Name: **Enrolment No:**	UPES UNIVERSITY OF TOMORROW

UNIVERSITY OF PETROLEUM AND ENERGY STUDIES
End Semester Examination, December 2022

Course: Compiler Design **Semester: V**
Program: BTech CSE (All Branches) **Time : 03 hrs.**
Course Code: CSEG3015 **Max. Marks: 100**

Instructions: Attempt all Questions

S. No.		Marks	CO
	SECTION A **(5Qx4M=20Marks)**		
Q 1	Give the syntax-directed definition for if-else statement.	**4**	**CO1**
Q 2	Discuss the error handling mechanism in Parsing Phase.	**4**	**CO2**
Q 3	Draw a DAG for the expression a + a*(b-c) +(b-c) *d	**4**	**CO3**
Q 4	Mention the issues to be considered while applying the techniques for code optimization.	**4**	**CO4**
Q 5	Illustrate the properties of optimizing compiler.	**4**	**CO1**
	SECTION B **(4Qx10M= 40 Marks)**		
Q 6	Construct SLR (1) Parsing table for the given grammar. S → P P → (P)P P → ε Also identify at least one entry in the Parsing table that would be a Shift/Reduce or Reduce/Reduce Conflict.	**10**	**CO2**
Q 7	Give a CFG for the language $\{0^i 1^j 2^k \mid i+j > 2k\}$	**10**	**CO3**
	OR		
Q 7	Consider the CFG given below: - S → EN E → E + T | E – T | T T → T * F | T / F | F F → (E) | digit N → ;	**10**	**CO3**

	(a) Obtain the Syntax Directed Definition for the above grammar. (b) Construct the annotated parse tree for the input string *5*6 +7*.		
Q 8	Demonstrate activation trees. Construct an activation tree for quick sort.	**10**	**CO4**
Q 9	Programming languages are usually compiled into the machine code of the target computer, but sometimes an interpretive system is used. Discuss the relative merits of these two approaches. Outline the key features of the design of an interpretive code that would be suitable for an implementation of the C programming language and describe the overall structure of an interpreter for it.	**10**	**CO1**
SECTION-C **(2Qx20M=40 Marks)**			
Q 10	Suppose that we want to describe Java style class declarations like these using a grammar: class Car extends Vehicle public class JavaIsCrazy implements Factory, Builder, Listener public final class President extends Person implements Official One such grammar for this is (1) C → P F class identifier X Y (2) P → public (3) P → ε (4) F → final (5) F → ε (6) X → extends identifier (7) X → ε (8) Y → implements I (9) Y → ε (10) I → identifier J (11) J →, I (note the comma before the I) (12) J → ε Construct an LL (1) parser table for this grammar. When indicating which productions should be performed, use the numbering system from above.	**20**	**CO2**
Q 11	Construct SLR (1) Parsing table for the grammar given below: S → R R → RR R → R | R (note that | is a terminal symbol in the grammar) R → R* R → ε (note that ε is a terminal symbol in the grammar)	**20**	**CO3**

	R → a R → (R) Resolve the conflicts (if any) by using precedence rules for regular expressions. LR (1) is a much stronger parsing algorithm than SLR (1). Would using an LR (1) parser instead of the SLR (1) parser resolve the ambiguities? Why or why not?		
	OR		

Q 11	Identify the basic blocks, draw a flow graph, and identify the loop invariant statement for the following code snippet.
	
	for (i = 1 to n)
	{ a[i] =0;
	}
		20	**CO3**