B.TECH DEGREE EXAMINATION, APRIL 2018

Course Code: CS304

Course Name: COMPILER DESIGN (CS, IT)

Max. Marks: 100 Duration: 3 Hours

PART A

Answer all questions, each carries 3 marks. Marks

1 Draw the transition diagram for the regular definition, (3)

relop→< | <= | = | <> | >= | >

2 With an example source language statement, explain tokens, lexemes and patterns. (3)

3 Define LL(1) grammars. (3)

4 Is the grammar S→S(S)S/ Ɛ ambiguous? Justify your answer. (3)

PART B

Answer any two full questions, each carries 9 marks.

5 a) Apply bootstrapping to develop a compiler for a new high level language P on machine N. (3)

b) Now I have a compiler for P on machine N. Apply bootstrapping to obtain a compiler for P on machine M. (4)

c) Define cross-compilers. (2)

6 a)

Consider the following grammar

E→ E or T | T
T→ T and F |F
F→ not F | (E) | true | false

(2)

(i) Remove left recursion from the grammar. (4)

(ii) Construct a predictive parsing table. (3)

(iii) Justify the statement " The grammar is LL (1)".

7 a) Design a recursive descent parser for the grammar S→cAd, A→ab/ b (5)

b) For a source language statement a= b*c - 2, where a, b and c are float variables, * and – represents multiplication and subtraction on same data types, show the input and output at each of the compiler phases. (4)

PART C

Answer all questions, each carries 3 marks.

8 Compute the FIRST and FOLLOW for the following Grammar. (3)

Reg No.:_____ Name:_____

APJ ABDUL KALAM TECHNOLOGICAL UNIVERSITY

SIXTH SEMESTER B.TECH DEGREE EXAMINATION, APRIL 2018

Course Code: CS304

Course Name: COMPILER DESIGN (CS, IT)

Max. Marks: 100 Duration: 3 Hours

PART A

Answer all questions, each carries 3 marks. Marks

1 Draw the transition diagram for the regular definition, (3)

relop→< | <= | = | <> | >= | >

2 With an example source language statement, explain tokens, lexemes and patterns. (3)

3 Define LL(1) grammars. (3)

4 Is the grammar S→S(S)S/ Ɛ ambiguous? Justify your answer. (3)

PART B

Answer any two full questions, each carries 9 marks.

5 a) Apply bootstrapping to develop a compiler for a new high level language P on machine N. (3)

b) Now I have a compiler for P on machine N. Apply bootstrapping to obtain a compiler for P on machine M. (4)

c) Define cross-compilers. (2)

6 a)

Consider the following grammar

E→ E or T | T
T→ T and F |F
F→ not F | (E) | true | false

(2)

(i) Remove left recursion from the grammar. (4)

(ii) Construct a predictive parsing table. (3)

(iii) Justify the statement " The grammar is LL (1)".

7 a) Design a recursive descent parser for the grammar S→cAd, A→ab/ b (5)

b) For a source language statement a= b*c - 2, where a, b and c are float variables, * and – represents multiplication and subtraction on same data types, show the input and output at each of the compiler phases. (4)

PART C

Answer all questions, each carries 3 marks.

8 Compute the FIRST and FOLLOW for the following Grammar. (3)

S→Bb/Cd B→aB/Ɛ C→cC/Ɛ

9 Demonstrate the identification of handles in operator precedence parsing? (3)

10 Design a Syntax Directed Definition for a Desk calculator that prints the result. (3)

11 Describe the type checking of functions. (3)

PART D

Answer any two full questions, each carries 9 marks.

12 a) Construct canonical LR(0) collection of items for the grammar below. (5)

S → L = R
S → R
L → * R
L → id
R → L

Also identify a shift reduce conflict in the LR(0) collection constructed above.

b) Define S-attributed and L-attributed definitions. Give an example each. (4)

13 a) Explain bottom- up evaluation of S- attributed definitions. (5)

b) With an SDD for a desk calculator, give the appropriate code to be executed at each reduction in the LR parser designed for the calculator. Also give the annotated parse tree for the expression (3*5) –2. (4)

14 a) Construct LALR parse table for the grammar S->CC,C->cC|d (9)

PART E

Answer any four full questions, each carries 10 marks.

15 a) Write syntax directed definitions to construct syntax tree and three address code for assignment statements. (10)

16 a) Explain quadruples and triples with an example each. (5)

b) Construct the syntax tree and then draw the DAG for the statement (5)

e := (a*b) + (c-d) * (a*b)

17 a) Explain static allocation and heap allocation strategies. (10)

18 a) With an example each explain the following loop optimization techniques: (i) Code motion (ii) Induction variable elimination and (iii) strength reduction (10)

19 a) Explain any two issues in the design of a code generator. (5)

b) Explain the optimization of basic blocks. (5)

20 a) Write the Code Generation Algorithm and explain the *getreg* function. (6)

b) Generate a code sequence for the assignment d=(a-b)+(a-c)+(a-c) (4)

****

S→Bb/Cd B→aB/Ɛ C→cC/Ɛ

9 Demonstrate the identification of handles in operator precedence parsing? (3)

10 Design a Syntax Directed Definition for a Desk calculator that prints the result. (3)

11 Describe the type checking of functions. (3)

PART D

Answer any two full questions, each carries 9 marks.

12 a) Construct canonical LR(0) collection of items for the grammar below. (5)

S → L = R
S → R
L → * R
L → id
R → L

Also identify a shift reduce conflict in the LR(0) collection constructed above.

b) Define S-attributed and L-attributed definitions. Give an example each. (4)

13 a) Explain bottom- up evaluation of S- attributed definitions. (5)

b) With an SDD for a desk calculator, give the appropriate code to be executed at each reduction in the LR parser designed for the calculator. Also give the annotated parse tree for the expression (3*5) –2. (4)

14 a) Construct LALR parse table for the grammar S->CC,C->cC|d (9)

PART E

Answer any four full questions, each carries 10 marks.

15 a) Write syntax directed definitions to construct syntax tree and three address code for assignment statements. (10)

16 a) Explain quadruples and triples with an example each. (5)

b) Construct the syntax tree and then draw the DAG for the statement (5)

e := (a*b) + (c-d) * (a*b)

17 a) Explain static allocation and heap allocation strategies. (10)

18 a) With an example each explain the following loop optimization techniques: (i) Code motion (ii) Induction variable elimination and (iii) strength reduction (10)

19 a) Explain any two issues in the design of a code generator. (5)

b) Explain the optimization of basic blocks. (5)

20 a) Write the Code Generation Algorithm and explain the *getreg* function. (6)

b) Generate a code sequence for the assignment d=(a-b)+(a-c)+(a-c) (4)

****