ASSIGNMENTS

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

ACADEMIC YEAR: **2019-20** YEAR/SEMESTER: **III-I Sem**
FACULTY NAME**: KRISHNA KISHORE THOTA**
SUBJECT: **COMPILER DESIGN**

UNIT-I

1. What is a Compiler? What are the differences between Compiler & Interpreter?
2. What are various Phases of a Compiler? Explain each phase for the following statement **position: = initial + rate * 60**
3. Draw the Transition diagram for recognition of tokens, Reserved words and identifiers
4. Explain the Terms: Token, patterns, Lexemes, Lexical Errors, Regular Expressions,Regular definitions each with an example
5. Construct Regular definitions for the language constructs – Strings, Sequences,Comments
6. Explain in detail about Language Processing System
7. Explain in detail about Structure of a Compiler
8. Discuss in brief about the Role of Lexical analyzer in a compiler.
9. a. Explain in brief about Lexical errors.
   b.Reasons for separating scanner and parser
10. What do you mean by front end in the compiler design? Show the output produced by it in different stages for ***a:=b*c/36***; where a, b and c are real numbers.

UNIT-II

1 .Construct LMD, RMD, Parse Tree for deriving the strings
(i) (a,a) **(ii) (a,(a,a))** **(iii) ((a,a),(a,a))**
from the following CFG:
S -> (L) / id **L -> L,S / S**

2 .Explain Top down parsing with an example? Compute First() and Follow() for the following CFG:

E -> TE'
E' -> +TE' / €
T -> FT'
T' -> *FT' / €
F -> (E) / id

3 a) What are the preprocessing steps required for predictive parse table construction? Consider the grammar

***S -> ACB|CbB|Ba, A->da|BC, B->g|* ε*, C ->h|* ε**

b) Construct the predictive parse table for the above grammar. And also check for the validity of the input string of your choice.

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

S -> AaAb / BaBb A -> ε B -> ε

5 a) Check whether the given grammar is LL(1) or not?

G: S->Aa|bAc|Bc|bBa, A->d, B->d

6) With neat sketch explain the structure of non-recursive predictive parser. How to handle errors in it.

7) Prove that the given grammar is ambiguous and eliminate ambiguity in it.

G: S ->iEtSeS|iEtS|a, E->b|c|d

8) Construct the recursive descent parser for

G: ***bexpr ->bexpr or bterm|bterm,***

bterm ->bterm and bfactor|bfactor, bfactor ->not factor|(bexpr)|true|false.

9.Explain in detail about Error recovery in predictive parsing

10. Discuss Left Recursion and Left Factoring in context free grammars?

UNIT-III

1.Implement Shift Reduce Parsing for the following CFG:

E -> E+T / T T -> T*F / F F -> (E) / id

2.Construct SLR Parsing Table for the following CFG:

S -> CC

C->cC / d

3 Construct CALR Parsing Table for the following CFG:

S -> L=R / R L -> *R / id R -> L

4. Differentiate between Top down and Bottom up Parsing methods. Construct CLR parser for the grammar

S -> L=R / R L -> *R / id R -> L

5. Explain the structure of LR parsers. How they are different from LL parsers?

6. Build LR(0) parser and check the validity of the input string *"id+id*id "* bythe LR(0) parser for the given grammar ***E->E+T/T, T->T*F/F, F-> (E)/id***

7. Explain the usage of precedence and association rules to handle shift reduce conflicts in LR parsers.

8. Explain the error recovery inLR parsers

9.a)What is the importance of look ahead symbol in LR(1) parser? Construct the canonical LR parser for
G: S->L=R|R, L->*R|id, R->L
b)Explain the rules to check the acceptance of input string : ****id=*id***

10. a.List out and explain the rules to construct simple precedence relation for a context free grammar.
b.Construct the operator precedence parse table for E->EAE|(E)|-E|id, A->+|-|*|/|

UNIT-IV

1. Construct SDT for a Simple Desk Calculator Program

2. Draw Annotated Parse Tree for deriving the following sentence:

int id1, id2, id3

from the following CFG:

D -> TL
T -> int / real
L -> L, id / id

3. Write the Quadruple, Triple & Indirect Triple Representation of the following expression:
a + a*(b-c) + (b-c)*d

4. Explain the type system in type checker? Write the syntax directed definition for typechecker.

5. What is syntax directed translation? Write the semantic rules for
D->TL, T ->int|real, L->L,id|id

6.What is an Abstract syntax tree? How to construct it using *mknode(), mkleaf()* functions? Give an example.

7. What is type expression? How to construct them using various type constructors? Explain.

8. Differentiate bottom up and top down evaluation of semantic rules for arithmetic expressions

9.If **(a< b+c *20)**

```
{       a =a* b – 50
        d = (a/b) + 25;
        print  ( a,d )
}
```

For the given code generate three-address code.

UNIT-V

1. Explain in detail about Symbol Tables? Also the use of symbol tables?
2. Construct DAG for the following code:

   A = B + C

   B = A – D

   C = B + C

   D = A - D
3. What is runtime stack? Explain storage allocation strategies used for recursive procedure calls.
4. What is scope of variable? Write about various ways to access non loca lvariables.
5. Generate target code from sequence of three address statements using simple code generator algorithm.
6. Explain in detail about Register Allocation
7. Explain about basic and flow graphs
8. Explain the issues in the code generation
9. Explain the Heap management
10. Explain the code generation algorithm

UNIT-VI

1. Explain in detail about Machine dependent code optimization?
2. Explain in detail about global common sub-expression & copy propagation optimization techniques each with an example?
3. Write a short note on Instruction Scheduling and Inter Procedural Optimization?
4. Write about the techniques in local and global transformations.What is machine independent optimization? What are the different techniques used for it.
5. How to schedule the instructions to produce optimized code? Explain.

6) Write the algorithm to generate basic blocks and flow graph for quick sort algorithm.

7) Apply the code optimization techniques on flow graph generated for quick sort.

8.Differentiate various techniques used for machine independent and dependent optimizations.

9.Explain how code motion and frequency reduction used for loop optimizations

10) a .Explain in detail about Loop Optimization.

b. Explain in brief about Peephole optimization techniques.