Virginia Mathematics

Discrete Mathematics

Logical Reasoning	**LR.1** The student will use reasoning to develop and apply logical arguments. DM.LR.1
	LR.2 The student will apply logic and proof techniques in the construction of a sound argument. DM.LR.2
	LR.3 The student will apply Boolean algebra to represent and analyze the function of logical gates and circuits. DM.LR.3
	LR.4 The student will use mathematical induction to prove formulas and mathematical statements. DM.LR.4
Use Venn diagrams to codify and solve logic problems. DM.LR.1.A	**a** Use Venn diagrams to codify and solve logic problems. DM.LR.1.A
Express logical statements in symbolic form. DM.LR.1.B	**b** Express logical statements in symbolic form. DM.LR.1.B
Represent a conditional statement as its converse, inverse, and contrapositive. DM.LR.1.C	**c** Represent a conditional statement as its converse, inverse, and contrapositive. DM.LR.1.C
Describe how symbolic logic can be used to map the processes of computer applications. DM.LR.1.D	**d** Describe how symbolic logic can be used to map the processes of computer applications. DM.LR.1.D
Construct a truth table to display all possible input combinations and their outputs. DM.LR.1.E	**e** Construct a truth table to display all possible input combinations and their outputs. DM.LR.1.E
Identify the rules of inf ba	**f** Identify the rules of inference and model basic logical statements including De Morgan's Law. DM.LR.1.F

Skip to main content

including De Morgan's Law. DM.LR.1.F	
Apply logical reasoning to model contextual situations and make decisions. DM.LR.1.G	g Apply logical reasoning to model contextual situations and make decisions. DM.LR.1.G
Apply informal logical reasoning to contextual problems (e.g., predicting the behavior of software, solving puzzles). DM.LR.2.A	a Apply informal logical reasoning to contextual problems (e.g., predicting the behavior of software, solving puzzles). DM.LR.2.A
Outline the basic structure of a proof technique (e.g., direct proof, proof by contradiction, induction). DM.LR.2.B	b Outline the basic structure of a proof technique (e.g., direct proof, proof by contradiction, induction). DM.LR.2.B
Deduce the best type of proof for a given problem. DM.LR.2.C	c Deduce the best type of proof for a given problem. DM.LR.2.C
Use the rules of inference to construct direct proofs and proofs by contradiction. DM.LR.2.D	d Use the rules of inference to construct direct proofs and proofs by contradiction. DM.LR.2.D
Construct induction proofs involving summations and inequalities. DM.LR.2.E	e Construct induction proofs involving summations and inequalities. DM.LR.2.E
Use a truth table to prove the logical equivalence of statements. DM.LR.2.F	f Use a truth table to prove the logical equivalence of statements. DM.LR.2.F
Explain basic properties of Boolean algebra: duality, complements, an for	a Explain basic properties of Boolean algebra: duality, complements, and standard forms. DM.LR.3.A

Skip to main content

Represent verbal statements as Boolean expressions. DM.LR.3.B	**b** **Represent verbal statements as Boolean expressions.** DM.LR.3.B
Apply Boolean algebra to prove identities and simplify expressions. DM.LR.3.C	**c** **Apply Boolean algebra to prove identities and simplify expressions.** DM.LR.3.C
Generate truth tables that encode the truth and falsity of two or more statements. DM.LR.3.D	**d** **Generate truth tables that encode the truth and falsity of two or more statements.** DM.LR.3.D
Explain the operation of discrete logic gates. DM.LR.3.E	**e** **Explain the operation of discrete logic gates.** DM.LR.3.E
Describe the relationship between Boolean algebra and electronic circuits. DM.LR.3.F	**f** **Describe the relationship between Boolean algebra and electronic circuits.** DM.LR.3.F
Analyze a combinational network using Boolean expressions. DM.LR.3.G	**g** **Analyze a combinational network using Boolean expressions.** DM.LR.3.G
Design simple combinational networks that use NAND (AND followed by NOT), NOR (OR followed by NOT), and XOR (exclusive-OR) gates. DM.LR.3.H	**h** **Design simple combinational networks that use NAND (AND followed by NOT), NOR (OR followed by NOT), and XOR (exclusive-OR) gates.** DM.LR.3.H
Compare and contrast inductive and deductive reasoning. DM.LR.4.A	**a** **Compare and contrast inductive and deductive reasoning.** DM.LR.4.A
Explain the relationship between weak and strong ind	**b** **Explain the relationship between weak and strong induction.** DM.LR.4.B

Skip to main content

Construct induction proofs involving a divisibility argument. DM.LR.4.C	**c** **Construct induction proofs involving a divisibility argument.** DM.LR.4.C
Prove the Binomial Theorem through mathematical induction. DM.LR.4.D	**d** **Prove the Binomial Theorem through mathematical induction.** DM.LR.4.D
Set and Number Theory	**SNT.1** **The student will identify and use the properties of sets and set operations.** DM.SNT.1 **SNT.2** **The student will apply the formulas of combinatorics.** DM.SNT.2 **SNT.3** **The student will use Pascal's Triangle to analyze numerical patterns and relationships.** DM.SNT.3
Compare and contrast sets, relations, and functions. DM.SNT.1.A	**a** **Compare and contrast sets, relations, and functions.** DM.SNT.1.A
Express relationships between sets using Venn diagrams. DM.SNT.1.B	**b** **Express relationships between sets using Venn diagrams.** DM.SNT.1.B
Describe a set using set-builder notation. DM.SNT.1.C	**c** **Describe a set using set-builder notation.** DM.SNT.1.C
Construct new sets using the set operations intersection, union, difference, and complement. DM.SNT.1.D	**d** **Construct new sets using the set operations intersection, union, difference, and complement.** DM.SNT.1.D
Identify the laws of set theory (e.g., associative, commutative, distributive, De Morgan's Law). DM.SNT.1.E	**e** **Identify the laws of set theory (e.g., associative, commutative, distributive, De Morgan's Law).** DM.SNT.1.E
Use the principle of inclusion and exclusion to a s	**f** **Use the principle of inclusion and exclusion to determine the size of a set.** DM.SNT.1.F

Skip to main content

Use the properties of set operations to prove set equality. DM.SNT.1.G	**g** **Use the properties of set operations to prove set equality.** DM.SNT.1.G
Create a tree diagram to represent relationships between independent events. DM.SNT.2.A	**a** **Create a tree diagram to represent relationships between independent events.** DM.SNT.2.A
Use the Fundamental (Basic) Counting Principle to determine the number of possible outcomes of an event. DM.SNT.2.B	**b** **Use the Fundamental (Basic) Counting Principle to determine the number of possible outcomes of an event.** DM.SNT.2.B
Determine the number of combinations possible when subsets of r elements are selected from a set of n elements without regard to order. DM.SNT.2.C	**c** **Determine the number of combinations possible when subsets of r elements are selected from a set of n elements without regard to order.** DM.SNT.2.C
Determine the number of permutations possible when r objects selected from n objects are ordered. DM.SNT.2.D	**d** **Determine the number of permutations possible when r objects selected from n objects are ordered.** DM.SNT.2.D
Use the pigeonhole principle to solve packing problems to facilitate proofs. DM.SNT.2.E	**e** **Use the pigeonhole principle to solve packing problems to facilitate proofs.** DM.SNT.2.E
Construct a proof by induction using principles of combinatorics. DM.SNT.2.F	**f** **Construct a proof by induction using principles of combinatorics.** DM.SNT.2.F
Construct Pascal's Triangle. DM.SNT.3.A	**a** **Construct Pascal's Triangle.** DM.SNT.3.A
Ex ha ex Skip to main content	**xpand binomials having positive integral exponents, using the Binomial Theorem nd Pascal's Triangle.** DM.SNT.3.B

Binomial Theorem and Pascal's Triangle. DM.SNT.3.B		
Compare the binomial coefficient to the calculation of combinations. DM.SNT.3.C	**c**	**Compare the binomial coefficient to the calculation of combinations.** DM.SNT.3.C
Identify the Fibonacci numbers within Pascal's Triangle. DM.SNT.3.D	**d**	**Identify the Fibonacci numbers within Pascal's Triangle.** DM.SNT.3.D
Graph Theory	**GT.1**	**The student will represent problems using vertex-edge graphs. The concepts of degree, connectedness, paths, planarity, and directed graphs will be analyzed.** DM.GT.1
	GT.2	**The student will solve problems through analysis and application of circuits, cycles, Euler paths, Euler circuits, Hamilton paths, and Hamilton circuits. Optimal solutions will be determined using existing algorithms and student-created algorithms.** DM.GT.2
	GT.3	**The student will apply graphs to conflict-resolution problems, such as graph coloring, scheduling, matching, and optimization.** DM.GT.3
	GT.4	**The student will recognize and apply algorithms to solve configuration, conflict-resolution, and sorting problems.** DM.GT.4
	GT.5	**The student will use algorithms to schedule tasks to determine a minimum project time.** DM.GT.5
Illustrate the basic terminology of graph theory (e.g., vertex, edge, graph, degree of a vertex). DM.GT.1.A	**a**	**Illustrate the basic terminology of graph theory (e.g., vertex, edge, graph, degree of a vertex).** DM.GT.1.A
Use graphs to map situations in which the vertices represent objects, and edges represent a particular relationship between objects. DM.GT.1.B	**b**	**Use graphs to map situations in which the vertices represent objects, and edges represent a particular relationship between objects.** DM.GT.1.B
Ide de co		**dentify and describe degree and connectedness.** DM.GT.1.C

Skip to main content

Determine whether a graph is planar or nonplanar. DM.GT.1.D	**d** Determine whether a graph is planar or nonplanar. DM.GT.1.D
Analyze the relationship between faces, edges, and vertices using Euler's formula (F = E – V + 2). DM.GT.1.E	**e** Analyze the relationship between faces, edges, and vertices using Euler's formula (F = E – V + 2). DM.GT.1.E
Use directed graphs (digraphs) to represent situations with restrictions in traversal possibilities. DM.GT.1.F	**f** Use directed graphs (digraphs) to represent situations with restrictions in traversal possibilities. DM.GT.1.F
Determine when graphs are trees. DM.GT.1.G	**g** Determine when graphs are trees. DM.GT.1.G
Determine whether a graph has an Euler circuit or path, and determine the circuit or path, if it exists. DM.GT.2.A	**a** Determine whether a graph has an Euler circuit or path, and determine the circuit or path, if it exists. DM.GT.2.A
Determine whether a graph has a Hamilton circuit or path, and determine the circuit or path, if it exists. DM.GT.2.B	**b** Determine whether a graph has a Hamilton circuit or path, and determine the circuit or path, if it exists. DM.GT.2.B
Count the number of Hamilton circuits for a complete graph with n vertices. DM.GT.2.C	**c** Count the number of Hamilton circuits for a complete graph with n vertices. DM.GT.2.C
Use an Euler circuit algorithm to solve optimization problems. DM.GT.2.D	**d** Use an Euler circuit algorithm to solve optimization problems. DM.GT.2.D
Model projects consisting of several subtasks, using a graph. DM.GT.3.A	**a** Model projects consisting of several subtasks, using a graph. DM.GT.3.A

Skip to main content

Use graphs to resolve conflicts that arise in scheduling. DM.GT.3.B	**b** **Use graphs to resolve conflicts that arise in scheduling.** DM.GT.3.B
Use graph coloring to determine the chromatic number of a graph. DM.GT.3.C	**c** **Use graph coloring to determine the chromatic number of a graph.** DM.GT.3.C
Recognize algorithms such as nearest neighbor, brute force, and cheapest link as they apply to graphs. DM.GT.4.A	**a** **Recognize algorithms such as nearest neighbor, brute force, and cheapest link as they apply to graphs.** DM.GT.4.A
Use Kruskal's algorithm to determine the shortest spanning tree of a connected graph. DM.GT.4.B	**b** **Use Kruskal's algorithm to determine the shortest spanning tree of a connected graph.** DM.GT.4.B
Use Prim's algorithm to determine the shortest spanning tree of a connected graph. DM.GT.4.C	**c** **Use Prim's algorithm to determine the shortest spanning tree of a connected graph.** DM.GT.4.C
Use Dijkstra's algorithm to determine the shortest spanning tree of a connected graph. DM.GT.4.D	**d** **Use Dijkstra's algorithm to determine the shortest spanning tree of a connected graph.** DM.GT.4.D
Specify in a digraph the order in which tests are to be performed. DM.GT.5.A	**a** **Specify in a digraph the order in which tests are to be performed.** DM.GT.5.A
Identify the critical path to determine the earliest completion time (minimum project time). DM.GT.5.B	**b** **Identify the critical path to determine the earliest completion time (minimum project time).** DM.GT.5.B

Skip to main content

Use the list-processing algorithm to determine an optimal schedule. DM.GT.5.C	**c** **Use the list-processing algorithm to determine an optimal schedule.** DM.GT.5.C
Create and test scheduling algorithms. DM.GT.5.D	**d** **Create and test scheduling algorithms.** DM.GT.5.D
Computational Methods	**CM.1** **The student will describe and apply sorting and searching algorithms used in processing and communicating information.** DM.CM.1
	CM.2 **The student will use recursive processes.** DM.CM.2
	CM.3 **The student will identify and apply cryptographic methods.** DM.CM.3
	CM.4 **The student will analyze the limitations of algorithms and their contextual relationships in computing.** DM.CM.4
Select and apply a sorting algorithm, such as a bubble sort, merge sort, or network sort. DM.CM.1.A	**a** **Select and apply a sorting algorithm, such as a bubble sort, merge sort, or network sort.** DM.CM.1.A
Describe the advantages and disadvantages of various sorting algorithms. DM.CM.1.B	**b** **Describe the advantages and disadvantages of various sorting algorithms.** DM.CM.1.B
Analyze the knapsack and bin-packing problems. DM.CM.1.C	**c** **Analyze the knapsack and bin-packing problems.** DM.CM.1.C
Select and apply search algorithms to analyze problems. DM.CM.1.D	**d** **Select and apply search algorithms to analyze problems.** DM.CM.1.D
Determine the average, best-case, and worst-case reasoning for different searches. DM.CM.1.E	**e** **Determine the average, best-case, and worst-case reasoning for different searches.** DM.CM.1.E
Compare and contrast ite pr	**a** **Compare and contrast iterative and recursive processes.** DM.CM.2.A

Skip to main content

Use recursive processes to model growth and decay. DM.CM.2.B	**b** **Use recursive processes to model growth and decay.** DM.CM.2.B
Use recursive processes to create fractals. DM.CM.2.C	**c** **Use recursive processes to create fractals.** DM.CM.2.C
Use recursive processes to generate the Fibonacci sequence. DM.CM.2.D	**d** **Use recursive processes to generate the Fibonacci sequence.** DM.CM.2.D
Determine if a recursive solution is more efficient than an iterative solution. DM.CM.2.E	**e** **Determine if a recursive solution is more efficient than an iterative solution.** DM.CM.2.E
Compare and contrast ciphers and codes. DM.CM.3.A	**a** **Compare and contrast ciphers and codes.** DM.CM.3.A
Describe the evolution of cipher systems. DM.CM.3.B	**b** **Describe the evolution of cipher systems.** DM.CM.3.B
Identify the Fundamental Theorem of Arithmetic. DM.CM.3.C	**c** **Identify the Fundamental Theorem of Arithmetic.** DM.CM.3.C
Describe how the complexity of prime factorization is used in cryptography. DM.CM.3.D	**d** **Describe how the complexity of prime factorization is used in cryptography.** DM.CM.3.D
Describe modular arithmetic in context (e.g., clocks, days of the week, measures of time). DM.CM.3.E	**e** **Describe modular arithmetic in context (e.g., clocks, days of the week, measures of time).** DM.CM.3.E
Analyze the relationship between divisibility and modulus. DM.CM.3.F	**f** **Analyze the relationship between divisibility and modulus.** DM.CM.3.F

Skip to main content

Determine congruence within modular arithmetic. DM.CM.3.G	**g** **Determine congruence within modular arithmetic.** DM.CM.3.G
Perform operations within modular arithmetic. DM.CM.3.H	**h** **Perform operations within modular arithmetic.** DM.CM.3.H
Apply modular arithmetic to problems in context (e.g., cryptography, International Standard Book Number (ISBN), International Bank Account Number (IBAN)). DM.CM.3.I	**i** **Apply modular arithmetic to problems in context (e.g., cryptography, International Standard Book Number (ISBN), International Bank Account Number (IBAN)).** DM.CM.3.I
Describe maximum complexity of an algorithm using Big O notation. DM.CM.4.A	**a** **Describe maximum complexity of an algorithm using Big O notation.** DM.CM.4.A
Describe Turing machines and how they are used to test the limits of computation. DM.CM.4.B	**b** **Describe Turing machines and how they are used to test the limits of computation.** DM.CM.4.B
Describe the halting problem and explain how it characterizes the fundamental limitations of computation and undecidability. DM.CM.4.C	**c** **Describe the halting problem and explain how it characterizes the fundamental limitations of computation and undecidability.** DM.CM.4.C
Explain the P versus NP problem and defend a justification for equality, inequality, or undecidability. DM.CM.4.D	**d** **Explain the P versus NP problem and defend a justification for equality, inequality, or undecidability.** DM.CM.4.D
Analyze how the equivalence of P- and NP-class problems might impact society. DM.CM.4.E	**e** **Analyze how the equivalence of P- and NP-class problems might impact society.** DM.CM.4.E