ADVANCED PLACEMENT COMPUTER SCIENCE A	
CURRICULUM/CONTENT AREA	**COURSE LENGTH**
Computer Science	*2 terms*
GRADE LEVEL	**DATE LAST REVIEWED**
9-12	2023
PREREQUISITE(s) ***if applicable***	**BOARD APPROVAL DATE**
Successful completion of Algebra I is highly recommended	*9/10/2024*
PRIMARY RESOURCE ***if applicable***	
College Board AP Classroom *Java Concepts, 5th Edition (2007), John Wiley & Sons*	

DESIRED RESULTS

COURSE DESCRIPTION AND PURPOSE

Computer science involves problem-solving, hardware, and algorithms that help people utilize computers and incorporate multiple perspectives to address real-world problems in contemporary life. As the study of computer science continues to evolve, students in AP Computer Science A will discover the power of computer science through rewarding yet challenging concepts. Through this course, students are prepared to take the AP Computer Science A exam, and may opt to work towards industry recognized certifications in Java programming or software development.

WEIGHTED GRADE
Opportunity to earn industry recognized credentials:
IT Specialist Suite

BIG IDEAS *Students will understand that...*	**ESSENTIAL QUESTIONS** *Students will keep considering...*
BIG IDEA 1: MODULARITY (MOD) Incorporating elements of abstraction, by breaking problems down into interacting pieces, each with their own purpose, makes writing complex programs easier. Abstracting simplifies concepts and processes by looking at the big picture rather than being overwhelmed by the details. Modularity in object-oriented programming allows us to use abstraction to break complex programs down into individual classes and methods.	How can we use programs to solve problems?
	How can we simulate election results using existing program code?
	How can using a model of traffic patterns improve travel time?
	How can programs leverage volcano data to make predictions about the date of the next eruption?
	Why is an ArrayList more appropriate for storing your music playlist, while an array might be more appropriate for storing your class schedule?
	Why might you want to use a 2D array to store the spaces on a game board or the pixels in a picture, rather than a 1D array or ArrayList?
	How might the use of inheritance help in writing a program that simulates crops being grown in a virtual world?
	How does inheritance make programs more versatile?
	What real-world processes do you follow that are recursive in nature?
	Why do programmers sometimes prefer using recursive solutions when sorting data in a large data set?
BIG IDEA 2: VARIABLES (VAR) Information used as a basis for reasoning, discussion, or calculation is referred to as data. Programs rely on variables to store data, on data structures to organize multiple values when program complexity increases, and on algorithms to sort, access, and manipulate this data. Variables create data abstractions, as they can represent a set of possible values or a group of related values.	In what ways are numbers used in the programs and apps you use most often?
	How are appropriate variables chosen to represent a remote control?
	How can programs be written to protect your bank account balance from being inadvertently changed?
	How can knowing standard algorithms be useful when solving new problems?
	How can we use statement execution counts to choose appropriate algorithms?

	Why does the order in which elements are accessed in 2D array traversal matter in some situations?
BIG IDEA 3: CONTROL (CON) Doing things in order, making decisions, and doing the same process multiple times are represented in code by using control structures and specifying the order in which instructions are executed. Programmers need to think algorithmically in order to define and interpret processes that are used in a program.	How are mathematical concepts being used in the programs and apps that you use most often?
	How do the games we play simulate randomness?
	How can you use different conditional statements to write a pick-your-own-path interactive story?
	Why is selection a necessary part of programming languages?
	How does iteration improve programs and reduce the amount of program code necessary to complete a task?
	What situations would warrant the use of one type of loop over another?
	What responsibility do programmers have for the consequences of programs they
	What personal data is currently being collected, and how?
BIG IDEA 4: IMPACT OF COMPUTING (IOC) Computers and computing have revolutionized our lives. To use computing safely and responsibly, we need to be aware of privacy, security, and ethical issues. As programmers, we need to understand how our programs will be used and be responsible for the consequences.	How are mathematical concepts being used in the programs and apps that you use most often?
COMPUTATIONAL THINKING PRACTICES *Students will be skilled at...*	**SKILLS (LEARNING TARGETS)**
Practice 1: PROGRAM DESIGN & ALGORITHM DEVELOPMENT **Determine required code segments to produce a given output.**	**1.A I can determine an appropriate program design to solve a problem or accomplish a task**
	1.B I can determine code that would be used to complete code segments.
	1.C I can determine code that would be used to interact with completed program code.
Practice 2: CODE LOGIC **Determine the output, value, or result of a given program code given initial values.**	**2.A I can apply the meaning of specific operators.**
	2.B I can determine the result or output based on statement execution order in a code segment without method calls (other than output).
	2.C I can determine the result or output based on the statement execution order in a code segment containing method calls.
	2.D I can determine the number of times a code segment will execute.
Practice 3: CODE IMPLEMENTATION **Write and implement program code.**	**3.A I can write program code to create objects of a class and call methods.**
	3.B I can write program code to define a new type by creating a class.
	3.C I can write program code to satisfy method specifications using expressions, conditional statements, and iterative statements.
	3.D I can write program code to create, traverse, and manipulate elements in 1D array or ArrayList objects.
	3.E I can write program code to create, traverse, and manipulate elements in 2D array objects.
Practice 4: CODE TESTING **Analyze program code for correctness, equivalence, and errors.**	**4.A I can use test-cases to find errors or validate results.**
	4.B I can identify errors in program code.
	4.C I can determine if two or more code segments yield equivalent results.
Practice 5: DOCUMENTATION **Describe the behaviors and condition that produce identifed results in a program.**	**5.A I can describe the behavior of a given segment of program code.**
	5.B I can explain why a code segment will not compile or work as intended.
	5.C I can explain how the result of program code changes, given a change to the initial code.
	5.D I can describe the initial conditions that must be met for a program segment to work as intended or described.
Potential INDUSTRY-RECOGNIZED CREDENTIALS (IRCs) **Opportunties associated with the course**	**Potential WORK BASED LEARNING (WBL) opportunities associated with the course**

Certifport: IT Specialist- Java and/or software development	
Potential DUAL CREDIT Opportunties associated with the course	
Successful completion of the AP exam may lead to college credit.	

UNIT 1: PRIMITIVE TYPES

This unit introduces students to the Java programming language and the use of classes, providing students with a firm foundation of concepts that will be leveraged and built upon in all future units. Students will focus on writing the main method and will start to call preexisting methods to produce output. The use of preexisting methods for input is not prescribed in the course; however, input is a necessary part of any computer science course so teachers will need to determine how they will address this in their classrooms. Students will start to learn about three built-in data types and learn how to create variables, store values, and interact with those variables using basic operations. The ability to write expressions is essential to representing the variability of the real world in a program and will be used in all future units. Primitive data is one of two categories of variables covered in this course.

STAGE 1: Desired Unit Results *What will students understand as a result of the unit?*		**STAGE 2: Assessment Evidence** *By what criteria will performances of understanding be assessed? Through what authentic performance tasks will students demonstrate the desired unit results?*
BIG IDEAS *Students will understand that...*	**ESSENTIAL QUESTIONS** *Students will keep considering...*	**Success Criteria with Big Ideas & Skills** *The criteria for evaluating performance on standards is constant.*
BIG IDEA 1: MODULARITY (MOD)	How can we use programs to solve problems?	Through rubrics and exemplars, students understand, goal set, and reflect on success criteria. Feedback is given through small group/ one-on-one conferring and through full class modeling.
BIG IDEA 2: VARIABLES (VAR)	In what ways are numbers used in the programs and apps you use most often?	Authentic Assessment: Industry Recognized Certification option in Java and/or Software Development (ongoing throughout the course).
BIG IDEA 3: CONTROL (CON)	How are mathematical concepts being used in the programs and apps that you use most often?	AP Classroom: Personal Progress Checks, AP Question Bank
BIG IDEA 4: IMPACT OF COMPUTING (IOC)		Preparing for AP Exam: This unit provides a lot of the foundational content and skills that students will continue to draw on throughout the course. While much of what is covered in this unit is not explicitly assessed on the AP Exam, this content exists in the program code of nearly every assessment question. Checks for understanding: Students will spend adequate time practicing writing expressions and applying meaning to specific operators during this unit.
Topic	**Skills (Learning Targets)** ***I can...***	
1.1 Why Programming? Why Java?	2.B Determine the result or output based on statement execution order in a code segment without method calls (other than output). 4.B Identify errors in program code.	
1.2 Variables and Data Types	1.A Determine an appropriate program design to solve a problem or accomplish a task (not assessed). 1.B Determine code that would be used to complete code segments.	
1.3 Expressions and Assignment Statements	1.B Determine code that would be used to complete code segments. 2.A Apply the meaning of specific operators.	
1.4 Compound Assignment Operators	2.B Determine the result or output based on statement execution order in a code segment without method calls (other than output). 5.A Describe the behavior of a given segment of program code.	
1.5 Casting and Ranges of Variables	2.B Determine the result or output based on statement execution order in a code segment without method calls (other than output). 5.B Explain why a code segment will not compile or work as intended.	

UNIT 2: USING OBJECTS

In the first unit, students used primitive types to represent real-world data and determined how to use them in arithmetic expressions to solve problems. This unit introduces a new type of data: reference data. Reference data allows real-world objects to be represented in varying degrees specific to a programmer's purpose. This unit builds on students' ability to write expressions by introducing them to Math class methods to write expressions for generating random numbers and other more complex operations. In addition, strings and the existing methods within the String class are an important topic within this unit. Knowing how to declare variables or call methods on objects is necessary throughout the course but will be very important in Units 5 and 9 when teaching students how to write their own classes and about inheritance relationships.

STAGE 1: Desired Unit Results *What will students understand as a result of the unit?*		**STAGE 2: Assessment Evidence** *By what criteria will performances of understanding be assessed? Through what authentic performance tasks will students demonstrate the desired unit results?*
BIG IDEAS *Students will understand that...*	**ESSENTIAL QUESTIONS** *Students will keep considering...*	**Success Criteria with Big Ideas & Skills** *The criteria for evaluating performance on standards is constant.*
BIG IDEA 1: MODULARITY (MOD)	How can we simulate election results using existing program code?	Through rubrics and exemplars, students understand, goal set, and reflect on success criteria. Feedback is given through small group/ one-on-one conferring and through full class modeling.
BIG IDEA 2: VARIABLES (VAR)	How are appropriate variables chosen to represent a remote control?	Authentic Assessment: Industry Recognized Certification option in Java and/or Software Development (ongoing throughout the course).
BIG IDEA 3: CONTROL (CON)	How do the games we play simulate randomness?	AP Classroom: Personal Progress Checks, AP Question Bank Preparing for AP Exam: During the free-response portion of the exam, students will be required to call methods of classes that they haven't been exposed to prior to the exam. Students should get plenty of practice identifying the proper parameters to use when calling methods of classes that are provided to them. Often, students struggle with free-response questions that require them to work with the String class. Using the Java Quick Reference (p. 219) regularly during class will help students become more familiar with this resource prior to the exam. Paying close attention to the method descriptions will ensure that students use the correct type and order of parameters when calling String methods. Practice close reading techniques with students prior to the exam, such as underlining keywords, return types, and parameters. Students have approximately 20 minutes to read, process, and answer each of the four free-response questions. These close reading techniques are valuable in helping students process the questions quickly without inadvertently missing key information.
Topic	**Skills (Learning Targets)** ***I can...***	
2.1 Objects: Instances of Classes	5.A Describe the behavior of a given segment of program code.	
2.2 Creating and Storing Objects (Instantiation)	1.C Determine code that would be used to interact with completed program code.	
2.3 Calling a Void Method	1.C Determine code that would be used to interact with completed program code. 3.A Write program code to create objects of a class and call methods	
2.4 Calling a Void Method with Parameters	2.C Determine the result or output based on the statement execution order in a code segment containing method calls. 3.A Write program code to create objects of a class and call methods.	
2.5 Calling a Non-void Method	1.C Determine code that would be used to interact with completed program code. 3.A Write program code to create objects of a class and call methods.	
2.6 String Objects: Concatenation, Literals, and More	2.A Apply the meaning of specific operators.	
2.7 String Methods	2.C Determine the result or output based on the statement execution order in a code segment containing method calls. 3.A Write program code to create objects of a class and call methods	
2.8 Wrapper Classes: Integer and Double	2.C Determine the result or output based on the statement execution order in a code segment containing method calls.	
2.9 Using the Math Class	1.B Determine code that would be used to complete code segments. 3.A Write program code to create objects of a class and call methods.	

UNIT 3: BOOLEAN EXPRESSIONS AND IF STATEMENTS

This unit focuses on iteration using while and for loops. As you saw in Unit 3, Boolean expressions are useful when a program needs to perform different operations under different conditions. Boolean expressions are also one of the main components in iteration. This unit introduces several standard algorithms that use iteration. Knowledge of standard algorithms makes solving similar problems easier, as algorithms can be modified or combined to suit new situations. Iteration is used when traversing data structures such as arrays, ArrayLists, and 2D arrays. In addition, it is a necessary component of several standard algorithms, including searching and sorting, which will be covered in later units.

STAGE 1: Desired Unit Results *What will students understand as a result of the unit?*		**STAGE 2: Assessment Evidence** *By what criteria will performances of understanding be assessed? Through what authentic performance tasks will students demonstrate the desired unit results?*
BIG IDEAS *Students will understand that...*	**ESSENTIAL QUESTIONS** *Students will keep considering...*	**Success Criteria with Big Ideas & Skills** *The criteria for evaluating performance on standards is constant.*
BIG IDEA 3: CONTROL (CON)	How can you use different conditional statements to write a pick-your-own-path interactive story? Why is selection a necessary part of programming languages?	Through rubrics and exemplars, students understand, goal set, and reflect on success criteria. Feedback is given through small group/ one-on-one conferring and through full class modeling. Authentic Assessment: Industry Recognized Certification option in Java and/or Software Development (ongoing throughout the course). AP Classroom: Personal Progress Checks, AP Question Bank Preparing for AP Exam: The study of computer science involves the analysis of program code. On the multiplechoice section of the exam, students will be asked to determine the result of a given program code segment based on specific input and on the behavior of program code in general. Students should be able to determine the result of program code that uses conditional statements and nested conditional statements to represent nonlinear processes in a program. Often, students will write program code that mishandles one of the given conditions. The ability to trace program code can be valuable when testing programs to ensure that all conditions are met. Testing for the different expected behaviors of conditional statements is a critical part of program development and is useful when writing program code or analyzing given code segments. Students should develop sample test cases to illustrate each unique behavior to aid in finding errors and validating results.
Topic	**Skills (Learning Targets)** ***I can...***	
3.1 Boolean Expressions	2.A Apply the meaning of specific operators.	
3.2 i f Statements and Control Flow	2.B Determine the result or output based on statement execution order in a code segment without method calls (other than output). 3.C Write program code to satisfy method specifications using expressions, conditional statements, and iterative statements.	
3.3 i f-else Statements	3.C Write program code to satisfy method specifications using expressions, conditional statements, and iterative statements. 4.A Use test-cases to find errors or validate results.	
3.4 else if Statements	3.C Write program code to satisfy method specifications using expressions, conditional statements, and iterative statements. 4.C Determine if two or more code segments yield equivalent results	
3.5 Compound Boolean Expressions	2.B Determine the result or output based on statement execution order in a code segment without method calls (other than output). 3.C Write program code to satisfy method specifications using expressions, conditional statements, and iterative statements.	
3.6 Equivalent Boolean Expressions	4.C Determine if two or more code segments yield equivalent results.	
3.7 Comparing Objects	2.C Determine the result or output based on the statement execution order in a code segment containing method calls. 3.A Write program code to create objects of a class and call methods.	

UNIT 4: ITERATION

This unit focuses on iteration using while and for loops. As seenin Unit 3, Boolean expressions are useful when a program needs to perform different operations under different conditions. Boolean expressions are also one of the main components in iteration. This unit introduces several standard algorithms that use iteration. Knowledge of standard algorithms makes solving similar problems easier, as algorithms can be modified or combined to suit new situations. Iteration is used when traversing data structures such as arrays, ArrayLists, and 2D arrays. In addition, it is a necessary component of several standard algorithms, including searching and sorting, which will be covered in later units.

STAGE 1: Desired Unit Results *What will students understand as a result of the unit?*		**STAGE 2: Assessment Evidence** *By what criteria will performances of understanding be assessed? Through what authentic performance tasks will students demonstrate the desired unit results?*
BIG IDEAS *Students will understand that...*	**ESSENTIAL QUESTIONS** *Students will keep considering...*	**Success Criteria with Big Ideas & Skills** *The criteria for evaluating performance on standards is constant.*
BIG IDEA 1: CONTROL (CON)	How does iteration improve programs and reduce the amount of program code necessary to complete a task? What situations would warrant the use of one type of loop over another?	Through rubrics and exemplars, students understand, goal set, and reflect on success criteria. Feedback is given through small group/ one-on-one conferring and through full class modeling. Authentic Assessment: Industry Recognized Certification option in Java and/or Software Development (ongoing throughout the course). AP Classroom: Personal Progress Checks, AP Question Bank Preparing for AP Exam: While the concept of iteration is tested in isolation on the multiple-choice exam, it is a foundational concept that students will use along with other topics, such as data structures, on free-response questions. Often, students struggle in situations that warrant variation in the Boolean condition of loops, such as when they want to terminate a loop early. Early termination of a loop requires the use of conditional statements within the body of the loop. If the order of the program statements is incorrect, the early termination may be triggered too early or not at all. Provide students with practice ordering statements by giving them strips of paper, each with a line of program code that can be used to assemble the correct and incorrect solutions. Ask them to reassemble the code and trace it to see if it is correct. Using manipulatives in this way makes it easier for students to rearrange the order of the program code to determine if it is in the correct order.
Topic	**Skills (Learning Targets)** ***I can...***	
4.1 while Looops	1.B Determine code that would be used to complete code segments. 2.B Determine the result or output based on statement execution order in a code segment without method calls (other than output). 3.C Write program code to satisfy method specifications using expressions, conditional statements, and iterative statements.	
4.2 for Loops	3.C Write program code to satisfy method specifications using expressions, conditional statements, and iterative statements. 4.C Determine if two or more code segments yield equivalent results. 5.C Explain how the result of program code changes, given a change to the initial code.	
4.3 Developing Algorithms Using Strings	2.C Determine the result or output based on the statement execution order in a code segment containing method calls. 3.C Write program code to satisfy method specifications using expressions, conditional statements, and iterative statements	
4.4 Nested Iteration	1.B Determine code that would be used to complete code segments. 3.C Write program code to satisfy method specifications using expressions, conditional statements, and iterative statements. 5.C Explain how the result of program code changes, given a change to the initial code.	
4.5 Informal Code Analysis	2.D Determine the number of times a code segment will execute	

UNIT 5: WRITING CLASSES

This unit will pull together information from all previous units to create new, user-defined reference data types in the form of classes. The ability to accurately model real-world entities in a computer program is a large part of what makes computer science so powerful. This unit focuses on identifying appropriate behaviors and attributes of real-world entities and organizing these into classes. Students will build on what they learn in this unit to represent relationships between classes through hierarchies, which appear in Unit 9. The creation of computer programs can have extensive impacts on societies, economies, and cultures. The legal and ethical concerns that come with programs and the responsibilities of programmers are also addressed in this unit.

STAGE 1: Desired Unit Results *What will students understand as a result of the unit?*		**STAGE 2: Assessment Evidence** *By what criteria will performances of understanding be assessed? Through what authentic performance tasks will students demonstrate the desired unit results?*
BIG IDEAS *Students will understand that...*	**ESSENTIAL QUESTIONS** *Students will keep considering...*	**Success Criteria with Big Ideas & Skills** *The criteria for evaluating performance on standards is constant.*
BIG IDEA 1: MODULARITY (MOD)	How can using a model of traffic patterns improve travel time?	Through rubrics and exemplars, students understand, goal set, and reflect on success criteria. Feedback is given through small group/ one-on-one conferring and through full class modeling.
BIG IDEA 2: VARIABLES (VAR)	How can programs be written to protect your bank account balance from being inadvertently changed?	Authentic Assessment: Industry Recognized Certification option in Java and/or Software Development (ongoing throughout the course).
BIG IDEA 3: IMPACT OF COMPUTING (IOC)	What responsibility do programmers have for the consequences of programs they create, whether intentional or not?	AP Classroom: Personal Progress Checks, AP Question Bank Preparing for AP Exam: On the free-response section of the exam, students are required to design program code that demonstrates the functionality available for an object of a new class, based on the prompt's specification. This process includes identifying the attributes (data) that define an object of a class and the behaviors (methods) that define what an object of a class can do. Students should have many opportunities in this course to design their own classes based on program specifications and on observations of real-world objects. Once the new class is designed, students should be able to implement program code to create a new type by creating a class. The behaviors of an object are expressed through writing methods in the class, which include expressions, conditional statements, and iterative statements. By being able to create their own classes, programmers are not limited to the existing classes provided within the Java libraries and can therefore represent their own ideas through classes.
Topic	**Skills (Learning Targets)** ***I can...***	
5.1 Anatomy of a Class	1.A Determine an appropriate program design to solve a problem or accomplish a task (not assessed). 1.B Determine code that would be used to complete code segments.	
5.2 Constructors	1.C Determine code that would be used to interact with completed program code. 3.B Write program code to define a new type by creating a class	
5.3 Documentation with Comments	5.D Describe the initial conditions that must be met for a program segment to work as intended or described.	
5.4 Accessor Methods	3.B Write program code to define a new type by creating a class. 5.B Explain why a code segment will not compile or work as intended	
5.5 Mutator Methods	3.B Write program code to define a new type by creating a class. 4.B Identify errors in program code	
5.6 Writing Methods	1.B Determine code that would be used to complete code segments. 3.B Write program code to define a new type by creating a class.	
5.7 Static Variables and Methods	3.B Write program code to define a new type by creating a class. 5.A Describe the behavior of a given segment of program code.	

UNIT 6: ARRAY

This unit focuses on data structures, which are used to represent collections of related data using a single variable rather than multiple variables. Using a data structure along with iterative statements with appropriate bounds will allow for similar treatment to be applied more easily to all values in the collection. Just as there are useful standard algorithms when dealing with primitive data, there are standard algorithms to use with data structures. In this unit, we apply standard algorithms to arrays; however, these same algorithms are used with ArrayLists and 2D arrays as well. Additional standard algorithms, such as standard searching and sorting algorithms, will be covered in the next unit.

STAGE 1: Desired Unit Results *What will students understand as a result of the unit?*		**STAGE 2: Assessment Evidence** *By what criteria will performances of understanding be assessed? Through what authentic performance tasks will students demonstrate the desired unit results?*
BIG IDEAS *Students will understand that...*	**ESSENTIAL QUESTIONS** *Students will keep considering...*	**Success Criteria with Big Ideas & Skills** *The criteria for evaluating performance on standards is constant.*
BIG IDEA 1: VARIABLES (VAR)	How can programs leveage volcano data to make predictions about the date of the next eruption?	Through rubrics and exemplars, students understand, goal set, and reflect on success criteria. Feedback is given through small group/ one-on-one conferring and through full class modeling.
BIG IDEA 2: CONTROL (CON)	How can knowing standard algorithms be useful when solving new problems?	Authentic Assessment: Industry Recognized Certification option in Java and/or Software Development (ongoing throughout the course).
		AP Classroom: Personal Progress Checks, AP Question Bank
Topic	**Skills (Learning Targets)** ***I can...***	Preparing for AP Exam: A specific iterative structure is commonly used to traverse an array: starting at the beginning and moving toward the last element in the array. When students are asked to determine the result of program code that contains arrays, they often use this same iterative structure. Knowing this iterative structure can be helpful when students use tracing to determine what algorithm is doing. Students can follow this traversal for the first few iterations and apply that pattern to the remaining elements in the array. When preparing for the free-response questions, students should be come familiar with how existing algorithms work, rather than just memorizing the program code. Students will write out an algorithm on paper and test it using manipulatives. This allows students to experience the algorithm on a deeper level than if they simply program it. A strong understanding of how existing algorithms work allows programmers to make modifications to those algorithms to accomplish similar tasks.
6.1 Array Creation and Access	1.C Determine code that would be used to interact with completed program code. 3.D Write program code to create, traverse, and manipulate elements in 1D array or ArrayList objects.	
6.2 Traversing Arrays	2.B Determine the result or output based on statement execution order in a code segment without method calls (other than output). 3.D Write program code to create, traverse, and manipulate elements in 1D array or ArrayList objects. 4.B Identify errors in program code.	
6.3 Enhanced for Loop for Arrays	3.D Write program code to create, traverse, and manipulate elements in 1D array or ArrayList objects. 4.C Determine if two or more code segments yield equivalent results.	
6.4 Developing Algorithms Using Arrays	1.B Determine code that would be used to complete code segments. 3.D Write program code to create, traverse, and manipulate elements in 1D array or ArrayList objects. 5.D Describe the initial conditions that must be met for a program segment to work as intended or described.	

UNIT 7: ARRAYLIST

As students learned in Unit 6, data structures are helpful when storing multiple related data values. Arrays have a static size, which causes limitations related to the number of elements stored, and it can be challenging to reorder elements stored in arrays. The ArrayList object has a dynamic size, and the class contains methods for insertion and deletion of elements, making reordering and shifting items easier. Deciding which data structure to select becomes increasingly important as the size of the data set grows, such as when using a large real-world data set. In this unit, students will also learn about privacy concerns related to storing large amounts of personal data and about what can happen if such information is compromised.

STAGE 1: Desired Unit Results *What will students understand as a result of the unit?*		**STAGE 2: Assessment Evidence** *By what criteria will performances of understanding be assessed? Through what authentic performance tasks will students demonstrate the desired unit results?*
BIG IDEAS *Students will understand that...*	**ESSENTIAL QUESTIONS** *Students will keep considering...*	**Success Criteria with Big Ideas & Skills** *The criteria for evaluating performance on standards is constant.*
BIG IDEA 1: VARIABLES (VAR)	Why is an ArrayList more appropriate for storing your music playlist, while an array might be more appropriate for storing your class schedule?	Through rubrics and exemplars, students understand, goal set, and reflect on success criteria. Feedback is given through small group/ one-on-one conferring and through full class modeling.
BIG IDEA 2: CONTRON (CON)	How can we use statement execution counts to choose appropriate algorithms?	Authentic Assessment: Industry Recognized Certification option in Java and/or Software Development (ongoing throughout the course).
BIG IDEA 3: IMPACT OF COMPUTING (IOC)	What personal data is currently being collected, and how?	AP Classroom: Personal Progress Checks, AP Question Bank
Topic	**Skills (Learning Targets)** ***I can...***	Preparing for AP Exam: When writing solutions to free-response questions that involve the use of an ArrayList, students are often asked to insert or delete elements from the ArrayList. In these cases, adjustments will need to be made to the loop counter to account for skipping an element or attempting to access elements that no longer exist. Students may also be asked to traverse multiple data structures simultaneously. These data structures may be a mixture of array and ArrayList objects. As such, it is very easy for students to become confused about how elements are accessed and manipulated within each structure. Additionally, the size of the data structures may not be the same. In these situations, it is best to have different index variables and bounds checks for each structure to avoid accessing elements that are out-of-bounds.
7.1 Introduction to ArrayList	1.B Determine code that would be used to complete code segments. 3.D Write program code to create, traverse, and manipulate elements in 1D array or ArrayList objects.	
7.2 ArrayList Methods	2.C Determine the result or output based on the statement execution order in a code segment containing method calls. 3.D Write program code to create, traverse, and manipulate elements in 1D array or ArrayList objects.	
7.3 Traversing ArrayLists	2.C Determine the result or output based on the statement execution order in a code segment containing method calls. 3.D Write program code to create, traverse, and manipulate elements in 1D array or ArrayList objects.	
7.4 Developing Algorithms Using ArrayLists	3.D Write program code to create, traverse, and manipulate elements in 1D array or ArrayList objects. 4.A Use test-cases to find errors or validate results.	
7.5 Searching	3.D Write program code to create, traverse, and manipulate elements in 1D array or ArrayList objects. 5.C Explain how the result of program code changes, given a change to the initial code.	
7.6 Sorting	2.D Determine the number of times a code segment will execute.	
7.7 Ethical Issues Around Data Collection		

UNIT 8: 2D ARRAY

In Unit 6, students learned how 1D arrays store large amounts of related data. These same concepts will be implemented with two-dimensional (2D) arrays in this unit. A 2D array is most suitable to represent a table. Each table element is accessed using the variable name and row and column indices. Unlike 1D arrays, 2D arrays require nested iterative statements to traverse and access all elements. The easiest way to accomplished this is in row-major order, but it is important to cover additional traversal patterns, such as back and forth or column-major.

STAGE 1: Desired Unit Results *What will students understand as a result of the unit?*		**STAGE 2: Assessment Evidence** *By what criteria will performances of understanding be assessed? Through what authentic performance tasks will students demonstrate the desired unit results?*
BIG IDEAS *Students will understand that...*	**ESSENTIAL QUESTIONS** *Students will keep considering...*	**Success Criteria with Big Ideas & Skills** *The criteria for evaluating performance on standards is constant.*
BIG IDEA 1: VARIABLES (VAR)	Why might you want to use a 2D array to store the spaces on a game board or the pixels in a picture, rather than a 1D array or ArrayList?	Through rubrics and exemplars, students understand, goal set, and reflect on success criteria. Feedback is given through small group/ one-on-one conferring and through full class modeling.
BIG IDEA 2: CONTROL (CON)	Why does the order in which elements are accessed in 2D array traversal matter in some situations?	Authentic Assessment: Industry Recognized Certification option in Java and/or Software Development (ongoing throughout the course).
		AP Classroom: Personal Progress Checks, AP Question Bank
Topic	**Skills (Learning Targets)** ***I can...***	Preparing for AP Exam: The free-response portion of the exam always includes one question that requires students to write program code involving 2D arrays. Because 2D arrays are arrays where each element is an array, it is not uncommon for the question to require students to write solutions involving array or ArrayList objects as well. While there is a specific nested structure to traverse elements in a 2D array in rowmajor order, this structure can be modified to traverse 2D arrays in other ways, such as column-major, by switching the nested iterative statements. Additional modifications can be made to traverse rows or columns in different ways, such as back and forth or up and down. However, when making these adjustments, students often neglect to adjust the bounds of the iterative statements appropriately. Students should practice traversing 2D arrays in these nonstandard ways, being sure to test the boundary conditions of the iterative statements, to be prepared for this type of freeresponse question.
8.1 2D Arrays	1.B Determine code that would be used to complete code segments. 1.C Determine code that would be used to interact with completed program code. 3.E Write program code to create, traverse, and manipulate elements in 2D array objects.	
8.2 Traversing 2D Arrays	2.B Determine the result or output based on statement execution order in a code segment without method calls (other than output). 2.D Determine the number of times a code segment will execute. 3.E Write program code to create, traverse, and manipulate elements in 2D array objects. 4.A Use test-cases to find errors or validate results.	

UNIT 9: INHERITANCE

Creating objects, calling methods on the objects created, and being able to define a new data type by creating a class are essential understandings before moving into this unit. One of the strongest advantages of Java is the ability to categorize classes into hierarchies through inheritance. Certain existing classes can be extended to include new behaviors and attributes without altering existing code. These newly created classes are called subclasses. In this unit, students will learn how to recognize common attributes and behaviors that can be used in a superclass and will then create a hierarchy by writing subclasses to extend a superclass. Recognizing and utilizing existing hierarchies will help students create more readable and maintainable programs.

STAGE 1: Desired Unit Results *What will students understand as a result of the unit?*		**STAGE 2: Assessment Evidence** *By what criteria will performances of understanding be assessed? Through what authentic performance tasks will students demonstrate the desired unit results?*
BIG IDEAS *Students will understand that...*	**ESSENTIAL QUESTIONS** *Students will keep considering...*	**Success Criteria with Big Ideas & Skills** *The criteria for evaluating performance on standards is constant.*
BIG IDEA 1: MODULARITY (MOD)	How might the use of inheritance help in writing a program that simulates crops being grown in a virtual world? How does inheritance make programs more versatile?	Through rubrics and exemplars, students understand, goal set, and reflect on success criteria. Feedback is given through small group/ one-on-one conferring and through full class modeling. Authentic Assessment: Industry Recognized Certification option in Java and/or Software Development (ongoing throughout the course). AP Classroom: Personal Progress Checks, AP Question Bank Preparing for AP Exam: One question on the free-response section of the exam will require students to write a class. This class could be part of an inheritance relationship. When overriding superclass methods in a subclass, method signatures must be the same. This includes the number, type, and order of any parameters of the overridden method. It is important for students to recognize when a method should be overridden, as well as when they can and should use methods from the superclass. Students who duplicate code unnecessarily will not earn full points on this question. Students will be asked to analyze program code that uses inheritance. In many cases, students struggle with determining whether a method is available to be called by an object of a class. When a method is called on a subclass object, the method that is executed is determined during run-time. If the subclass does not contain the called method, the superclass method will automatically be executed.
Topic	**Skills (Learning Targets)** ***I can...***	
9.1 Creating Superclasses and Subclasses	1.A Determine an appropriate program design to solve a problem or accomplish a task (not assessed). 3.B Write program code to define a new type by creating a class.	
9.2 Writing Constructors for Subclasses	3.B Write program code to define a new type by creating a class. 5.A Describe the behavior of a given segment of program code.	
9.3 Overriding Methods	3.B Write program code to define a new type by creating a class. 5.D Describe the initial conditions that must be met for a program segment to work as intended or described.	
9.4 super Keyword	1.C Determine code that would be used to interact with completed program code. 3.B Write program code to define a new type by creating a class	
9.5 Creating Reference Using Inheritance Hierarchies	3.A Write program code to create objects of a class and call methods. 5.B Explain why a code segment will not compile or work as intended.	
9.6 Polymorphism	3.A Write program code to create objects of a class and call methods. 5.B Explain why a code segment will not compile or work as intended	
9.7 Object Superclass	1.C Determine code that would be used to interact with completed program code. 3.B Write program code to define a new type by creating a class	

UNIT 10: RECURSION

Sometimes a problem can be solved by solving smaller or simpler versions of the same problem rather than attempting an iterative solution. This is called recursion, and it is a powerful math and computer science idea. In this unit, students will revisit how control is passed when methods are called, which is necessary knowledge when working with recursion. Tracing skills introduced in Unit 2 are helpful for determining the purpose or output of a recursive method. In this unit, students will learn how to write simple recursive methods and determine the purpose or output of a recursive method by tracing.

STAGE 1: Desired Unit Results *What will students understand as a result of the unit?*		**STAGE 2: Assessment Evidence** *By what criteria will performances of understanding be assessed? Through what authentic performance tasks will students demonstrate the desired unit results?*
BIG IDEAS *Students will understand that...*	**ESSENTIAL QUESTIONS** *Students will keep considering...*	**Success Criteria with Big Ideas & Skills** *The criteria for evaluating performance on standards is constant.*
		Through rubrics and exemplars, students understand, goal set, and reflect on success criteria. Feedback is given through small group/ one-on-one conferring and through full class modeling.
		Authentic Assessment: Industry Recognized Certification option in Java and/or Software Development (ongoing throughout the course).
BIG IDEA 3: CONTROL (CON)	How can you use different conditional statements to write a pick-your-own-path interactive story? Why is selection a necessary part of programming languages?	AP Classroom: Personal Progress Checks, AP Question Bank Preparing for AP Exam: Recursion is primarily assessed through the multiple-choice section of the exam. Students are often asked to determine the result of a specific call to a recursive method or to describe the behavior of a recursive method. A call to a recursive method is just like a call to a nonrecursive method. Because a method is an abstraction of a process that has a specific result for each varied input, using a specific input value provides an understanding of how the method functions for that input. By understanding several instances of the method call, students can abstract and generalize the method's overall purpose or process. While students will not be required to write a recursive solution in the free-response section, recursive solutions are often a more straightforward way of writing the solutions than iterative designs. Writing recursive solutions and analyzing calls to recursive methods help engage students with all aspects of recursive methods and provide them with a deeper understanding of how recursion works.
Topic	**Skills (Learning Targets)** ***I can...***	
10.1 Recursion	1.B Determine code that would be used to complete code segments. 5.A Describe the behavior of a given segment of program code.	
10.2 Recursive Searching and Sorting	2.C Determine the result or output based on the statement execution order in a code segment containing method calls. 2.D Determine the number of times a code segment will execute.	