

# **Mathematical Applications and Algorithms**

## **Curriculum Framework**

**2012**

Course Title: Mathematical Applications and Algorithms  
 Course/Unit Credit: 1  
 Course Number: 439080  
 Teacher Licensure: Please refer to the Course Code Management System (<https://adedata.arkansas.gov/ccms/>) for the most current licensure codes.  
 Grades: 9-12  
 Prerequisite: Algebra I, Algebra II

Mathematical Applications and Algorithms

This course is designed to provide students with experiences in using the computer to solve problems that can be set up as mathematical models. Students should have experience working with computer spreadsheets. Students will develop and refine skills in logic, organization, and precise expression, thereby enhancing learning in other disciplines. Programming will be introduced in the context of mathematical concepts and problem solving. Students will define a problem; develop, refine, and implement a plan; and test and revise the solution. Students will use manipulatives, graphing calculators, and computer spread sheet applications to develop and attach meaning to abstract ideas. Teachers are responsible for including the eight Standards for Mathematical Practice found in the Common Core State Standards for Mathematics (CCSS-M). Mathematical Applications and Algorithms does not require Arkansas Department of Education approval.

Prerequisite: Algebra I, Algebra II

Strand	Standard
Functions	
	1. The student will graphically, numerically, and algebraically evaluate concepts of different types of functions; include recursively defined functions, series, and sequences; and apply them to programming applications.
Equations and Formulas	
	2. The student will manipulate formulas and equations and apply them to programming applications.
Systems of Equations and Matrices	
	3. The student will create, manipulate, and solve systems of equations and matrices and apply them to programming arrays.
Problem Solving	
	4. The student will develop and apply logical reasoning skills to solve real-world problems through the development of mathematical models.
Program Design	
	5. The student will design a step-by-step plan to solve a given problem.
Program Implementation	
	6. The student will create, edit, and execute programs using a programmable calculator and/or computer spreadsheet application program.
Data Manipulation and Testing	
	7. The student will manipulate data to adjust and test programs designed using a programmable calculator and/or computer spreadsheet application.

Strand: Functions

Content Standard 1: The student will graphically, numerically, and algebraically evaluate concepts of different types of functions; include recursively defined functions, series, and sequences; and apply them to programming applications.

Connections  
to CCSS-M

F.1.MAA.1	Recognize that sequences are functions, sometimes defined recursively, whose domain is a subset of the integers [e.g., the <i>Fibonacci sequence</i> is defined recursively by $f(0) = f(1) = 1, f(n + 1) = f(n) + f(n - 1)$ for $n \geq 1$ ]	F.IF.3
F.1.MAA.2	Graph functions expressed symbolically and show key features of the graph, by hand in simple cases and using technology for more complicated cases: <ul style="list-style-type: none"> <li>graph exponential and logarithmic functions showing intercepts, end behavior, and trigonometric functions, showing period, midline, and <i>amplitude</i></li> <li>graph linear and quadratic functions and show intercepts, maxima, and minima</li> <li>graph rational functions identifying zeros and asymptotes when suitable factorizations are available and showing end behavior</li> </ul>	F.IF.7a, F.IF.7d, F.IF.7e
F.1.MAA.3	Write a function defined by an expression in different but equivalent forms to reveal and explain different properties of the function	F.IF.8
F.1.MAA.4	Compare properties of two functions each represented in a different way: algebraically, graphically, numerically in tables, or by verbal descriptions (e.g., given a graph of one quadratic function and an algebraic expression for another, determine which has the larger maximum)	F.IF.9
F.1.MAA.5	Write a function that describes a relationship between two quantities: <ul style="list-style-type: none"> <li>compose functions [e.g., if <math>T(y)</math> is the temperature in the atmosphere as a function of height, and <math>h(t)</math> is the height of a weather balloon as a function of time, then <math>T(h(t))</math> is the temperature at the location of the weather balloon as a function of time]</li> <li>combine standard function types using arithmetic operations (e.g., build a function that models the temperature of a cooling body by adding a constant function to a decaying exponential and relate these functions to the model)</li> <li>determine an explicit expression, a recursive <i>process</i>, or steps for calculation from a context</li> </ul>	F.BF.1
F.1.MAA.6	Write arithmetic and geometric sequences both recursively and with an explicit formula, use them to model situations, and translate between the two forms	F.BF.2
F.1.MAA.7	Understand that restricting a trigonometric function to a domain on which it is always increasing or always decreasing allows its inverse to be constructed	F.TF.6
F.1.MAA.8	Use inverse functions to solve trigonometric equations that arise in modeling contexts, evaluate the solutions using technology, and interpret them in terms of the context	F.TF.7
F.1.MAA.9	Know there is a <i>complex number</i> $i$ such that $i^2 = -1$ , and every <i>complex number</i> has the form $a + bi$ with $a$ and $b$ real	N.CN.1
F.1.MAA.10	Use the relation $i^2 = -1$ and the commutative, associative, and distributive properties to add, subtract, and multiply <i>complex numbers</i>	N.CN.2
F.1.MAA.11	Find the conjugate of a complex number; use conjugates to find moduli and quotients of <i>complex numbers</i>	N.CN.3

Strand: Equations and Formulas

Content Standard 2: The student will manipulate formulas and equations and apply them to programming applications.

		Connections to CCSS-M
EF.2.MAA.1	Represent <i>constraints</i> by equations or inequalities and by systems of equations and/or inequalities (two and three variable systems); interpret solutions as viable or nonviable options in a modeling context (e.g., represent inequalities describing nutritional and cost <i>constraints</i> on combinations of different foods)	A.CED.3
EF.2.MAA.2	Rearrange formulas to highlight a quantity of interest using the same reasoning as in solving equations (e.g., rearrange Ohm's law $V = IR$ to highlight resistance R)	A.CED.4
EF.2.MAA.3	Give an informal argument for the formulas for the circumference of a circle, area of a circle, and volume of a cylinder, pyramid, and cone; use dissection arguments, <i>Cavalieri's principle</i> , and informal limit arguments	G.GMD.1
EF.2.MAA.4	Give an informal argument using <i>Cavalieri's principle</i> for the formulas for the volume of a sphere and other solid figures	G.GMD.2
EF.2.MAA.5	Use volume formulas for cylinders, pyramids, cones, and spheres to solve problems	G.GMD.3

Strand: Systems of Equations and Matrices

Content Standard 3: The student will create, manipulate, and solve systems of equations and matrices and apply them to programming arrays.

		Connections to CCSS-M
SEM.3.MAA.1	Represent a system of linear equations as a single matrix equation in a vector variable	A.REI.8
SEM.3.MAA.2	Find the inverse of a matrix if it exists and use it to solve systems of linear equations; use technology for matrices of dimension $3 \times 3$ or greater	A.REI.9
SEM.3.MAA.3	Use matrices to represent and manipulate data (e.g., to represent payoffs or incidence relationships in a network)	N.VM.6
SEM.3.MAA.4	Multiply matrices by scalars to produce new matrices (e.g., as when all of the payoffs in a game are doubled)	N.VM.7
SEM.3.MAA.5	Add, subtract, and multiply matrices of appropriate dimensions	N.VM.8
SEM.3.MAA.6	Understand that, unlike multiplication of numbers, matrix multiplication for square matrices is not a commutative operation but still satisfies the associative and distributive properties	N.VM.9
SEM.3.MAA.7	Understand that the zero and identity matrices play a role in matrix addition and multiplication similar to the role of 0 and 1 in the real numbers; the <i>determinant</i> of a square matrix is nonzero if and only if the matrix has a multiplicative inverse	N.VM.10
SEM.3.MAA.8	Work with $2 \times 2$ matrices as transformations of the plane and interpret the absolute value of the <i>determinant</i> in terms of area	N.VM.12

Strand: Problem Solving

Content Standard 4: The student will develop and apply logical reasoning skills to solve real-world problems through the development of mathematical models.

		Connections to CCSS-M
PS.4.MAA.1	Analyze and interpret graphs, charts, and tables in the design and implementation of a computer <i>program</i>	Not Applicable
PS.4.MAA.2	Write an <i>algorithm</i> to solve mathematical problems using formulas, equations, and functions	Not Applicable
PS.4.MAA.3	Analyze and interpret truth tables from basic statements using <i>Boolean</i> operators (AND, OR, XOR, and NOT)	Not Applicable
PS.4.MAA.4	Write an <i>algorithm</i> from a mathematical model	Not Applicable

Strand: Program Design

Content Standard 5: The student will design a step-by-step plan to solve a given problem.

		Connections to CCSS-M
PD.5.MAA.1	Translate a mathematical expression into a computer statement which involves writing assignment statements and using the order of operations	Not Applicable
PD.5.MAA.2	Implement conditional statements that include if/then, if/then/else, case statements, and <i>Boolean logic</i>	Not Applicable
PD.5.MAA.3	Define and differentiate Decision (selection) and Sequence ( <i>process</i> )	Not Applicable
PD.5.MAA.4	Represent an <i>algorithm</i> representation as a <i>flowchart</i> and in <i>pseudocode</i>	Not Applicable
PD.5.MAA.5	Use <i>flowchart</i> terminology such as terminals (starts and stops), <i>subroutines</i> , and connectors	Not Applicable
PD.5.MAA.6	Develop recursive relationships from mathematical models (e.g., arithmetic and geometric sequences)	Not Applicable
PD.5.MAA.7	Define and use variable <i>data types</i> (e.g., integers, real, character)	Not Applicable

Strand: Program Implementation

Content Standard 6: The student will create, edit, and execute programs using a programmable calculator and/or computer spreadsheet application program.

		Connections to CCSS-M
PI.6.MAA.1	Create, edit, and execute a <i>program</i> utilizing an <i>array</i> in a programmable calculator and spreadsheet application	Not Applicable
PI.6.MAA.2	Create, edit, and execute <i>programs</i> using recursions and <i>loops</i> in a programmable calculator and spreadsheet application	Not Applicable
PI.6.MAA.3	Create, edit, and execute <i>programs</i> to calculate mathematical formulas (e.g., quadratic formula, volume of a simple solid)	Not Applicable
PI.6.MAA.4	Develop functional <i>programs</i> from <i>algorithms</i> developed from the mathematical models	Not Applicable
PI.6.MAA.5	Create <i>programs</i> using various display modes including tables and graphs	Not Applicable
PI.6.MAA.6	Locate, categorize, and implement programming commands	Not Applicable
PI.6.MAA.7	Use <i>subroutines</i> to reduce keystrokes and memory use	Not Applicable
PI.6.MAA.8	Use a spreadsheet application to <i>sort</i> data using various methods (e.g., <i>bubble, quick, shell</i> )	Not Applicable

Strand: Data Manipulation and Testing

Content Standard 7: The student will manipulate data to adjust and test programs designed using a programmable calculator and/or computer spreadsheet application.

		Connections to CCSS-M
DMT.7.MAA.1	Compare results from mathematical formulas to their <i>program</i> equivalent	Not Applicable
DMT.7.MAA.2	Identify and eliminate error messages using troubleshooting techniques ( <i>debug</i> )	Not Applicable
DMT.7.MAA.3	Understand and differentiate the different error types (e.g., <i>syntax, runtime, logic</i> )	Not Applicable
DMT.7.MAA.4	Design and investigate best-case or worst-case scenarios of a <i>program</i>	Not Applicable
DMT.7.MAA.5	Name a range, one cell or a group of cells; use the name to select cells	Not Applicable
DMT.7.MAA.6	Estimate best-case or worst-case scenarios using a spreadsheet application scenario tool	Not Applicable

## Glossary for Mathematical Applications and Algorithms

Algorithm	A formula or set of steps for solving a particular problem
Amplitude	Half the difference between the minimum and maximum values of the range; only periodic functions with a bounded range have an amplitude
Array	A series of elements of the same data type placed in contiguous memory locations that can be individually referenced by a unique identifier
Boolean logic	<i>Boolean</i> logic is a form of algebra in which all values are reduced to either true or false
Bubble sort	Sort by comparing each adjacent pair of items in a list, swapping the items if necessary, and repeating the pass through until no swaps are done
Cavalieri's principle	A method of finding the volume of any solid for which cross-sections by parallel planes have equal areas
Complex number(s)	Number(s) that can be written as the sum or difference of a real number and an imaginary number [e.g., $3 - 2i$ or $-1 + \sqrt{5}i$ ]
Constraint(s)	Condition(s) or proposition(s) that must be maintained as true
Data type(s)	Specifies and limits the kind of data that may be entered into a field
Debug	Find and remove programming errors (runtime, syntax, and logic)
Determinant	A single number obtained from a matrix that reveals a variety of the matrix's properties
Fibonacci sequence	The sequence of numbers 1, 1, 2, 3, 5, 8, 13, 21, 34, ... for which the next term is found by adding the previous two terms
Flowchart	A graphic structured representation of the major steps in a process
Logic	A mathematical treatment of formal logic whereby a system of symbols (AND, OR, and NOT) is used to represent quantities and relationships
Loop	A single execution of a set of instructions that are repeated until a certain condition is met
Process	An instance of a running program
Program	An organized list of instructions that, when executed, causes the computer to behave in a predetermined manner
Pseudocode	An English language version of an algorithm that will ultimately be translated into real computer code
Quick sort	Selection of an element (which becomes a pivot) from the array, partitions the remaining elements into greater and less than the pivot, and recursively sorts the partitions
Runtime	The period of time during which a program is executing
Shell sort	A sorting algorithm developed by Donald Shell that compares items of the list that lie far apart; it is also known as the diminishing increment sort
Sort	Arrange items in a predetermined order
Subroutine	A short program segment that performs a specific function and is available for general use by other programs and routines
Syntax	The rules for how symbols and words can be combined within a particular programming language