**Written viva question bank based on the program list you provided from your Compiler Design Lab Exercise :**

**1. Symbol Table Implementation**

1. What is a symbol table? What data structure is best suited for implementing it?
2. How does the symbol table handle scope and redeclaration?
3. What operations can be performed on a symbol table?
4. How do you differentiate between global and local symbols in your implementation?
5. What are the typical entries in a symbol table?

**2. Lexical Analyzer for Patterns in C**

1. What patterns did you recognize in your lexical analyzer (keywords, identifiers, etc.)?
2. What is the role of regular expressions in lexical analysis?
3. How did you handle whitespace and comments in your analyzer?
4. What is the difference between token, lexeme, and pattern?
5. How do you detect and report invalid tokens?

**3. Lexical Analysis using LEX**

1. What are the sections of a LEX program?
2. Explain how LEX works internally to generate the lexical analyzer.
3. What is yytext, yyleng, and yylex() in LEX?
4. How does LEX handle precedence when multiple rules can match?
5. How do you integrate error handling in a LEX program?

**4. YACC-Based Programs**

**a) Arithmetic Expression Recognition**

1. What is the grammar used to recognize arithmetic expressions?
2. How do you define operator precedence and associativity in YACC?
3. What is a shift/reduce conflict, and how do you resolve it in this program?
4. What role does the %left, %right, and %nonassoc directive play?

**b) Valid Variable Name Recognition**

1. What are the rules for valid variable names in C?
2. How did you represent this grammar in YACC?
3. How does YACC use LEX-generated tokens in the parsing phase?

**c) Calculator Using LEX and YACC**

1. How are expressions evaluated in your calculator program?
2. How did you manage multiple expressions or line-by-line evaluation?
3. What are the tokens and non-terminals used in your calculator?
4. How does the calculator handle invalid inputs?

**5. Convert BNF to YACC & Generate Abstract Syntax Tree**

1. What is the difference between BNF and YACC grammar representation?
2. What data structure did you use to implement the Abstract Syntax Tree?
3. How is an AST different from a parse tree?
4. Why is AST generation important in a compiler?
5. How did you traverse or evaluate the AST?

**6. Type Checking**

1. What is type checking, and why is it necessary?
2. What kinds of type errors can be detected at compile time?
3. How did you implement type rules in your program?
4. What is the difference between static and dynamic type checking?
5. How do you handle type conversion (implicit/explicit)?

**7. Control Flow & Data Flow Analysis**

1. What is control flow analysis? How is it represented?
2. What is a basic block? What is a control flow graph (CFG)?
3. What is data flow analysis? Give examples like reaching definitions, live variable analysis.
4. What are use-def chains?
5. How do you implement and test for data flow problems in code?

**8. Storage Allocation Strategies (Heap, Stack, Static)**

1. What are the differences between stack, heap, and static memory allocation?
2. Which data structure is used to implement the stack allocation model?
3. What are the advantages and disadvantages of heap allocation?
4. How is memory allocated and freed in heap-based allocation?
5. Explain storage allocation with a code example.

**9. DAG (Directed Acyclic Graph) Construction**

1. What is the purpose of DAG in compiler optimization?
2. How do you construct a DAG for an expression?
3. How does DAG help in eliminating common subexpressions?
4. What data structures are used in your DAG implementation?
5. What is the difference between a DAG and a parse tree?

**10. Back-End Code Generation (8086 Assembly from 3AC)**

1. What is Three-Address Code (TAC)? Give an example.
2. How do you convert TAC to 8086 assembly code?
3. What are the challenges in generating target-specific code?
4. How do you handle registers and memory access in 8086?
5. What are the steps in assembling and running the generated code?