SASTRA DEEMED TO BE UNIVERSITY THINK MERIT | THINK TRANSPARENCY | THINK SASTRA	**SCHOOL OF COMPUTING** **Second CIA Exam – Sep 2024** **Course Code: CSE320** **Course Name: Compiler Design** **Duration: 90minutes Max. Marks: 50**

PART A

ANSWER ALL THE QUESTIONS **10 MARKS**

1. Consider the grammar **S → aSbS | bSaS |ε**. Show that the grammar is ambiguous for the string abab. (2)
2. What are the Inherited and Synthesized attributes. (2)
3. Write the significance of Augmented Grammar. (2)
4. What conflicts do LR parsers handle during syntax analysis? (2)
5. Illustrate the Three-Address Code (TAC) for the following Python procedure call statement sum(x+y,n); (2)

PART B

ANSWER ALL THE QUESTIONS **25 MARKS**

6. (a) Construct LALR Parsing table for the following grammar.

S → Aa | bAc | Bc | bBa
A → d
B → a (10)

OR

(b) Construct CLR Parsing table for the following grammar

S → (L) |id
L → L * S | S.
Also parse the input string **(id*id*id).** (10)

7. (a) Check whether the given context free grammar is SLR (1) Grammar or not.

A → B * C | C
B → *C | num
C → B (10)

OR

(b) Translate the given Boolean expression

(a<(c+d))&&((–(a+b)*(a+b)-(a+b)*d)>=MAX)

into the sequence of TAC Statements and represent with Quadruples, Triples and Indirect Triples. (10)

8.(a) Specify the various levels of Errors in programming paradigm. Discuss about the Error Recovery strategies adopted by the Parsers when transforming a code into a parse tree. (5)

OR

(b) Show the Stack implementation of Shift Reduce Parser to validate the string **(id+id)*id** from the grammar

E→ E + T | T
T → T * id | (E) | id (5)

PART C

ANSWER ALL THE QUESTION **15 MARKS**

9. Write a LEX and YACC specification to evaluate the given expression involving only arithmetic operators in C++. Use the LEX code to validate the numbers and identifiers in the given input. With YACC, augment a context free grammar to validate the syntactical correctness of the expression then define the syntax directed translation rules to evaluate the resultant value of the given arithmetic expression. [note: Evaluation should be made as per the Rule of Precedence and Rule of associativity of Arithmetic operators defined in C++]. (15)

--- END---

SASTRA DEEMED TO BE UNIVERSITY (U/S 3 of the UGC Act, 1956) THINK MERIT | THINK TRANSPARENCY | THINK SASTRA	**SCHOOL OF COMPUTING** **Second CIA Exam – Sept 2024** **Course Code: CSE320** **Course Name: Compiler Design** **Duration: 90minutes Max. Marks: 50** **RBT**

PART A

ANSWER ALL THE QUESTIONS **10 MARKS**

1. Consider the grammar **S → aSbS | bSaS |ε**. Show that the grammar is ambiguous for the string abab. (L2)
2. What are the Inherited and Synthesized attributes. (L2)
3. Write the significance of Augmented Grammar. (L2)
4. What are the conflicts that the LR parsers are handling during syntax analysis. (L2)
5. Illustrate the Three Address Code statements for the following procedure call statement in Python: ***sum(x+y, n);*** (L2)

PART B

ANSWER ALL THE QUESTIONS **25 MARKS**

6. (a) Construct LALR Parsing table for the following grammar.

S → Aa | bAc | Bc | bBa
A → d
B → a (L3)

OR

(b) Construct CLR Parsing table for the following grammar

S → (L) |id
L → L * S | S.
Also parse the input string **(id*id*id).** (L3)

7. (a) Check whether the given context free grammar is SLR (1) Grammar or not.

A → B * C | C
B → *C | num
C → B (L3)

OR

(b) Translate the given Boolean expression

(a<(c+d))&&((–(a+b)*(a+b)-(a+b)*d)>=MAX)

into the sequence of TAC Statements and represent with Quadruples, Triples and Indirect Triples. (L3)

8.(a) Specify the various levels of Errors in programming paradigm. Discuss about the Error Recovery strategies adopted by the Parsers when transforming a code into a parse tree. (L1)

OR

(b) Show the Stack implementation of Shift Reduce Parser to validate the string **(id+id)*id** from the grammar

E→ E + T | T
T → T * id | (E) | id (L2)

PART C

ANSWER ALL THE QUESTION **15 MARKS**

9. Write a LEX and YACC specification to evaluate the given expression involving only arithmetic operators in C++. Use the LEX code to validate the numbers and identifiers in the given input. With YACC, augment a context free grammar to validate the syntactical correctness of the expression then define the syntax directed translation rules to evaluate the resultant value of the given arithmetic expression. [note: Evaluation should be made as per the Rule of Precedence and Rule of associativity of Arithmetic operators defined in C++]. (L3)

--- **END**---

[RBT Levels: L1 – Remember; L2 – Understand; L3 – Apply]

SASTRA DEEMED TO BE UNIVERSITY (U/S 3 of the UGC Act, 1956) THINK MERIT | THINK TRANSPARENCY | THINK SASTRA	**SCHOOL OF COMPUTING** **Second CIA Exam – Sept 2024** **Course Code: CSE320** **Course Name: Compiler Design** **Duration: 90minutes Max. Marks: 50** **ANSWER KEY**

PART A

ANSWER ALL THE QUESTIONS **10 MARKS**

1. *Consider the grammar S → aSbS | bSaS |ε. Show that the grammar is ambiguous for the string abab.* *(2)*

   Two different LMD / RMD or Derivation Trees – 1 for each.

2. *What are the Inherited and Synthesized attributes.* *(2)*

   An inherited attribute is an attribute whose value is determined based on the values of attributes of its parent or sibling nodes.

   A synthesized attribute is an attribute whose value is determined based on the values of attributes of its child nodes.

3. *Write the significance of Augmented Grammar.* *(2)*

   The use of augmented grammar is crucial for:

   - Establishing a clear start and end point for parsing.
   - Simplifying the construction and management of parsing tables.
   - Handling the end-of-input situation effectively.

4. *What are the conflicts that the LR parsers are handling during syntax analysis*. (2)

   (i) A **shift-reduce conflict** occurs when the parser encounters a situation where it can either shift a new symbol onto the stack or reduce the symbols on the stack according to a grammar rule, but it is not clear which action to take.

   (ii) A **reduce-reduce conflict** occurs when the parser can apply more than one reduction rule to the symbols currently on the stack, but it is unclear which reduction to apply.

5. *Illustrate the Three Address Code statements for the following procedure call statement in Python: **sum(x+y, n);*** (2)

```
t0 = x+y
param t1
param n
call sum, 2
```

PART B

ANSWER ALL THE QUESTIONS **25 MARKS**

6. (a) *Construct LALR Parsing table for the following grammar*.

   S → Aa | bAc | Bc | bBa

   A → d

   B → a (10)

Sl No	Content	Allocation
1	Computation of LR(1) Item sets Augmented Grammar – 1 Mark Closure – initial Set – 1 mark Other Item Sets – 3 Marks	5
2	Merging of LR(1) Item sets	2
3	LALR Parsing Table Construction	3
TOTAL Marks		**10**

OR

(b) *Construct CLR Parsing table for the following grammar*

S → (L) |id

L → L * S | S.

Also parse the input string **(id*id*id).** (10)

Sl No	Content	Allocation
1	Computation of LR(1) Item sets Augmented Grammar – 1 Mark Closure – initial Set – 1 mark Other Item Sets – 3 Marks	5
2	CLR Parsing Table Construction	3
3	Parsing the Input String	2
Total Marks		**10**

7. (a) *Check whether the given context free grammar is SLR (1) Grammar or not.*

A → B * C | C

B → *C | num

C → B (10)

Sl No	Content	Allocation
1	Computation of LR(0) Item sets Augmented Grammar – 1 Mark Closure – initial Set – 1 mark Other Item Sets – 3 Marks	5
2	First & Follow sets computation	1
3	SLR Parsing Table Construction	3
4	Conclusion	1
Total Marks		**10**

OR

(b) *Translate the given Boolean expression*

(a<(c+d))&&((–(a+b)*(a+b)-(a+b)*d)>=MAX)

into the sequence of TAC Statements and represent with Quadruples, Triples and Indirect Triples. (10)

Sl No	Content	Allocation
1	Generation of TAC Statements	4
2	Quadruple Representation	3
3	Triple representation	2
4	Indirect Triple representation	1
Total Marks		**10**

8.(a) *Specify the various levels of Errors in programming paradigm. Discuss about the Error Recovery strategies adopted by the Parsers when transforming a code into a parse tree*. (5)

Specifying the Levels of Error – 1 Mark

Explanation of Recovery Strategy – 1 each, total 4 marks

OR

(b) Show the Stack implementation of Shift Reduce Parser to validate the string **(id+id)*id** from the grammar

E→ E + T | T

T → T * id | (E) | id (5)

Sl No	Content	Allocation
1	Stepwise Reduction, ends with Start Symbol, tabular representation, showing the handle for the step.	5

PART C

ANSWER ALL THE QUESTION **15 MARKS**

9. Write a LEX and YACC specification to evaluate the given expression involving only arithmetic operators in C++. Use the LEX code to validate the numbers and identifiers in the given input. With YACC, augment a context free grammar to validate the syntactical correctness of the expression then define the syntax directed translation rules to evaluate the resultant value of the given arithmetic expression. [note: Evaluation should be made as per the Rule of Precedence and Rule of associativity of Arithmetic operators defined in C++]. (15)

Sl No	Content	Marks Allocation
1	LEX Code – Definition Section	2
2	LEX Code – Translation Section	3
3	YACC Code – Definition Section	2
4	YACC Code – Rules Section including token definition & precedence definition.	5
5	YACC Code – main () definition	1
6	YACC Code – yyerror () definition	1
7	YACC Code – yywarp () definition	1
TOTAL Marks		**15**

--- **END**---