Advanced Knowledge and Skills
For High School Mathematics
Discrete Mathematics

It is essential that the advanced knowledge and skills for high school mathematics be addressed in instructional contexts that promote *problem solving, reasoning, communication, making connections, designing and analyzing representations,* and *reflecting on solutions*.

D.1 Set Theory: **Operate with sets and use set theory to solve problems.**

D.1.1 Demonstrate understanding of the definitions of set equality, subset and null set.

D.1.2 Perform set operations such as union and intersection, difference, and complement.

D.1.3 Use Venn diagrams to explore relationships and patterns, and to make arguments about relationships between sets.

D.1.4 Demonstrate the ability to create the cross-product or set-theoretic product of two sets.

D.2 Relations and Functions: **Demonstrate understanding of relations and functions.**

D.2.1 Determine whether simple examples of discrete functions are injective and/or surjective.

D.2.2 Demonstrate ability to interpret examples of simple discrete functions by mapping elements from a discrete domain to a discrete range.

D.2.3 Demonstrate ability to produce the subset of the cross product of the domain and image of a relation corresponding to simple examples of relations.

D.2.4 Use concepts of reflexivity, symmetry, and transitivity to establish that a relation is an equivalence relation.

D.2.5 Design simple algorithms such as hashing, checksum or error-correction functions.

D.3 Modular Arithmetic: **Demonstrate understanding of modular arithmetic and its relationship to set theory.**

D.3.1 Perform modular arithmetic operations.

D.3.2 Demonstrate understanding of the relationship of modular arithmetic to two numbers being congruent modulo n.

D.3.3 Solve practical problems or develop algorithms using modular arithmetic or congruence relations such as creating error detection codes, calculating greatest common factor, and solving simple coin-change problems.

D.4 Graph Theory: **Understand how graphs of vertices joined by edges can model relationships and be used to solve a wide variety of problems.**

D.4.1 Use graphs to model and solve problems such as shortest paths, vertex coloring, critical paths, routing, and scheduling problems.

D.4.2 Convert from a graph to an adjacency matrix and vice versa.

D.4.3 Use directed graphs, spanning trees, rooted trees, binary trees, or decision trees to solve problems.

D.4.4 Demonstrate understanding of algorithms such as depth-first and breadth-first walk of a tree or maximal matching.

D.4.5 Use matching or bin-packing techniques to solve optimization and other problems.

D.4.6 Compare and contrast different graph algorithms in terms of efficiency and types of problems that can be solved.

D.5 Combinatorics and Discrete Probability: **Understand and apply fundamental counting techniques in solving combinatorial and probability problems.**

D.5.1 Produce all combinations and permutations of sets.

D.5.2 Calculate the number of combinations and permutations of sets of *m* items taken *n* at a time.

D.5.3 Apply basic fundamental counting principles such as The Pigeonhole Principle, Multiplication Principle, Addition Principle, and Binomial Theorem to practical problems.

D.5.4 Solve probability problems such as conditional probability, probability of simple events, mutually exclusive events, and independent events.

D.5.5 Find the odds that an event will occur given the probabilities and vice versa.

D.6 Sequences and Series: **Analyze and evaluate sequences and series.** [1]

D.6.1 Define, recognize, and discriminate among arithmetic, geometric and other sequences and series.

D.6.2 Find the explicit and recursive formulas for arithmetic and geometric sequences and use these formulas to determine a specific term or term number.

D.6.3 Convert between a series and its sigma notation representation.

D.6.4 Find partial sums of arithmetic and geometric series and find sums of convergent infinite series.

D.6.5 Generate and describe other recursive sequences such as factorials and the Fibonacci sequence.

D.7 Recurrence, Recursion and Induction: **Understand and apply recurrence, recursive, and inductive methods to solve problems.**

D.7.1 Use recursive and iterative thinking to solve problems such as population growth and decline, exponential functions, problems involving sequential change and compound interest.

D.7.2 Use finite differences to solve problems and to find explicit formulas for recurrence relations.

D.7.3 Use mathematical induction to prove recurrence relations and concepts in number theory such as sums of infinite integer series, divisibility statements, and parity statements.

D.7.4 Use mathematical induction to analyze the validity of an iterative algorithm.

D.7.5 Describe arithmetic and geometric sequences recursively.

D.7.6 Use understanding of relationship of finite and infinite geometric series, including how the concept of limits connects them.

---

[1] Included in both Advanced Algebra and Discrete Mathematics

D.7.7 Apply recurrence or recursion to the design and understanding of sorting and searching algorithms.

D.7.8 Analyze algorithms for efficiency, including how the number of steps grows as a function of the size of the problem.

D.7.9 Compare the efficiency of iterative and recursive solutions of a problem.

D.8 Logic: Understand the fundamentals of propositional logic, arguments, and methods of proof.

D.8.1 Use truth tables to determine truth values of compounded propositional statements.

D.8.2 Find the converse, inverse, and contrapositive of a statement.

D.8.3 Determine whether two propositions are logically equivalent.

D.8.4 Identify and give examples of undefined terms, definitions, axioms, and theorems.

D.8.5 Construct logical arguments using laws of detachment (modus ponens), syllogism, tautology, and contradiction; judge the validity of arguments, and give counterexamples to disprove statements.

D.8.6 Use applications of the universal and existential quantifiers to propositional statements.

D.8.7 Appropriately select and use methods of deductive, inductive, and indirect proof and determine whether a short proof is logically valid.

D.9 Social Choice: Analyze election data to evaluate different election methods and use weighted voting techniques to decide voting power within a group. Understand and use fair division techniques to solve apportionment problems.

D.9.1 Use election theory techniques to analyze election data such as majority, plurality, runoff, approval, the Borda method in which points are assigned to preferences, and the Condorcet method in which each pair of candidates is run off head to head.

D.9.2 Use fair division techniques to divide continuous objects.

D.9.3 Use fair division techniques to solve apportionment problems.

D.10 Game Theory: Understand and use game theory methods to solve strictly determined games and non-strictly determined games.

D.10.1 Use game theory to solve strictly determined games.

D.10.2 Use game theory to solve non-strictly determined games.

D.10.3 Use game theory to create models for games.

D.10.4 Use game theory to find optimal mixed strategies such as expected values or payoff values.

D.11 Coding Theory, Compression and Cryptography: Understand coding of alphabets and simple encryption methods.

D.11.1 Use integer functions to encode alphabets and to create error-checking correcting codes.

D.11.2 Use permutations, combinations of digraph encoding and affine transformation and hash functions, to create encryption codes.

D.11.3 Demonstrate understanding of asymmetric public key cryptography algorithms such as RSA and Diffie-Hellman.

D.11.4 Demonstrate understanding of error-detecting and error-correcting codes and data compression through Huffman codes.

D.12 Algorithm Design: Understand methods of algorithm design and its relationship to data structures.

D.12.1 Design algorithms using recurrence or iteration.

D.12.2 Design algorithms using divide-and-conquer.

D.12.3 Design algorithms using recursion.

D.12.4 Evaluate the efficiency of an algorithm including the order of complexity of algorithms.

D.12.5 Demonstrate understanding of the relationship of set theory, relations, functions, combinatorics, sequences, series, graph theory, and matrices to the design of data structures and algorithms.

D.12.6 Demonstrate understanding of the relationship of data structures to the design of algorithms and use this understanding to analyze algorithms.