Reg. No.															

Question Paper Code	12984

B.E. / B.Tech. - DEGREE EXAMINATIONS, NOV / DEC 2024

Sixth Semester

Computer Science and Engineering

20CSPC602 - COMPILER DESIGN

Regulations - 2020

Duration: 3 Hours | Max. Marks: 100

PART - A (MCQ) (20 × 1 = 20 Marks)

Answer ALL Questions

		Marks	K – Level	CO
1.	What is the output of a lexical analyzer? (a) Syntax tree (b) Intermediate code (c) Tokens (d) Machine code	1	K1	CO1
2.	During which phase does the compiler check for syntax errors in the code? (a) Lexical analysis (b) Semantic analysis (c) Syntax analysis (d) Intermediate code generation	1	K1	CO1
3.	Suppose a compiler uses LEX and YACC in combination for analysis. LEX would handle ___, while YACC would handle ___. (a) Token recognition; syntax parsing (b) Syntax parsing; semantic analysis (c) Code generation; code optimization (d) Code optimization; lexical analysis	1	K1	CO1
4.	The number of tokens in the following C statement is printf("i = %d, &i = %x", i, &i); (a) 3 (b) 26 (c) 10 (d) 21	1	K2	CO2
5.	In a compiler, keywords of a language are recognized during (a) parsing of the program (b) the code generation (c) the lexical analysis of the program (d) dataflow analysis	1	K1	CO2
6.	In a two-pass assembler, symbol table is (a) Generated in first pass (b) Generated in second pass (c) Not generated at all (d) Generated and used only in second pass	1	K1	CO2
7.	The grammar A → AA | (A) | ε is not suitable for predictive-parsing because the grammar is? (a) ambiguous (b) left-recursive (c) right-recursive (d) A and B	1	K2	CO3
8.	If a state does not know whether it will make a shift operation or reduction for a terminal is called? (a) Shift/reduce conflict (b) Reduce /shift conflict (c) Shift conflict (d) Reduce conflict	1	K1	CO3
9.	Which one of the following is a top-down parser? (a) Recursive descent parser (b) Operator precedence parser (c) An LR(k) parser (d) An LALR(k) parser	1	K1	CO3
10.	In a compiler's intermediate code, procedure calls typically use an "activation record." This record generally includes: (a) The syntax tree of the procedure (b) The return address and local variables (c) Only the parameters of the function (d) The entire code of the procedure	1	K1	CO4
11.	The semantic analyzer uses which of the following structures to store information about identifiers? (a) Control flow graph (b) Syntax tree (c) Symbol table (d) Intermediate code	1	K1	CO4
12.	Which of the following is NOT a common form of intermediate representation (IR)? (a) Three-address code (b) Control flow graph (CFG) (c) Assembly language (d) High-level programming language	1	K1	CO4
13.	Which of the following is NOT a component of the run-time environment? (a) Stack (b) Heap (c) Intermediate code (d) Static data area	1	K1	CO5
14.	In a run-time environment, which of the following is typically stored on the stack? (a) Global variables (b) Static data (c) Local variables (d) Heap memory	1	K1	CO5

15.	The graph that shows basic blocks and their successor relationship is called: (a) DAG (b) Control graph (c) Flow graph (d) Hamiltonian graph	*1*	*K1*	*CO5*
16.	In a basic block's DAG, a node typically represents: (a) A variable declaration (b) An individual operation or subexpression (c) A storage location (d) A data type	*1*	*K1*	*CO5*
17.	The optimization which avoids test at every iteration is? (a) Loop unrolling (b) Loop jamming (c) Constant folding (d) None of the mentioned	*1*	*K1*	*CO6*
18.	Substitution of values for names (whose values are constants) is done in (a) Local optimization (b)Loop optimization (c)Constant folding (d) Strength reduction	*1*	*K1*	*CO6*
19.	Dead-code elimination in machine code optimization refers to: (a) Removal of all labels (b) Removal of values that never get used (c) Removal of function which are not involved (d) Removal of a module after its use	*1*	*K1*	*CO6*
20.	When performing DAG optimization, what happens to identical subexpressions in the graph? (a) They are duplicated to avoid confusion (b) They are merged to reduce redundancy (c) They are ignored completely (d) They are converted into temporary variables	*1*	*K1*	*CO6*

PART - B (10 × 2 = 20 Marks)

Answer ALL Questions

21.	Give the significance of symbol table. Draw a sample table.	*2*	*K2*	*CO1*
22.	Differentiate tokens, patterns and lexeme.	*2*	*K2*	*CO1*
23.	List the errors that can occur at different stages of compiler.	*2*	*K1*	*CO2*
24.	How and why input buffering is occurring?	*2*	*K1*	*CO2*
25.	Write down the CFG for representing the if-else statement of any language.	*2*	*K2*	*CO3*
26.	Define YAAC.	*2*	*K1*	*CO3*
27.	State the rules of type checking.	*2*	*K1*	*CO4*
28.	Define back patching.	*2*	*K1*	*CO4*
29.	What are two common strategies used for allocating memory dynamically?	*2*	*K1*	*CO5*
30.	Define Peep-hole Optimization.	*2*	*K1*	*CO6*

PART - C (6 × 10 = 60 Marks)

Answer ALL Questions

31.	a)	Explain the phases of compiler in detail. Illustrate the output of each phase of compilation for the input a = (b+c)* (b+c)*2.	*10*	*K2*	*CO1*
		OR			
	b) i)	Explain Compiler Construction Tools.	*5*	*K2*	*CO1*
	ii)	Describe the need for grouping of phases.	*5*	*K2*	*CO1*
32.	a) i)	Organize the Role of Lexical Analyzer with an example.	*5*	*K3*	*CO2*

	ii)	Divide the following C++ program into lexemes. Which lexemes should have associated lexical values, and what should those values represent?
		float limitedSquare(x) {
		/* returns x-squared, but never more than 100 */
		return (x <= -10.0 || x == 10.0) ? 100 : x * x;
		}
			5	*K3*	*CO2*

		OR			
	b)	Construct minimized DFA for the regular expression (a/b)* abb using direct method.	*10*	*K3*	*CO2*

33. a) Construct a predictive parsing table for the grammar *10 K3 CO3*
S→ (L) | a
L→L, S| S.
and show whether the following string will be accepted or not. (a,(a,(a,a))).

OR

b) Construct the SLR parsing table for the following Grammar: *10 K3 CO3*
E→E+T | T
T→TF | F
F→ F* | a | b.

34. a) Generate Three address codes for the following piece of code and hence write the Syntax Directed Translation. *10 K3 CO4*

```
while (a<b and c>b)
do
if c <d
then
      x := y + z
      break
else
   x:= y - z
```

OR

b) Generate Three address code and apply back patching for the given expression *10 K3 CO4*

```
if(a > 0 && b < 1 || c!=0)
  x = a+b+c;
else
  x = 1;
```

35. a) i) Discuss the various issues in design of Code Generator. *5 K2 CO5*

ii) Write the code generation algorithm. Explain the process of register allocation and assignment. *5 K2 CO5*

OR

b) Elaborate on Storage Organization and Allocation Strategies in detail. *10 K2 CO5*

36. a) Illustrate the Principal Sources of Optimization with example. *10 K2 CO6*

OR

b) i) Explain in detail about Optimization of basic blocks. *7 K2 CO6*

ii) Consider the basic block given below. *3 K2 CO6*

```
a=b+c
c = a + d
d = b + c
e=d-b
a=e+b
```

Find out the minimum number of nodes and edges present in the DAG representation of the above basic block.