PHD ADMISSION WRITTEN ENTRANCE EXAMINATION

Computer Science and Engineering

#	Subject Area	Core Modules
(i)	**Discrete Mathematics**	Logic, Sets, Relations, Functions, Partial Orders, Lattices, Algebraic Structures, Combinatorics, and Graph Theory
(ii)	**Data Structures and Algorithms**	Asymptotic Analysis, Linear & Non-Linear ADTs, Heaps, Hashing, Searching, Sorting, and Algorithm Design Paradigms
(iii)	**C Programming**	Data Types, Control Flow, Functions & Recursion, Arrays, Pointers, Structures, Dynamic Memory, and File I/O
(iv)	**Computer Organization**	Number Systems, Fixed/Floating-Point Representation, Binary Arithmetic, ALU, and Cache Memory Architecture

(i) Discrete Mathematics

- **Propositional and First-Order Predicate Logic:** Logical connectives, truth tables, tautologies, contradictions, logical equivalences, rules of inference, valid arguments, universal and existential quantifiers, nested quantifiers, Prenex Normal Form (PNF), and proof techniques (direct proof, proof by contradiction, and mathematical induction).
- **Sets, Relations, and Functions:** Set representations, subsets, power sets, Cartesian products, set operations, Venn diagrams, set identities, and principle of inclusion-exclusion; binary relations, matrix and digraph representations, properties of relations (reflexive, symmetric, antisymmetric, transitive), closures, equivalence relations, equivalence classes, and partitions; types of functions (injective, surjective, bijective), inverse functions, composition, and floor/ceiling functions.
- **Partial Orders, Lattices, and Boolean Algebra:** Partial orderings (posets), Hasse diagrams, maximal/minimal and greatest/least elements, lattices (bounded, complemented, and distributive lattices); Boolean algebra axioms, canonical forms (SOP and POS), minterms, maxterms, and functional completeness.
- **Algebraic Structures (Group Theory):** Semi-groups, monoids, groups, subgroups, cyclic groups, Abelian groups, cosets, Lagrange's theorem, and group homomorphism and isomorphism.
- **Combinatorics:** Basic counting principles (sum and product rules), permutations and combinations (with and without repetition), pigeonhole principle, binomial theorem, recurrence relations (linear homogeneous and non-homogeneous), and generating functions.
- **Graph Theory:** Undirected and directed graphs, multigraphs, degree of vertices, handshaking theorem, special graphs (complete, bipartite, regular), adjacency matrix and adjacency list representations, graph isomorphism, graph connectivity (cut-vertices, bridges, vertex/edge connectivity), Eulerian and Hamiltonian paths and circuits, trees and spanning trees (Cayley's formula), matching and covering (vertex cover, independent sets, bipartite matching), planar graphs (Euler's formula), and graph coloring.

(ii) Data Structures and Algorithms

- **Asymptotic Analysis & Recurrences:** Time and space complexity, asymptotic notations (Big-O, Big-Ω, Big-Θ, small-o, small-ω), worst-case, average-case, and amortized analysis, and solving recurrence relations (substitution, recursion-tree, and Master theorem).
- **Linear Abstract Data Types (ADTs):** Arrays (1D and multidimensional row/column-major representation) and Linked Lists (singly, doubly, and circular linked lists; insertion, deletion, reversal, cycle detection, and traversal); Stacks (array and linked implementations, expression evaluation, infix-to-postfix/prefix conversion, recursion and call stack); Queues (linear queue, circular queue, double-ended queue / deque, and applications).
- **Priority Queue ADT & Binary Heaps:** Min-heap and max-heap properties, array representation of binary heaps, heapify operations, insertion, extract-min/extract-max, decrease/increase-key, and O(n) heap construction.
- **Non-Linear ADTs - Trees:** General trees, binary trees, strict/complete/full binary trees, tree traversals (inorder, preorder, postorder, level-order) and tree reconstruction from traversals, Binary Search Trees (search, insertion, deletion, inorder successor/predecessor), balanced search trees (AVL trees and rotations, B-trees and B+ trees basics).
- **Non-Linear ADTs - Graphs:** Graph traversal algorithms (Breadth-First Search and Depth-First Search, edge classification), topological sorting, strongly connected components, Minimum Spanning Trees (Prim's and Kruskal's algorithms), and shortest path algorithms (Dijkstra's, Bellman-Ford, and Floyd-Warshall algorithms).
- **Searching, Hashing, and Sorting:** Linear search, binary search, interpolation search; hashing (hash functions, load factor, collision resolution via separate chaining and open addressing - linear probing, quadratic probing, double hashing); comparison-based sorting algorithms (bubble, selection, insertion, merge sort, quick sort, and heap sort), non-comparison sorting (counting sort, radix sort), stability, and best, average, and worst-case complexities.
- **Algorithm Design Techniques:** Greedy algorithms (fractional knapsack, Huffman coding, activity selection), Divide and Conquer (Strassen's matrix multiplication, closest pair), and Dynamic Programming (0/1 knapsack, longest common subsequence, matrix chain multiplication).

(iii) C Programming

- **Lexical Elements, Data Types & I/O Statements:** Identifiers, keywords, primitive data types, type modifiers, type promotion and explicit casting, constants, operators, precedence and associativity, sequence points, formatted I/O (`printf`, `scanf`, format specifiers, return values), and unformatted character/string I/O (`getchar`, `putchar`, `fgets`).
- **Selection and Repetition Control Flow:** Conditional execution using `if`, `if-else`, dangling-else resolution, conditional (ternary) operator, and multi-way branching using `switch-case` (fall-through, `break`, `default`); loops (`for`, `while`, `do-while`), nested loops, and jump statements (`break`, `continue`, `goto`).

- **Functions, Recursion & Storage Classes:** Function prototypes, parameter passing mechanisms (call-by-value and simulated call-by-reference), recursion (tail recursion, tree recursion, activation records/stack frames), variable scope, lifetime, and storage classes (`auto`, `static`, `extern`, `register`).
- **Arrays, Strings & Pointers:** 1D and multidimensional arrays, null-terminated character arrays and standard string functions (`strlen`, `strcpy`, `strcmp`, `strcat`), pointer declaration, dereferencing, pointer arithmetic, pointer-to-array vs. array-of-pointers, void pointers, null pointers, dangling pointers, function pointers, and dynamic memory allocation (`malloc`, `calloc`, `realloc`, `free`).
- **Structures, Unions & User-Defined Types:** Defining and initializing `struct` variables, structure padding and alignment, accessing members via dot (`.`) and arrow (`->`) operators, arrays of structures, nested and self-referential structures, pointers to structures, `union`, bit-fields, `enum`, and `typedef`.

(iv) Computer Organization

- **Number Systems & Fixed/Floating-Point Representation:** Positional number systems and base conversions; signed integer representations (sign-magnitude, 1's complement, and 2's complement), sign extension, and representable ranges; IEEE 754 floating-point standard (single-precision 32-bit and double-precision 64-bit formats, sign bit, biased exponent, normalized and denormalized mantissa, rounding modes, and special values: ±0, ±∞, NaN).
- **Binary Arithmetic & ALU Design:** Addition and subtraction in 2's complement, carry/overflow detection flags; Fast adders (ripple-carry adder, carry-lookahead adder); Binary multiplication algorithms (unsigned shift-and-add multiplication, signed multiplication using Booth's algorithm and bit-pair recoding); Binary division algorithms (restoring and non-restoring division methods).
- **Cache Memory Architecture & Hierarchy:** Memory hierarchy (registers, L1/L2/L3 cache, main memory) and principle of locality (temporal and spatial locality); cache mapping techniques (direct mapping, fully associative mapping, and k-way set-associative mapping); physical address breakdown (tag, index/set, and block offset bits) and tag directory size calculation; cache replacement policies (LRU, FIFO, Optimal, Random); write policies (write-through, write-back, dirty bit, write-allocate, no-write-allocate); and cache performance metrics (hit ratio, miss rate, miss penalty, Average Memory Access Time - AMAT, and multi-level cache AMAT).