Compiler Concepts - 60 MCQ Questions with Answers

Questions 1-10: Compiler Fundamentals

1. What is the primary purpose of a compiler? a) Execute programs directly b) Translate source language to target language c) Debug program errors d) Manage memory allocation

Answer: b) Translate source language to target language

---

2. Which of the following is NOT an advantage of high-level languages? a) Closer to human reasoning b) Faster development c) Direct hardware control d) Error detection

Answer: c) Direct hardware control

---

3. What type of input does a compiler process? a) Single instruction b) Entire program c) Machine code d) Assembly language

Answer: b) Entire program

---

4. Which language generation does C belong to? a) 1GL b) 2GL c) 3GL d) 4GL

Answer: c) 3GL

---

5. What is the main advantage of using a compiler over an interpreter? a) Easier debugging b) Faster execution c) Line-by-line error detection d) Interactive development

Answer: b) Faster execution

---

6. Which of the following is an example of a hybrid approach? a) C++ b) Python c) Java d) Assembly

Answer: c) Java

---

7. What does portability mean in the context of compilers? a) Programs can run on any hardware b) Same source code can generate multiple machine languages c) Code can be easily moved between folders d) Programs are lightweight

Answer: b) Same source code can generate multiple machine languages

---

8. Which language generation represents machine code? a) 1GL b) 2GL c) 3GL d) 4GL

Answer: a) 1GL

---

9. What is the main disadvantage of interpreters compared to compilers? a) Harder debugging b) Slower execution c) No error detection d) Complex syntax

Answer: b) Slower execution

---

10. Which of the following is a 4GL language? a) C b) Java c) SQL d) Assembly

Answer: c) SQL

Questions 11-20: Compiler Architecture & Phases

11. What is the first phase of compilation? a) Syntax Analysis b) Lexical Analysis c) Semantic Analysis d) Code Generation

Answer: b) Lexical Analysis

---

12. Which phase detects the error in the symbol `@` in C code? a) Lexical Analysis b) Syntax Analysis c) Semantic Analysis d) Code Generation

Answer: a) Lexical Analysis

---

13. What does the syntax analyzer (parser) create? a) Symbol table b) Tokens c) Parse tree d) Machine code

Answer: c) Parse tree

---

14. Which phase validates type compatibility (e.g., int + string)? a) Lexical Analysis b) Syntax Analysis c) Semantic Analysis d) Code Optimization

Answer: c) Semantic Analysis

---

15. What is the purpose of the Intermediate Code Generator? a) Generate machine code b) Produce machine-independent IR c) Optimize code d) Parse tokens

Answer: b) Produce machine-independent IR

---

16. Which phase is optional in the compilation process? a) Lexical Analysis b) Syntax Analysis c) Code Optimizer d) Code Generator

Answer: c) Code Optimizer

---

17. What does the symbol table store? a) Machine instructions b) Tokens c) Identifiers, types, and scopes d) Parse trees

Answer: c) Identifiers, types, and scopes

---

18. Which phase is machine-dependent? a) Lexical Analysis b) Semantic Analysis c) Code Optimization d) Code Generation

Answer: d) Code Generation

---

19. What is the output of the lexical analyzer for `int a = 5;`? a) Parse tree b) Tokens: int, a, =, 5, ; c) Symbol table d) Machine code

Answer: b) Tokens: int, a, =, 5, ;

---

20. Which phases constitute the "Front End" of a compiler? a) Lexical to Semantic Analysis b) Code Optimization to Code Generation c) Only Lexical Analysis d) Only Code Generation

Answer: a) Lexical to Semantic Analysis

Questions 21-30: Compilation Toolchain

21. What does the preprocessor handle? a) Syntax checking b) #include and #define directives c) Code optimization d) Linking libraries

Answer: b) #include and #define directives

---

22. Which tool converts assembly code to object code? a) Preprocessor b) Compiler c) Assembler d) Linker

Answer: c) Assembler

---

23. What is the output of the assembler? a) Assembly code b) Relocatable object code (.o) c) Executable file d) Preprocessed code

Answer: b) Relocatable object code (.o)

---

24. Which tool combines object files and libraries? a) Preprocessor b) Compiler c) Assembler d) Linker

Answer: d) Linker

---

25. What command is used to see preprocessed output in gcc? a) gcc -S file.c b) gcc -E file.c c) gcc -c file.c d) gcc -o file.c

Answer: b) gcc -E file.c

---

26. What does `gcc -S file.c` produce? a) Preprocessed code b) Assembly code c) Object code d) Executable

Answer: b) Assembly code

---

27. Which stage resolves external references? a) Preprocessing b) Compilation c) Assembly d) Linking

Answer: d) Linking

---

28. What is the typical output filename of the linker? a) file.s b) file.o c) a.out d) file.i

Answer: c) a.out

---

29. Which tool is used for assembly in the toolchain? a) gcc b) as c) ld d) cpp

Answer: b) as

---

30. What type of code does the compiler generate? a) Machine code b) Assembly code c) Object code d) Preprocessed code

Answer: b) Assembly code

Questions 31-40: Error Types & Interpretation

31. Which error type occurs during lexical analysis? a) Missing semicolon b) Type mismatch c) Invalid symbol ($) d) Division by zero

Answer: c) Invalid symbol ($)

---

32. What type of error is "missing } in C code"? a) Lexical error b) Syntax error c) Semantic error d) Runtime error

Answer: b) Syntax error

---

33. Which error occurs during program execution? a) Lexical error b) Syntax error c) Semantic error d) Runtime error

Answer: d) Runtime error

---

34. What does REPL stand for? a) Read-Execute-Print-Loop b) Read-Eval-Print-Loop c) Run-Execute-Parse-Loop d) Read-Evaluate-Process-Loop

Answer: b) Read-Eval-Print-Loop

---

35. In the REPL cycle, what happens in the "Read" phase? a) Execute the code b) Parse statement to AST c) Print output d) Loop back

Answer: b) Parse statement to AST

---

36. Which phase in REPL computes expressions? a) Read b) Evaluate c) Print d) Loop

Answer: b) Evaluate

---

37. What is an example of a semantic error? a) Missing semicolon b) Invalid character c) Type mismatch (int + string) d) Division by zero

Answer: c) Type mismatch (int + string)

---

38. Which languages commonly use REPL? a) C and C++ b) Python and JavaScript c) Assembly d) Machine code

Answer: b) Python and JavaScript

---

39. When do runtime errors occur? a) During lexical analysis b) During syntax analysis c) During semantic analysis d) During program execution

Answer: d) During program execution

---

40. What is created during the "Read" phase of REPL? a) Tokens b) Machine code c) Abstract Syntax Tree (AST) d) Symbol table

Answer: c) Abstract Syntax Tree (AST)

Questions 41-50: Formal Grammar

41. What does a formal grammar define? a) Only syntax rules b) Alphabet, lexicon, and syntax c) Only production rules d) Only terminals

Answer: b) Alphabet, lexicon, and syntax

---

42. In grammar notation G = (N, Σ, S, P), what does Σ represent? a) Non-terminals b) Terminals c) Start symbol d) Production rules

Answer: b) Terminals

---

43. What are production rules? a) Rules for executing code b) Rules for transforming strings by replacing symbols c) Rules for error handling d) Rules for memory management

Answer: b) Rules for transforming strings by replacing symbols

---

44. Which symbols can be replaced during derivation? a) Terminals b) Non-terminals c) Both terminals and non-terminals d) Neither terminals nor non-terminals

Answer: b) Non-terminals

---

45. What does the empty string ε represent in grammar? a) Error symbol b) Start symbol c) Rule that removes a symbol d) End symbol

Answer: c) Rule that removes a symbol

---

46. For the grammar with rules S → aSb and S → ba, what pattern does the language follow? a) a^n b^n b) a^n b a b^n c) (ab)^n d) a^n b^m

Answer: b) a^n b a b^n

---

47. What makes a grammar ambiguous? a) Having multiple non-terminals b) Having multiple terminals c) A string can be generated in multiple ways d) Having no production rules

Answer: c) A string can be generated in multiple ways

---

48. What is the role of the start symbol in grammar? a) It's the final symbol in derivation b) It's the initial non-terminal from which derivation begins c) It represents empty string d) It's used only in production rules

Answer: b) It's the initial non-terminal from which derivation begins

---

49. In the derivation S → aSb → aaSbb → aababb, what is the value of n? a) 1 b) 2 c) 3 d) 4

Answer: b) 2

---

50. What acts as a "divider" in the pattern a^n b a b^n? a) The first 'a' b) The last 'b' c) The central 'ba' d) The entire string

Answer: c) The central 'ba'

Questions 51-60: Advanced Concepts

51. What is the main difference between single-pass and multi-pass compilers? a) Single-pass is faster b) Single-pass requires "define-before-use" c) Multi-pass generates better code d) Multi-pass uses less memory

Answer: b) Single-pass requires "define-before-use"

---

52. Which is an example of a single-pass compiler requirement? a) Java b) C++ c) Pascal d) Python

Answer: c) Pascal

---

53. What is the purpose of a recognizer in formal language theory? a) Generate strings b) Check if a string belongs to the language c) Create production rules d) Define grammar

Answer: b) Check if a string belongs to the language

---

54. According to automata theory, what is true about recognizers? a) All languages can be recognized b) Some languages cannot have a recognizer c) Only formal languages can be recognized d) Recognizers are always efficient

Answer: b) Some languages cannot have a recognizer

---

55. What type of analysis does the "Back End" of a compiler perform? a) Language-specific analysis b) Machine-independent synthesis c) Machine-dependent synthesis d) Lexical analysis

Answer: c) Machine-dependent synthesis

---

56. Which tool is used for parser generation? a) Lex b) Yacc c) gcc d) as

Answer: b) Yacc

---

57. What is the main purpose of studying compiler construction? a) Only to build compilers b) To integrate algorithms, formal languages, and computer architecture c) Only for academic purposes d) To understand machine code

Answer: b) To integrate algorithms, formal languages, and computer architecture

---

58. In operator precedence, which operators are evaluated first in the expression `a + b * c / d`? a) + and - b) * and / c) All have equal precedence d) Left to right evaluation

Answer: b) * and /

---

59. What does IR stand for in compiler terminology? a) Instruction Register b) Intermediate Representation c) Internal Reference d) Input/Output Reference

Answer: b) Intermediate Representation

---

60. Which phase maintains the symbol table? a) Lexical Analysis b) Syntax Analysis c) Semantic Analysis d) Code Generation

Answer: c) Semantic Analysis

Answer Key Summary

1-10: b, c, b, c, b, c, b, a, b, c **11-20:** b, a, c, c, b, c, c, d, b, a **21-30:** b, c, b, d, b, b, d, c, b, b **31-40:** c, b, d, b, b, b, c, b, d, c **41-50:** b, b, b, b, c, b, c, b, b, c **51-60:** b, c, b, b, c, b, b, b, b, c