Syllabus of Compiler Design (CSE420)

Lecture No	Topic
Lecture 1:	Introduction to Compilers, Review the fundamentals concepts of computer language processing system.
Lecture 2:	Different phases of a compiler, and analysis-synthesis model of compilation
Lecture 3:	Lexical Analysis: Token, Pattern, Lexeme, Regular Expressions, DFA, NFA, RE to NFA using Thompson's rule.
Lecture 4:	Lexical Analysis: RE directly to DFA by computing followpos
Lecture 5:	Syntax Analysis: Introduction to Parsing, Context Free Grammar, Parse tree
Lecture 6:	Syntax Analysis: Dealing with Ambiguous Grammar
Lecture 7:	Syntax Analysis: Left Recursion Elimination, Left factoring.
Lecture 8:	Syntax Analysis: Top down parsing, Recursive descent parsing, computing FIRST set.
Lecture 9:	Computing FOLLOW set, Predictive parsing, LL (1) Grammar. Dealing with errors in Top down parsing, Panic and phrase level error recovery.
Lecture 10:	Bottom up parsing, Shift-Reduce parsing, SR parsers, LR parsing, Computing closure for SLR
Lecture 11:	Computing GOTO, LR(0) items, LR(0) automation, basic SLR parser
Lecture 12:	Mid Term Examination
Lecture 13:	LR(1) items, LR(1) automation, LR(1) parsing table
Lecture 14:	LALR parsing, using ambiguous grammar
Lecture 15:	Syntax Directed Translation: Semantic rules, synthesized and inherited attributes
Lecture 16:	Syntax Directed Translation: SDD vs SDT; SDD based on bottom up parsing, Annotated parse tree, Dependency graph, Evaluation order
Lecture 17:	Syntax Directed Translation: SDD based on top down parsing
Lecture 18:	Syntax Directed Translation: Type Checking
Lecture 19:	Intermediate Code Generation: Linear and Graphical representation, Three address code, DAG, Implementation of DAG, Value number method, Triple, Quadruple, Indirect triples.

Lecture 20:	Intermediate Code Generation: Translating Expressions, SDT into three address code, Short circuit code, SDT for flow of control statements, Generating for three address code for Booleans.
Lecture 21:	Code optimization: basic blocks, flow graphs, instruction-level optimization, common-sub expression elimination, peep-hole optimization.
Lecture 22:	Code optimization: Larger-scale parallelism detection and exploitation, Loop optimization.
Lecture 23:	Code generation: Instruction selection, Register Allocation, Cost analysis.
Lecture 24:	Code Generation: Next use information, Code generation algorithm.
Lecture 25:	Run time environment: Activation tree, storage allocation, static, stack and heap allocation
Lecture 26:	Run time environment: Activation record, Scoping rules, Links and Parameter passing