

**SUBJECT: COMPILER DESIGN**

**QUESTION BANK**

A.Y.:**2024-2025 SEM-II**

UNIT-1

1. Write about Phases of a compiler. Explain each with an example.
2. Discuss the phases of a compiler indicating the inputs and outputs of each phase in translating the statement “a=p+r\*36.0”.
3. Generate object code for x1=x2\*x3/15 through different phases of compiler
4. What are different phases of a Compiler? Explain with a diagram.
5. Explain different phases of a compiler with a neat diagram.
6. Explain the approach for designing of lexical analyzer.
7. Specify the need and role of Lexical Analyzer
8. Discuss about the role of lexical analyzer. Explain with program.
9. Why lexical and syntax analyzers are separated? Explain in detail.
10. Explain various data structures used in lexical analysis.
11. Write a Regular Expression for identifier, reserved words & relation operators. Design a transition diagram for each of them.
12. Explain about Input Buffering in lexical Analyzer with an example.
13. Describe the need and functionality of linkers, assemblers and loaders.
14. Explain various building blocks used to design a language translator.
15. Differentiate between i) Phase and a pass ii) single-pass and multi-pass compiler.
16. What is LEX? Discuss the usage of LEX in Lexical Analyzer generation.
17. Explain in detail about LEX tool.
18. Write the steps to convert Non-Deterministic Finite Automata (NDFA) into Deterministic Finite Automata (DFA)
19. Explain the boot strapping process with suitable examples and diagrams.
20. Define Regular Expression. Explain the properties of Regular Expressions. Discuss with suitable examples.
21. What is a regular expression? Design a transition diagram for keywords.
22. State the steps to convert a regular expression to NFA. Explain with an example.
23. Construct an FA equivalent to the regular expression (0+1)\*(00+11)(0+1)\*.
24. What is the role of transition diagrams in the construction of lexical analyzer?
25. Construct a Finite Automata and Scanning algorithm for recognizing identifiers, numerical constants in C language.
26. Write about tokens generated by lexical analyzers. Describe the lexical errors and various error recovery strategies with suitable examples.
27. How to specify the tokens? Differentiate token, lexeme and pattern with suitable examples.
28. What are the cousins of Compiler? Explain their role in language processing.
29. Differentiate:(i) Pass and phase of a compiler;(ii)Frontend and back end of a compiler.
30. Write a simple ‘C’ program to read and display a string. Design lexical analyzerto identify tokens in it.
31. Derive the regular expression for the tokens given below and also draw a transition diagram to recognize the following tokens: i) Relational operators ii) Integer constant iii) Identifier iv) White spaces v) Exponent part of a number
32. Explain the role of parser. Explain types of grammars used for parsing.
33. What is an ambiguous grammar? Write a procedure to eliminate the same withan example.
34. Consider the following grammarS → (L) |aL → L, S |S ConstructleftmostandRightmostderivationsandparsetrees for the followingsentences:
35. i. (a,(a,a))
36. ii.(a,((a,a),(a,a))).
37. Giveanalgorithmtoeliminateproductionscontaininguseless symbols andambiguous productions from a grammar.
38. Define Context Free Grammar. Explain how it is suitable for parsing?
39. Given the following grammar: E -> E + E | E - E | E \* E | E /E | - E | int Showtwo different left-most derivations with the helpofparsetreesforthestringint+int\*int/int.Whatdoes this tell you?
40. Explainleftrecursionandleftfactoringwithexamples.
41. What is mean by left recursion? How to eliminate left recursion for thefollowing grammar:E->E+T/T, T->T\*F/F, F->(E)/id
42. What is an LL(1) grammar? Can you convert every context free grammar intoLL(1). How to check the grammar is LL(1) or not? Explain the rules,
43. Constructaparsetreeforthegivengrammar:S->iCtS S->iCtSeSS->aC->b.
44. Differentiate between right most derivation and left most derivation with anexample.
45. Discuss the following: i) Left Recursion; ii) Left factoring.b) Computefirstandfollowfunctionsforthegivengrammar: E->E+T|TT->T\*F|FF->F\*|a|b.
46. DescribetheErrorrecoveryschemeinYACC

# UNIT-2

1. Differentiatebetweentopdownparsingandbottomup parsing techniques.
2. Constructthetopdownparserusingrecursivedescent parser.
3. WritethealgorithmforconstructionofacanonicalLR parsing table.
4. WhatisanLL(1)grammar?Canyouconverteverycontext free grammar intoLL(1).
5. Computefirstandfollowfunctionsforthegivengrammar: E->E+T|T T->T\*F|F F->F\*|a|b.
6. HowtocheckthegrammarisLL(1)ornot?Explainthe rules,
7. ComputeFIRSTandFOLLOWforthegrammar: S→SS+

\SS\*\a.

1. Designanon-recursivepredictiveparserforthefollowing grammar:
2. S → AaAb | BbBb A → e B → e ,wherea,b,eare terminals.
3. Constructpredictiveparserforthefollowinggrammar:S-

->(L)/a L-->L,S/Sandparseanyinputstring.

1. Writeaboutvarioustypesoftopdownparsing.Discuss about the error recoverin predictive parsing.
2. Writeanalgorithmforconstructingapredictiveparsingtable. Give Example
3. Constructpredictiveparsetableforthefollowinggrammar.
4. E→E+T/TT→T\*F/FF→F/a/b
5. Consider the following grammar E → T + E|T T→V\*T|V V→idWritedowntheproceduresforthenon-terminalsof the grammar to make arecursive descent parser
6. Whatarethepreprocessingstepsrequiredforconstructing Predictive parsingtable. Explain with example.
7. Define a Parser. What is the role of grammars in Parser construction?ConstructthePredictiveparsingtableforthe grammar G: E → E+T |T, E →T\*F |F, F →(E) |id.
8. ExplainthestructureoftheLRParsersandDifference between LR and LLParsers.
9. ListandexplaindifferenttypesofLRParsers.Differentiate LR(0) and LR(1)items and their associated parsers.
10. Explaintheparsingtechniqueswithahierarchicaldiagram. Write Recursive Decent parser for the grammar S→cAd, A→ab|a.
11. DifferentiateTopDownparsingandBottomUp Parsing.
12. Consider the grammar:E→ E+EE→ E\*E E→id Perform shift reduce parsing of the input string “id1+id2+id3”.
13. DefineLR(k)parser.ExplainthemodelofLRparserand various functionsused in it for parser construction.
14. HowtohandleambiguitythroughLRparsers?Discussabout theDangling–Elseambiguity.
15. ConstructCanonicalLRparsingtableforthefollowing grammar. S→L=R | R L→\*R | id R→L
16. WhatisanLR(0)item?ConstructanSLRparsingtablefor the grammar
17. G:S→L=R|R,L→\*R|id,R→L.IsitSLR(1)grammar?
18. Showthatthefollowinggrammar:S→Aa|bAc|Bc|bBaA→ dB → dIs LR(1) but not LALR(1).
19. DevelopLRparserforthegivengrammarandcheckthe acceptance ofinput string of your own:

R->R+|+R|RR|R\*|(R)|a|b.

1. ConstructaLALRparsingtableforfollowinggrammar:
2. Sʹ->S S->CC C->cC/d.
3. WhatarethelimitationsinSLRparser?Howtheyare rectified in CLR and LALRparsers?
4. WritetherulesusedtoconstructSLRParser. Giveexample.
5. What is an LALR(1) grammar?. Construct LALR parsing table for thefollowing grammar: S→ CC, C → cC , C → c|d .
6. WriteandexplaintheLRParsingalgorithm..
7. Explaingrammarwithanexample.HowLRparserscan handle ambiguity?What is the role of priority and associativity in it?
8. ConstructasetofLR(0)itemsforagrammargivenbelow S->L=R

S->R L->\*R L->id R->L.

1. Differentiatebetweenparsetreeandsyntax tree.

# UNIT-3

1. What is the Syntax Directed Translation (SDT)? How attributedgrammarsareusedtogenerateintermediatecode for expressions using SDTs?
2. What are different intermediate code forms? Discuss differentThreeAddresscodetypesandimplementationsof Three Address statements.
3. Writeanoteonsimpletypecheckerandlistthedifferent types of typechecking.
4. What do you mean by attributed grammars? Discuss the translationschemeforConvertinganinfixexpressiontoits equivalent postfix form.
5. Givesyntaxdirectedtranslationschemeforsimpledesk circulator
6. Generate the three address code for the following code fragment.a = b + 1 x = y + 3 y = a / b a=b+c.
7. Givethevariousrepresentationsof3addresscodefor, A=B+C\*(D/E)-(F+6)\*9
8. Writeatranslationalschemetoperformtypecheckingof statements.
9. Differentiatebetweenintermediatecodeandcode generation.
10. Explainthedifferentintermediatecodegeneration techniques.
11. Whatdoyoumeantbyabstracttranslationscheme? Discuss about its structure.
12. ListtheproductionandsemanticactionforBoolean expressions using abstracttranslation scheme.
13. Writethesyntax-directedtranslationschemetoconstruct syntax trees.
14. WhatismeantbyBackpatching?Whatare functionsofit?
15. Writeandexplainsemanticrulesforflow-of-control statements likeif-then, do-while and switch-case.
16. Convertthefollowingexpressionintosyntaxtreeandthree addresscode: h=(b\*-(a+b)/d)-c+6.
17. IllustratehowtheSDTschemeisusedforassignment statements to generate the intermediate code
18. Writetheshortnoteon:(i)Abstractsyntaxtree(ii)Polish notation(iii) Three address code(iv) Back patching
19. Writequadruples,triplesandindirecttriplesforthe expression:-(a\*b)+(c+d)-(a+b+c+d)

# UNIT-4

1. Explainvariousstorageallocationstrategieswithitsmerits and demerits
2. Defineactivationrecords.Explainhowitisrelatedwith runtime storageallocation.
3. Whatisruntimestack?Explainthestorageallocation strategies used forrecursive procedure calls.
4. Whatisaflowgraph?Explainhowflowgraphcanbe constructed for a givenprogram.
5. Main()

{

intsum,n,i; sum=0;

for i:=1 to n do sum:=sum+i;

write(sum);

}

1. Whataretheprinciplesassociatedwithdesigningcalling sequences and thelayout of activation records?
2. WhatistheroleofcodeOptimizerincompiler?Isita mandatory phase?Explain the various sources of optimization.
3. Explainhowdataflowequationsaresetupandsolvedfor improving code.
4. Discussbasicblocksandflowgraphswithan example.
5. Givethegeneralstructureofanactivationrecord?Explain the purpose ofeach component involved in it.
6. Explainvariousmachineindependentcodeoptimization techniques.
7. Writeashortnoteonpeepholeoptimizationandvarious operations used in it.
8. DescribeLoopunrolling?Describeitsadvantagewithyour own examples.
9. Explainstaticandstackstorage allocations?
10. Translatethearithmeticexpressiona[i]=b\*c-b\*dintoasyntax tree, quadruplesand triples.
11. Write pseudocode for finding sum of ‘n’ numbers. And identifybasicblocksthenconstructtheflowgraphforit. Explain the rules used for this.
12. Explainthefollowingpeepholeoptimizationtechniques;
    1. EliminationofRedundantCode
    2. EliminationofUnreachableCode
13. Listthedifferentstorageallocationstrategiesandexplain them.
14. Explainindetailasimplestackallocationscheme.
15. Whatis a DAG? Mention its applications.
16. Explainthecharacteristicsofpeepholeoptimization
17. WritethealgorithmtoconstructflowgraphforfindingSumof ‘N’ natural numbers.
18. Defineinductionvariables.Illustrateeliminationofinduction variable with anexample.
19. Illustratehowamachinemodelaffectscodegeneration?
20. ExplainthebasicfunctionalitiesofIN,OUTandKILLindata flow analysis.
21. Whycodeoptimizationisrequired?Discussaboutdifferent types of codeoptimization techniques.
22. Whatisaflowgraph?Describewhereitis used?
23. Explainthevariousoptimizationtechniquesusedfor peephole.
24. Whatiscodeoptimization?Comparemachinedependent andindependent code optimization techniques.
25. Generatethecontrolflowforifandwhilestatementsusing Boolean expression.
26. Writeshortnoteson:i)Instruction Scheduling;

ii)EliminationofLoopinvariantvariable.

1. Explaintheequationforidentifyinglivevariablesinagiven flowgraph with data flow analysis.
2. Illustratethestorageorganizationmemoryintheperspective ofcompiler writer with neat diagram.
3. DistinguishbetweenStaticandDynamicstorageallocation.
4. Whatisactivationrecord?Writethevariousfieldsof ActivationRecord.
5. Writethedefinitionofsymboltableandproceduretostore thenames in symbol table.
6. Whatissymboltable?Explainhowthehashtableisusedto construct a symbol table.
7. What are the reasons to use intermediate code in a compiler?Writetheintermediatecodefortheexpression

a+a\*(b+c)\*d.

1. Whatarethecontentsofasymboltable?Explainindetail aboutthe symbol table organization for block structured languages.
2. Writedownthetranslationprocedureforcontrolstatement
3. Whatarebasicblocks?Writethealgorithmforpartitioning intoBlocks.

# UNIT-5

1. Whatisaninductionvariable,invariantvariable,deadcode? Explain with anexample.
2. Discuss Global Register Allocation in code generation. Give anexampletoshowhowDAGisusedforregisterallocation.
3. GeneratecodeforthefollowingCstatements: i)x=f(a)+f(a)

ii)y=x/5;

1. GeneratecodeforthefollowingCprogramusinganycode generationalgorithm.

main() { int I;

int a[10]; while(i<=10)

a[i]=0;

}

1. Explainthemainissuesincodegeneration.Howtohandle them? Discuss.
2. Discussaboutregisterallocationandassignmentintarget code generation.
3. Explain the following terms: i)RegisterDescriptorii) Address Descriptor iii) Instruction Costs
4. Discusshowinductionvariablescanbedetectedand eliminated from the givenintermediate code

B2: i:= i+1

t1:=4\*j t2:=a[t1]

if t2<10 goto B2

1. Explainthecodegenerationalgorithmindetailwithan example.
2. Discussbasicblocksandflowgraphswithan example
3. Generatecodeforthefollowing:
   1. x=f(a)+f(a)+f(a) ii)x=f(f(a)) iii)x=++f(a) iv)x=f(a)/g(b)
4. WriteaboutRegisterAllocationandAssignment.
5. Differentiatebetweenregisterdescriptorsandaddress descriptors.
6. Whataretheissuestobeconsideredwhilegenerating code? Explainwith code generation algorithm.
7. Writeaboutallissuesincodegeneration.Describe it.
8. Explaindivisionoftasksbetweencallerandcalleeinstack allocationscheme.
9. Comparestaticversusdynamicmemory allocation.
10. Describeindetailaboutstackallocationofspaceandheap management
11. ExplainStorageallocationstrategieswithsuitableexamples?
12. Explainthesimplecodegeneratorandgeneratetargetcode sequence for the following statement d:=(a-b)+(a-c)+(a-c).