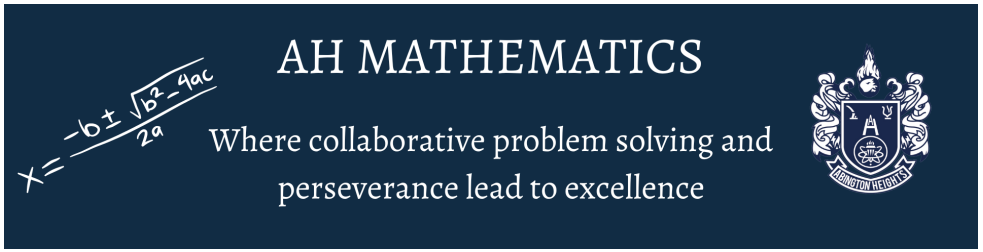
Abington Heights School District
Computer Science II AP: Java II Curriculum

In Computer Science II AP: Java II, students develop their computer programming skills through the following areas of study:

1. Introductory Data Structures (Arrays, 2D Arrays, ArrayLists)
2. Object-Oriented Programming Principles
3. Recursion
4. Searching and Sorting Algorithms

Board Approval Date: June 7, 2023
Adoption: 2023 - 2024 SY
Review Date:



Abington Heights Math Framework

Stakeholders	Actions
Students	★ Engage in mathematical discussions, share their ideas openly, be inquisitive, seek to understand and learn more about mathematical concepts, and try their best daily. ★ Exhibit creativity and curiosity in problem solving individually and collaboratively. ★ Persevere in engaging and challenging daily mathematical practice. ★ Come prepared to learn every day.
Teachers	★ Create a safe and collaborative classroom environment where students feel vested in a shared vision for mathematical excellence. ★ Develop high quality instruction that meets the needs of all learners through differentiation. ★ Use a variety of 21st century methodologies to advance learning. ★ Partner with parents and guardians to support student success. ★ Establish a collaborative community within the building and amongst grade levels to ensure a cohesive level of instruction.
Building Leaders	★ Deeply understand the needs of teachers, students, the instructional materials being used, programs being implemented, and the expectations for state-level assessment scores ○ Knowledgeable about program and grade level standards ○ Ensure consistent and equal access to high-quality instructional materials and resources, building. ★ Be partners with teachers, students and families: ○ Provide guidance and support to the mathematical community. ○ Understand needs of teachers, students and families. ★ Trust the educators to make professional decisions based on program, student, and district needs.
Central Admin	★ Effectively communicate to the school board and community specific areas of need and how to support teachers and building leaders in a quest for mathematical excellence ★ Deeply understand the needs of teachers, students, the instructional materials being used, programs being implemented, and the expectations for state-level assessment scores ○ Have a common metric for mathematical excellence. ○ Ensure consistent and equal access to high-quality instructional materials and resources, district. ○ Re-examine best practices/curriculum routinely (6 years). ★ Support a culture of collaboration between the other stakeholder groups to maintain the standard of excellence of the Abington Heights ★ Trust the educators to make professional decisions based on program, student, and district needs.
Parents/ Community	★ Be a strong support system and contribute by building a positive math community for students. ★ Encourage a positive math mindset. ★ Have conversations with their children about school and ask what they are learning about in school. ★ Be open, receptive to the district's ideas about student learning and reach out to teachers/school to learn more about how they can support. ★ Trust the educators to make professional decisions based on program, student, and district needs.
School Board	★ Provide the fiscal resources to support: ○ Highly qualified professionals for mathematics ○ High-quality instructional materials ○ Effective and efficient math interventions for remediation ○ Professional development for math content and instructional practices ★ Trust the educators to make professional decisions based on program, student, and district needs.

Computer Science II AP: Java II Scope and Sequence

Month	Unit	Estimated Number of Weeks
September	Introduction & Arrays	3 1/2
October	2d Arrays	3 1/2
	References	1/2
November	References	1/2
	Object Oriented Programming, Part 1 (Classes And Objects)	3
December	Object Oriented Programming, Part 1 (Classes And Objects)	1
	Object Oriented Programming, Part 2	2
January	Inheritance & Polymorphism	3
	Mid-Term Review	1
February	Interfaces	2
	Arraylists	2
March	Arraylists	1
	Recursion	1 1/2
	Searching and Sorting Algorithms	1 1/2
April	Searching and Sorting Algorithms	1/2
	AP Exam Prep	3 1/2
May	AP Exam Prep	1/2
	Post-Exam Final Project	3 1/2
June	Post-Exam Final Project	1/2

	Essential Questions	Content	Skills	Activities	Assessment / Evidence of Learning
INTRODUCTION & ARRAYS	How do you create an array? What's the syntax for accessing / setting elements? What's the difference between an index and an element? What are the options for printing an array and how does output differ? What are the pros and cons of using a for loop vs. enhanced for loop with an array? How can you simulate removing an element from an array? How do you use an array's length when looping through it?	Arrays and elements Declaring and initializing empty array - Array length - Zero equivalents Setting elements Declaring and initializing an array with values Reassigning an array Strings as arrays Accessing individual elements Accessing elements via for loop Accessing elements via enhanced for loops - Enhanced for loop definition - Lack of index access Limitation of arrays - The Arrays class	Create an empty array of a certain size Create an array with values Access / set individual elements within an array Using a loop (for loop or enhanced for loop) to access/print elements Pass array as a parameter Print an array using Arrays.toString method Sum all values in an array Count all occurrences of a value or values that fit a condition in an array Find the largest and smallest values in an array "Remove" occurrences of a value or elements that fit a condition from an array Reverse an array Fill in new array using certain values from a given array	Towers of Hanoi logic problem in groups Allstar Algorithms: Summer homework review coding activity Arrays slides - Coding "Try Its" throughout slides - Code tracing throughout slides - Code-along at end of slides AP Classroom Progress Check: MCQ Arrays quiz Arrays worksheet Arrays programming lab Arrays free response question Arrays Jeopardy review game Arrays unit test	Q&A during slides Coding "Try Its" throughout slides Code tracing throughout slides AP Classroom Progress Check: MCQ Arrays quiz Arrays worksheet Arrays programming lab Arrays unit test

	Essential Questions	Content	Skills	Activities	Assessment / Evidence of Learning
INTRODUCTION & ARRAYS (continued)		Arrays as parameters	Determine if an array is ascending / descending Determine if an array has repeated elements		
2D ARRAYS	How do you create a 2D array? What's the syntax for accessing / setting elements? What's the syntax for accessing / setting rows? When using nested loops to access a 2D array, how do you determine what the outer and inner loop each iterate over (row vs column index)? How do you use a 2D array's length and a row's length when looping through it?	Definition of 2D array - Term: "Matrix" - Rectangular 2D arrays Declaring and initializing empty 2D array - Rows vs columns Declaring and initializing 2D array with values Accessing and setting - Accessing elements - Accessing entire row - Setting elements - Setting entire row Length and 2D arrays - Getting number of rows - Getting number of columns	Create an empty 2D array of a certain size Create a 2D array with values Access/set individual elements Using nested loops (for loops or enhanced for loops) to access/print all or some elements Pass 2D array as a parameter Print a 2D array using Arrays.deepToString method Determine if given location is valid in a 2D array Sum values in 2D array Access elements bordering or relative to a given location Find patterns within a 2D array's elements Check what row and / or column index the loop currently is at	2D arrays slides - Coding "Try Its" throughout slides - Code tracing throughout slides - Code-along at end of slides AP Classroom Progress Check: MCQ 2D Arrays quiz 2D Arrays worksheet 2D Arrays programming lab 2D Arrays free response question 2D Arrays Jeopardy review game 2D Arrays unit test	Q&A during slides Coding "Try Its" throughout slides Code tracing throughout slides AP Classroom Progress Check: MCQ 2D Arrays quiz 2D Arrays worksheet 2D Arrays programming lab 2D Arrays unit test

	Essential Questions	Content	Skills	Activities	Assessment / Evidence of Learning
2D ARRAYS (continued)	How do you determine if a given index pair (i.e. location) is valid? What are the options for printing a 2D array and how does output differ?	Nested for loops and 2D arrays - Nested for loops - Nested enhanced for loops - Mixed nested loops Printing 2D arrays	Fill in a new 2d array within nested loop Access elements in a specified order Access the last row / column Check for a value within a 2D array		
REFERENCES & OBJECT ORIENTED PROGRAMMING (3 part unit) ***References*** ***Object Oriented Programming, Part 1 (Classes and Objects)***	What's the difference between stack and heap memory? How does the keyword "new" affect what gets added to memory? How can you determine if an variable's stack value or heap value is being modified? What represents state and what represents behavior of an object in a class?	Stack memory Heap memory Primitives vs objects in memory Strings as reference variables Arrays as reference variables - Dereferencing Passing reference variables as parameters Objects and classes - Object instances - State and behavior - Instance variables	Specify what stack memory and heap memory store for both primitive values and objects Analyze if two reference variables are aliases Trace code involving reference variables to determine output Determine if the elements of an array are changed on the heap Determine if reference variables passed as parameters change within a method Declare a class for a new object type with instance variables representing state and methods representing behavior Create object instances of your new type in a client class	References slides - Code tracing throughout slides AP Classroom Progress Check: MCQ References worksheet References quiz Object Oriented Programming (OOP) Part 1: Classes and Objects slides - Coding "Try Its" throughout slides	Q&A during slides Code tracing throughout slides AP Classroom Progress Check: MCQ References worksheet References quiz Q&A during slides Coding "Try Its" throughout slides

	Essential Questions	Content	Skills	Activities	Assessment / Evidence of Learning
Object Oriented Programming, Part 1 (Classes and Objects) (continued)	What's the difference between a class and an object? When does a class invisibly have Java's default constructor? What's the purpose of: constructor, of mutator, accessor and toString methods? When should you use the keyword "this"? Given multiple constructors, how does a client class determine which to use?	Constructors - Constructor parameters Accessors and mutator methods - Private variables Printing objects - toString method Custom types as parameters - equals method Arrays of objects	Implement a constructor method with zero or more parameters that sets all instance variables, using the keyword "this" when appropriate Implement accessor and mutator methods for instance variables Call constructors, accessors, mutators and other instance methods on an object from within a client class Implement a toString method for a custom object type Print an object instance from within a client class without calling toString explicitly Implement a method that compares two instances of the same class and call it from the client class	AP Classroom Progress Check: MCQ Classes and Objects worksheet Classes and Objects quiz Classes and Objects programming lab Gradebook programming project OOP free response question References and OOP Part 1 Kahoot review game References and OOP Part 1 unit test	AP Classroom Progress Check: MCQ Classes and Objects worksheet Classes and Objects quiz Classes and Objects programming lab Gradebook programming project References and OOP Part 1 unit test
Object Oriented Programming, Part 2	What's the difference between a static variable, a constant, and an instance variable?	public vs. private Static variables - constants Static methods Method overloading	Declare and assign a static variable Declare and assign a constant Access a public constant from within a client class	Object Oriented Programming (OOP) Part 2: Advanced Topics slides - Code tracing throughout slides	Q&A during slides Code tracing throughout slides

	Essential Questions	Content	Skills	Activities	Assessment / Evidence of Learning
Object Oriented Programming, Part 2 (continued)	What's the difference between an instance method and static method? What is method overloading? What does == compare vs. .equals?	Passing the current instance as a parameter ==, equals method, compareTo method	Call a static method from within client class Declare multiple methods with the same name to overload it Pass the current instance as a parameter using the keyword "this" Compare reference variables using ==, equals method, and compareTo method Trace code involving multiple custom objects	AP Classroom Progress Check: MCQ OOP Part 2 programming lab OOP Part 2 quiz	AP Classroom Progress Check: MCQ OOP Part 2 programming lab OOP Part 2 quiz
INHERITANCE & POLYMORPHISM	How do you recognize when you should use inheritance? What keyword do you use to specify inheritance when declaring a class? What variables and methods get inherited from the superclass? What keyword do you use when overriding a method?	"Is-a" relationships between classes Subclasses and superclasses Extending a superclass Inheriting methods Overriding methods Multiple levels of inheritance Calling methods in the super class Subclass constructors	Implement inheritance by writing a subclass of another class Override a method from the superclass in the subclass Make instances of the superclass and subclass in a client class, and call methods on each From within a subclass' method, call that same method in the superclass as part of the subclass method's implementation Create a subclass constructor	Inheritance slides - Coding "Try Its" throughout slides - Code tracing throughout slides AP Classroom Progress Check: MCQ Inheritance quiz Inheritance programming lab Polymorphism slides - Code tracing throughout slides	Q&A during both of slides Coding "Try Its" throughout Inheritance slides Code tracing throughout both set of slides AP Classroom Progress Check: MCQ Inheritance quiz

	Essential Questions	Content	Skills	Activities	Assessment / Evidence of Learning
INHERITANCE & POLYMORPHISM (continued)	What is the first thing all subclass constructors should do? When tracing code, how do you determine if the subclass or superclass' method is being used? When is it valid to cast from a superclass to a subclass?	Passing the subclass type as a parameter of type superclass Storing the subclass type in an array of type superclass Casting from the superclass to the subclass	Specify if a line of code declaring a subclass instance is valid Trace code that involves both subclass and superclass functionality Pass a subclass instance as an argument to a method whose parameter is of type superclass Create an array of type superclass; store subclass instances within it Cast an instance of the superclass to the appropriate subclass type Trace code involving inheritance and polymorphism	Polymorphism worksheet Inheritance and Polymorphism review Inheritance and Polymorphism unit test Inheritance free response question	Inheritance programming lab Polymorphism worksheet Inheritance and Polymorphism unit test
MIDTERM REVIEW	N/A	All content from Comp Sci I H and this year thus far	All skills from Comp Sci I H and this year thus far	Midterm review packet Midterm exam Reviewing midterm afterwards	Midterm exam

	Essential Questions	Content	Skills	Activities	Assessment / Evidence of Learning
INTERFACES	How do you recognize when you should use interfaces? What keyword do you use to specify an interface when declaring it? Are methods and variables in an interface public or private my default? What type of variable is any variable in an interface? What's the syntax for declaring an abstract method? Can an interface be instantiated? Can interfaces extend other interfaces? Can interfaces extend classes?	When to use interfaces Declaring an interface Limitations of an interface Abstract methods Default modifiers for methods Default modifiers for variables Implementing an interface Creating an object of type interface Casting with interfaces Polymorphism with interfaces Interfaces extending one to many interfaces A class implementing one to many interfaces Declaring a class that extends a class and implements interfaces	Identify when to use an interface vs. inheritance Write an interface with abstract methods Write a class that implements an interface, and therefore implements its abstract methods Declare an object whose type matches the interface Trace code involving interfaces and polymorphism Write an interface that extends another interface Write an interface that extends multiple other interfaces Write a class that implements multiple interfaces Compare and contrast classes and interfaces re: abstract methods Compare and contrast classes and interfaces re: constructors and instantiation Compare and contrast classes and interfaces re: instance variables	Interfaces slides - Coding "Try Its" throughout slides - Code tracing throughout slides AP Classroom Progress Check: MCQ Interfaces programming lab Interfaces unit test Interfaces free response question	Q&A during slides Coding "Try Its" throughout slides Code tracing throughout slides AP Classroom Progress Check: MCQ Interfaces programming lab Interfaces unit test

	Essential Questions	Content	Skills	Activities	Assessment / Evidence of Learning
INTERFACES (continued)	Can a class extend an interface? Can a class be implemented? Can an interface implement another interface? When both extending and implementing, which keyword comes first? Can a class contain abstract methods? Can an interface contain non-abstract methods? Can an interface contain constructors?	Comparing interfaces and classes	Compare and contrast classes and interfaces re: static variables Compare and contrast classes and interfaces re: constants Compare and contrast classes and interfaces re: class constants Compare and contrast classes and interfaces re: being extended Compare and contrast classes and interfaces re: being implemented		

	Essential Questions	Content	Skills	Activities	Assessment / Evidence of Learning
ARRAYLISTS	Can ArrayLists store primitives? How can you create an ArrayList that stores whole numbers or decimal numbers? What is autoboxing? What is unboxing? Are ArrayLists resizeable? What type does ArrayList's add method return? What type does ArrayList's set method return? What type does ArrayList's remove method return? What type does ArrayList's contain method return?	Integer objects Double objects Autoboxing Unboxing Creating an ArrayList Generic types Adding elements to an ArrayList Setting elements in an ArrayList Removing elements from a ArrayList More ArrayList modifier methods Getting elements from an ArrayList Getting the size of an ArrayList Printing an ArrayList Searching an ArrayList with contains method	Create an Integer object from an int Access the int value of an Integer object Create a Double object from a double Access the double value of a Double object Trace code involving autoboxing and unboxing Declare and assign a new ArrayList Add an element to the end of an ArrayList Add an element to a certain index in an ArrayList Set an element at a certain index in an ArrayList Remove a certain element from an ArrayList Remove an element at certain index from an ArrayList Get an element at a certain index in an ArrayList	ArrayLists slides - Coding "Try Its" throughout slides - Code tracing throughout slides - Code-along at end of slides AP Classroom Progress Check: MCQ ArrayList quiz ArrayList worksheet ArrayList programming lab ArrayList free response question ArrayList review ArrayList unit test	Q&A during slides Coding "Try Its" throughout slides Code tracing throughout slides AP Classroom Progress Check: MCQ ArrayList quiz ArrayList worksheet ArrayList programming lab ArrayList unit test

	Essential Questions	Content	Skills	Activities	Assessment / Evidence of Learning
ARRAYLISTS (continued)	What method retrieves an element in an ArrayList? What method changes an element in an ArrayList? Can ArrayLists be passed as parameters? Can ArrayLists be returned from methods? How does removing an element from an ArrayList affect the remaining elements? Why can't you use an enhanced for loop when removing and adding elements to an ArrayList? When can you use an enhanced for loop with an ArrayList?	More ArrayList accessor methods ArrayLists vs Arrays ArrayLists in Methods ArrayLists and enhanced loops	Get the size of an ArrayList Print an ArrayList using the toString method Use the contains method to determine if an ArrayList includes a certain element Trace code involving an ArrayList's elements being modified Perform all the same algorithms on an ArrayList that could be done on an array		

	Essential Questions	Content	Skills	Activities	Assessment / Evidence of Learning
RECURSION	What does the word "recursion" mean? What coding problems best lend themselves to recursion? What's the base case? What's the recursive case? How do you prevent infinite looping with recursion? How is the for loop counter initialization useful when converting code to use recursion? How is the for loop condition useful when converting code to use recursion?	Defining recursion Reasons to use recursion instead of loops Base and recursive cases Tracing recursive code Converting code with for loops to use recursion	Trace recursive code involving one recursive call Trace recursive code with a math operation with the recursive call Trace recursive code with two recursive calls Write recursive code by converting existing code that involves for loop to code that uses recursion (using the loop counter initialization, loop condition, loop counter update, and loop body to aid in the conversion) Write recursive code to print Write recursive code to perform math operations Write recursive code that involves manipulating Strings	Recursion slides - Code tracing throughout slides - Code-along at end of slides AP Classroom Progress Check: MCQ Recursion worksheet Recursion programming lab Recursion test review sheet Recursion unit test	Q&A during slides Code tracing throughout slides AP Classroom Progress Check: MCQ Recursion worksheet Recursion programming lab Recursion unit test

	Essential Questions	Content	Skills	Activities	Assessment / Evidence of Learning
RECURSION (continued)	How is the for loop counter update useful when converting code to use recursion? How is the for loop body useful when converting code to use recursion?				
SEARCHING AND SORTING ALGORITHMS	What's the algorithm for Sequential Search? What does the Sequential Search algorithm return? Does Sequential Search require elements to be sorted? How does the order of the elements affect the performance of Sequential Search?	Sequential Search algorithm Best and worst case code efficiency for Sequential Search Binary Search algorithm - loop version Binary Search algorithm - recursive version Best and worst case code efficiency for Binary Search Binary Search "number of iterations" formula Selection Sort algorithm Best and worst case code efficiency for Selection Sort	Trace code involving Sequential Search where the sought after element you're searching for exists in the array Trace code involving Sequential Search where the element you're searching for does not exist in the array Perform the Sequential Search algorithm on a data set without being given the Sequential Search code Trace code involving Binary Search where the element you're searching for exists in the array Trace code involving Binary Search where the element you're searching for does not exist in the array	Searching Algorithms slides - Code tracing throughout slides Searching Algorithms Worksheet Sorting Algorithms slides - Code tracing throughout slides Sorting Algorithms worksheet AP Classroom Progress Check: MCQ Searching and Sorting Algorithms Test review sheet Searching and Sorting Algorithms unit test	Q&A during slides Code tracing throughout slides Searching Algorithms Worksheet Sorting Algorithms Worksheet AP Classroom Progress Check: MCQ Searching and Sorting Algorithms unit test

	Essential Questions	Content	Skills	Activities	Assessment / Evidence of Learning
SEARCHING AND SORTING ALGORITHMS (continued)	What's the best and worst case efficiency for Sequential Search? What's the algorithm for Binary Search? What is the prerequisite of an array in order to use Binary Search on it? What should you start min and max at for Binary Search? What's the best and worst case efficiency for Binary Search? What's the algorithm for Selection Sort? What's the best and worst case efficiency for Selection Sort?	Insertion Sort algorithm Best and worst case code efficiency for Insertion Sort Mergesort algorithm Best and worst case code efficiency for Mergesort	Perform the Binary Search algorithm on a data set without being given the Binary Search code Calculate the number of iterations of Binary Searching using log Trace code involving Selection Sort Determine how many steps and swaps are involved in a Selection Sort for a given data set Trace code involving Insertion Sort Determine how many shifts are involved in an Insertion Sort for a given data set Trace code involving Mergesort Determine how many splits and merges are involved in a Selection Sort for a given data set Be able to compare the sorting algorithms in terms of space and efficiency		

	Essential Questions	Content	Skills	Activities	Assessment / Evidence of Learning
SEARCHING AND SORTING ALGORITHMS (continued)	What's the algorithm for Insertion Sort? What's the best and worst case efficiency for Insertion Sort? What's the algorithm for Mergesort? What's the best and worst case efficiency for Mergesort? What are the advantages and disadvantages to Mergesort?				
AP EXAM PREP	N/A	All content from Comp Sci I H and Computer Science II AP	All skills from Comp Sci I H and Computer Science II AP	General Review for AP Exam slides Personal Review Sheet assignment FRQs from past AP exams (3 full sets of 4 FRQs + miscellaneous FRQs) 2015 Released AP Exam - multiple choice	Personal Review Sheet assignment FRQs from past AP exams (3 full sets of 4 FRQs + miscellaneous FRQs)

	Essential Questions	Content	Skills	Activities	Assessment / Evidence of Learning
AP EXAM PREP (continued)				2020 Practice AP Exam - multiple choice and FRQ AP Test Taking Tips sheet CollegeBoard's AP Computer Science A's Course Exam Description's multiple choice questions	2015 Released AP Exam - multiple choice 2020 Practice AP Exam - multiple choice and FRQ CollegeBoard's AP Computer Science A's Course Exam Description's multiple choice questions
POST-EXAM FINAL PROJECT	N/A			Final Project description document Final Project research Final Project idea form Final Project progress journal document Final Project learning summary document Final Project resources document	Final Project progress journal document Final Project learning summary document Final Project resources document