Unit – 1

1.Define compiler , interpreter. And translator

2.Define construction tool kit

3.Explain the different phases of compiler with an example

OR

Explain the different phases of compiler .Explain the kind of transformation done on the source program

By the individual phases of compiler on the statement position= initial +rate \* 60

4.Difference between multi parse and single parse compiler.

5.Explain bootstrapping

6. Construct minimum state dfa for regular expression

(0+1 )\* 00+ 01

OR

Convert nfa to dfa



7.consider the following program

Main ()

{

int x,y,z ;

z = x + y ;

}

List down all the lexemes, tokens and the attributes of the tokens at the end of lexical analysis of the above program .

8. Draw the transition diagram of

a. Identifier

b .Relational operator

9. What is lex ? Explain with suitable code

10. Show that the following grammaris ambiguous

E = E+E |E\*E |( E) | id

**Unit-2**

**SHORT QUESTIONS**

1. Consider the grammar

S (S)S / ε

Show top down parsing for string **(( ))**

2. Define context free grammar with suitable example and answer the following

Amongst which of them will be the first machine to parse CFG successfully

a. Turing Machine

b. Push Down Automata

c. Finite Automata

d. Lower bound automata

3. Define left factoring and eliminate left factoring from follow grammar (if there is any)

S ietSeS / ietS / a

E t

4. Eliminate left recursion from the follow grammar

E E + T / T

T T \* F / F

F (E) / id

**LONG QUESTIONS**

1. Explain classification of parser with suitable diagram

2. Write the rules to find first and follow. Also calculate first and follow for the following grammar

E TA

A +TA / ε

T FB

B \*FB / ε

F (E)/id

3. Consider the following grammar.

S AA

A aA

A b

Construct the LALR parsing table

4. Consider following grammar

E E + T / T

T T \* F / F

F (E) / id

Construct predictive parse table and also check that given string is successfully parsed or not.

W= id\*id+id

5. Construct CLR table

S AaAb / BbBa

A ε

B ε

6. Construct the SLR parsing table for the following grammar

S xAy/ xBy/xAz

A As/b

B b

7. Construct operator precedence parse table for the given grammar id+id\*id

E E + T

E T

T T \* F

T F

F (E)

F id

COMPILER DESIGN UNIT-3

**LONG QUESTIONS**:-

(Q1) Explain synthesized and Inherited attribute with example.

(Q2) For the given grammar

E→E₁+T, E→E₁-T, E→T, T→T₁\*F,

T→T₁/F, T→F, F→ (E), F→num

Give the semantic rule to represent syntax directed

definition for arithmatic expression and draw

annotated parse tree for 4-6/3+5.

(Q3) For the given arithmatic expression,

Q: = -b \* (c+d)

give the quadruple, triple and indirect triple representation with brief explanation.

(Q4) Explain S-attributed and L-attributed definitions. Give examples to support your answer.

(Q5) What are quadruples, triples and indirect triples.Give their examples.

Give the quadruples, triple and indirect triple representation of the statement:

x = (a + b) \* + − c/d.

(Q6) Translate the expression a <b and c>d to three-address code.

(Q7) Write the three address code for the following program fragment:

while (P<q and r > s) do

if P=1 then q=q+1

else

while (p <= s) do

P= P + 3

(Q8) Write short notes on :

(i) Postfix Notation

(ii) Translation of procedure calls.

(Q9) Consider the following grammar:

E-> E+T

E->T

T-> T\*F

T->F

F-> id

And write :

(i) Postfix representation

(ii) Syntax tree representation

(iii) Three address code representation

(Q10) Explain in details the construction of a syntax tree for a binary infix arithmetic expression :

a + a\*(b − c) + (b − c)\*d

**SHORT QUESTIONS**

(Q1) What is the role of semantic analyser?

(Q2) What are synthesized and inherited attributes?

(Q3) Define SDD(Syntax Directed Defination).

COMPILER DESIGN UNIT-4

**LONG QUESTIONS**

1)explain in details parameter passing and storage allocation.

2)explain structural linear and Hybrid types of intermediate representations with examples.

3)differentiate between static stack and heap allocation strategies.

4)describe the following terms - i)activation tree ii)symbol table manager

5)discuss the following parameter passing technique with example-

i)call by reference

ii)call by value

iii)call by name

6)explain dynamic storage allocation techniques.

7)explain different data structures used for symbol table organization.

8)explain activation record and its fields.

9)what are different parameter passing methods explain with examples.

10)explain different allocation strategies.

**SHORT QUESTIONS**

1) Define local and global variable.

2) What is activation tree?

3) Define allocation strategies.

**Unit-5**

2 marks

* Define code optimization
* Discuss code generator
* What are basic blocks
* Explain partition algo for basic blocks
* Advantages of loop optimization
* Describe flow graph
* Explain any two local optimization techniques

8 marks

* Explain various issues that occur during code generation.
* Explain various code optimization techniques.
* Generate code sequence for a: = x-y + x-z + x-z.
* Translate code to basic block and flow graph.
* Apply code generation algorithm to : a = (p+q) – ((r+s) – t)
* Construct the following expression with its three address code :

a = (p+q) – ((r+s) – t)

t1 = p+q

t2 = r+s

t3 = t2-t

a = t1 + t2

* Explain loop optimization, benefits, techniques and its steps.
* Explain global data flow analysis with suitable example.
* Difference between local and global optimization
* Write short notes on
* Error recovery techniques
* Target machine
* Dead code elimination
* Loop in variant computation
* IR types
* Register allocations