Reg.No | | | | | | | | | | |

SNS College of Technology, Coimbatore-35.

ɔnomous)

B.E/B.Tech- Internal Assessment -III

Academic Year 2023-2024(ODD)

Fifth Semester

Computer Science and Engineering

19CSB301 – Automata Theory and Compiler Design

B

Time: 1.5 Hours **Maximum Marks: 50**

Part-A (5 x 2 =10 Marks)

		CO	Blooms
1.	Differentiate between Syntax Tree and Parse Tree	CO4	Ana

Parse Tree	Syntax Tree
A parse tree is a graphical representation of a replacement process in a derivation	A syntax tree (AST) is a condensed form of parse tree
Each interior node represents a grammar rule	Each interior node represents an operator
Each leaf node represents a terminal	Each leaf node represents an operand
Parse tree represent every detail from the real syntax	Syntax tree does not represent every detail from the real syntax Eg : No parenthesis

2. Define Activation Record and Activation Tree — CO4 — Und

An activation record stores all information that is required to call a procedure.

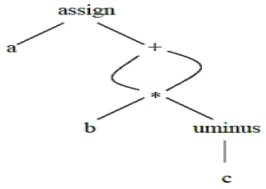
An activation tree is a tree structure that represents function calls made by a program during execution. When a function is called a new activation record is pushed to the stack and popped from the stack when the function returns.

3. Define Constant folding with an example — CO5 — Rem

In this method, if the value of an expression is constant,use the constant directly instead of the expression.

pi = 22/7; ⟹ pi = 3.14;

4. Draw the DAG for the statement a = b * – c + b * – c CO5 App



5. Find the Object code Sequence for t:=a+b produced by a typical code generator CO5 APP

```
MOV x, R0
ADD y, R0
```

Part-B (2x13+14=40 Marks)

6. a. Construct the canonical parsing table for the grammar given below. Check whether the string "cdcd " is accepted or not. 13 CO4 App

S->CC

C->cC

C->d

1. Construct Augumented Grammar
2. Construct Canonical LR(1) items
3. Construct CLR Parsing table

4.Parsing i/p string using CLR Parse table

or

b. Define three address code. Describe the various methods of implementing three address statements with an example. 13 CO4 Und

- Three-address code is an intermediate code. It is used by the optimizing compilers.
- In three-address code, the given expression is broken down into several separate instructions. These instructions can easily translate into assembly language.
- Each Three address code instruction has at most three operands. It is a combination of assignment and a binary operator.t1 = uminus c

t2 = b * t1

t3 = uminus c

t4 = b * t3

t5 = t2 + t4

a = t5

#	Op	Arg1	Arg2	Res
(0)	uminus	c		t1
(1)	*	b	t1	t2
(2)	uminus	c		t3
(3)	*	b	t3	t4
(4)	+	t2	t4	t5
(5)	:=	t5		a

Quads (quadruples)

#	Op	Arg1	Arg2
(0)	uminus	c	
(1)	*	b	(0)
(2)	uminus	c	
(3)	*	b	(2)
(4)	+	(1)	(3)
(5)	:=	a	(4)

Triples

#	Stmt
(0)	(14)
(1)	(15)
(2)	(16)
(3)	(17)
(4)	(18)
(5)	(19)

#	Op	Arg1	Arg2
(14)	uminus	c	
(15)	*	b	(14)
(16)	uminus	c	
(17)	*	b	(16)
(18)	+	(15)	(17)
(19)	:=	a	(18)

7. a. Explain the various techniques for storage allocation with examples 13 CO5 App

The different ways to allocate memory are:

1. Static Allocation: It is for all the data objects at compile time.
2. Stack Allocation: In this a stack is used to manage the run time storage. For example recursive calls make use of this area.

or

b. Analyzehow the Code optimization is performed in compiler with Examples 13 CO5 Ana

1. Common-Subexpression Elimination:
2. Copy Propagation
3. Dead Code Elimination
4. Constant folding
5. Loop Optimizations

8. a. Demonstrate about the translation scheme to generate three address code for Declarations and Assignment Statements 14 CO4 Und

$P \rightarrow D ; E$
$D \rightarrow D ; D$
$D \rightarrow id : T \{ addtype (id.entry , T.type) \}$
$T \rightarrow char \{ T.type : = char \}$
$T \rightarrow integer \{ T.type : = integer \}$
$T \rightarrow \uparrow T1 \{ T.type : = pointer(T1.type) \}$
$T \rightarrow array [num] of T1 \{ T.type : = array (1 \ldots num.val , T1.type) \}$

```
S->id:=E {p:=lookup(id.name);
          if (p!=nil) then
          emit(p ':=' E.place); else error}
E->E1+E2 {E.place = newtemp;
          emit(E.place ':=' E1.place '+' E2.place);}
E->E1*E2 {E.place = newtemp;
          emit(E.place ':=' E1.place '*' E2.place);}
E->-E1    {E.place = newtemp;
          emit(E.place ':=' '-' E1.place);}
E->(E1)  {E.place = E1.place;}
E->id     {p = lookup(id.name);
          if (p!= nil) then E.place := p; else error;}
```

or

b. Discuss the Various Issues in the design of Code Generator 14 CO5 Rem

1. Input to code generator
2. Target program
3. Memory Management
4. Instruction selection
5. Register allocation issues
6. Evaluation order