COMPILER DESIGN IMPORTANT QUESTIONS.

Unit -1

SHORT QUESTIONS:

Define compiler.
What is Context free grammar?
Define pre-processor. What are the functions of pre-processor?
What is input buffer?
Differentiate compiler and interpreter
What is input buffering?
Define the following terms: a) Lexeme b) Token
Define interpreter.
What are the differences between the NFA and DFA?

LONG questions:

Explain the various phases of a compiler with an illustrative example
Define Regular expression. Explain the properties of Regular expressions.
Differentiate between top down and bottom up parsing techniques.
Construct an FA equivalent to the regular expression (0+1)*(00+11)(0+1)*
Explain the various phases of a compiler in detail. Also write down the output for the following expression: position: =initial + rate * 60
Construct an FA equivalent to the regular expression 10+(0+11)0*1

Unit -2

Short questions :

Define augmented grammar
Compare the LR Parsers.
Compare and contrast LR and LL Parsers
Differentiate between top down parsers
Define Dead code elimination?
Eliminate immediate left recursion for the following grammar: E->E+T | T T->T* F | F F-> (E) | id
Mention the types of LR parser.
Explain bottom up parsing method

LONG questions :

Discuss in about left recursion and left factoring with examples.
Construct the predictive parser for the following grammar S->(L)/a L->L,S/S
Check whether the following grammar is SLR (1) or not. Explain your answer with Reasons. S→ L=R S→R L→*R L→id R→L
Construct SLR parse table for S->L=R/R R->L L->*R/id
State and explain the rules to compute first and follow functions E->E+T/T T->T*F/F F->F*/a/b
Construct CLR parse table for S->L=R/R R->L L->*R/id
Construct the LR Parsing table for the following grammar: E→E + T | T T→T * F | F F → (E)/id
Construct an LALR Parsing table for the following grammar: E-> E+T |T T-> T*F|F F->id
Find the SLR parsing table for the given grammar: E->E+E | E*E | (E) | id. And parse the sentence (a+b)*

UNIT-3

SHORT QUESTIONS:

Define Type Equivalence
Explain the role of intermediate code generator in compilation process
Define left most derivation and right most derivation with example
What are the various types of intermediate code representation?
Write a note on the specification of a simple type checker.
Explain intermediate code representations?
Define type expression with an example?
State general activation record?
Explain type expression and type systems

LONG QUESTIONS:

Explain in brief about equivalence of type expressions with examples
Explain about Type checking and Type Conversion with examples
What is a three address code? Mention its types. How would you implement the three address statements? Explain with examples.
What is type checker? Explain the specification of a simple type checker
Translate the following expression: (a + b) * (c + d) + (a + b + c) into a) Quadruples b) Triples
Construct a quadruple, triples for the following expression: a +a*(b-c)+(b-c)*d?
Explain various storage allocation strategies with examples.
Explain static and stack storage allocations?

UNIT -4

SHORT QUESTIONS

Write the quadruple for the following expression (x + y)∗(y + z) + (x + y + z)
What is a DAG? Mention its applications.
What are Abstract Syntax trees?
Define address descriptor and register descriptor
Discuss about common sub expression elimination
What is a Flow graph?
Define constant folding?
Define reduction in strength?

LONG QUESTIONS :

Explain the issue and the difference between the heap allocated activation records versus stack allocated activation records
Write the principal sources of optimization
Discuss about the following: a) Copy Propagation b) Dead code Elimination c) Code motion.
Explain Lazy-code motion problem with an algorithm
Explain the following with an example: a) Redundant sub expression elimination b) Frequency reduction c) Copy propagation
Explain various method to handle peephole optimization.
Explain the following peephole optimization techniques: a) Elimination of Redundant Code b) Elimination of Unreachable Code
Illustrate loop optimization with suitable example.
Explain various code optimization techniques in detail.

UNIT -5

SHORT QUESTIONS

What are the induction variables?
Explain about code motion.
What are induction variables? What is induction variable elimination?
What is machine independent code optimization?
Write a short note on copy Propagation
What are the induction variables?
Write a short note on Flow graph.

LONG QUESTIONS

Explain data-flow schemas on basic blocks with flow graphs
Explain Lazy-code motion problem with an algorithm
Explain in brief about different Principal sources of optimization techniques with suitable examples.

Give an example to show how DAG is used for register allocation
Explain in detail about machine dependent code optimization techniques with their drawbacks
Explain in brief about the issues in the design of code generator.
Explain in detail about peep hole optimization.
Explain machine dependent and machine independent optimization?
Explain data-flow analysis of structural programs.
Explain in detail the procedure that eliminates global common sub expression