TRUMBULL PUBLIC SCHOOLS

Trumbull, Connecticut

Advanced Placement Computer Science A

Mathematics Department

2019

Curriculum Writing Team

Katie Laird	**Department Chair**
Scott Kaminski	**Teacher**

Jonathan S. Budd, Ph.D., Assistant Superintendent of Curriculum, Instruction, & Assessments

AP Computer Science A
Table of Contents

CORE VALUES AND BELIEFS

The Trumbull High School community engages in an environment conducive to learning which believes that all students will **read and write effectively**, therefore communicating in an articulate and coherent manner. All students will participate in activities **that present problem-solving through critical thinking**. Students will use technology as a tool applying it to decision making. We believe that by fostering self-confidence, self-directed and student-centered activities, we will promote **independent thinkers and learners**. We believe **ethical conduct** to be paramount in sustaining the welcoming school climate that we presently enjoy.

Approved 8/26/2011

INTRODUCTION & PHILOSOPHY

AP Computer Science A (CSA) introduces students to computer science through programming. Fundamental topics in this course include the design of solutions to problems, the use of data structures to organize large sets of data, the development and implementation of algorithms to process data and discover new information, the analysis of potential solutions, and the ethical and social implications of computing systems. The course emphasizes object-oriented programming and design using the Java programming language.

The AP CSA course requires that potential solutions of problems be written in the Java programming language. In addition to precision of expression, Java supports important elements of problem solving, including object-orientation, abstraction, and encapsulation. The use of Java allows students to test potential solutions to problems by running programs. The Java programming language is extensive, with far more features than could be covered in a single introductory course. So, the AP CSA course and College Board Examination cover only a subset of Java.

COURSE GOALS

AP Computer Science Principles satisfies three sets of goals:

- The Learning Objectives of the 2017 College Board Curriculum Framework for Advanced Placement Computer Science Principles;
- The Standards of the 2017 Computer Science Teachers' Association (CSTA); and
- The Standards of the 2016 International Society for Technology in Education Standards.

Each Unit presents the Goals appropriate for that Unit.

COURSE ENDURING UNDERSTANDINGS

Students will understand that . . .

- the goal of designing a piece of software (a computer program) is to correctly solve the given problem. At the same time, this goal should encompass specifying and designing a program that is understandable, can be adapted to changing circumstances, and has the potential to be reused in whole or in part. The design process needs to be based on a thorough understanding of the problem to be solved.

- the goals of program implementation should parallel those of program design. Classes that fill common needs should be built so that they can be reused easily in other programs. Object-oriented design is an important part of program implementation.
- the analysis of programs includes examining and testing programs to determine whether they correctly meet their specifications. It also includes the analysis of programs or algorithms in order to understand their time and space requirements when applied to different data sets.
- data structures are used to represent information within a program. Abstraction is an important theme in the development and application of data structures.
- standard algorithms serve as examples of good solutions to standard problems. Many are intertwined with standard data structures. These algorithms provide examples for analysis of program efficiency.
- an awareness of the ethical and social implications of computing systems is necessary for the study of computer science.
- some objects or concepts are so frequently represented that programmers can draw upon existing code that has already been tested, enabling them to write solutions more quickly and with a greater degree of confidence. (MODULARITY-1)
- programmers use code to represent a physical object or nonphysical concept, real or imagined, by defining a class based on the attributes and/or behaviors of the object or concept. (MODULARITY-2)
- when multiple classes contain common attributes and behaviors, programmers create a new class containing the shared attributes and behaviors forming a hierarchy. Modifications made at the highest level of the hierarchy apply to the subclasses. (MODULARITY-3)
- to find specific solutions to generalizable problems, programmers include variables in their code so that the same algorithm runs using different input values. (VARIABLES-1)
- to manage large amounts of data or complex relationships in data, programmers write code that groups the data together into a single data structure without creating individual variables for each value. (VARIABLES-2)
- the way variables and operators are sequenced and combined in an expression determines the computed result. (CONTROL-1)
- programmers incorporate iteration and selection into code as a way of providing instructions for the computer to process each of the many possible input values. (CONTROL-2)
- while programs are typically designed to achieve a specific purpose, they may have unintended consequences. (IMPACT OF COMPUTING-1)

COURSE ESSENTIAL QUESTIONS

- What is hardware? What is software?
- What happens to information when it is stored digitally?
- What is a memory address?
- What are Java identifiers?
- What is meant by the syntax and semantics of a programming language?
- What are the primary concepts that support object-oriented programming?
- Why is an object an example of abstraction?

- What is primitive data, and how are primitive data types different from objects?
- How can we represent a primitive value as an object?
- What is an algorithm? What is pseudocode?
- What is meant by the flow of control through a program?
- What type of conditions are conditionals and loops based on?
- How are strings compared for equality?
- What is an assignment operator?
- What is the difference between an object and a class?
- What is the scope of a variable?
- What is the difference between public/private methods/variables?
- What are constructors used for? How are they defined?
- What is a null reference?
- What does the `this` reference refer to?
- What is an alias?
- How are objects passed as parameters?
- What are the differences between classes and interfaces?
- What are the differences between arrays, 2D arrays, and `ArrayList`s?
- How are sequential and binary searches different?
- How are selection and insertion sorts different?
- When should an `ArrayList` be used instead of an array?
- How does inheritance support software reuse?
- How does a child class override methods of its parent class?
- What is an abstract class, and when should abstract classes be used?
- How does inheritance support polymorphism?
- What is recursion, and why is it needed?
- What are the differences between merge sort and quick sort?

COURSE KNOWLEDGE & SKILLS

Students will know . . .

- numbers as conveyers of meaningful information, when placed in a context.
- the difference between primitive data and objects.
- the declaration and use of variables.
- mathematical computations in Java.
- the basic program development steps of a Java program.
- the definitions of the flow of control through a Java program.
- the definitions of Java expressions that let them make complex decisions.
- `while` and `for` statements.
- the definition of classes so that they act like the blueprints for new objects, made of variables and methods.
- methods that can be invoked while passing parameters.
- the divisions of complicated methods into simpler supporting methods.
- the descriptions of relationships between objects.
- the creation and use of reference aliases.
- the `static` modifier and its uses.
- the definition and use of formal interfaces and their class implementations.

- the definitions and uses of arrays, 2D arrays, and `ArrayList`s.
- the passing of arrays and array elements as parameters.
- the combination of arrays and other objects to manage complex information.
- the searching-through and sorting of elements of arrays.
- the derivation of new classes from existing ones.
- the design of class hierarchies.
- the proper use of polymorphism.
- the underlying ideas of recursion.
- when recursion should and should not be used.
- recursion's uses to solve problems and in sorting.

Students will be able to . . .

- create executable programs in the Java programming language.
- design, implement, and analyze solutions to problems.
- use and implement commonly used algorithms.
- use standard data structures.
- develop and select appropriate algorithms and data structures to solve new problems.
- write solutions fluently in an object-oriented paradigm.
- write, run, test, and debug solutions in the Java programming language, utilizing standard Java library classes and interfaces from the AP Java subset.
- read and understand programs consisting of several classes and interacting objects.
- read and understand descriptions of the design and development process leading to such a program.
- understand the ethical and social implications of computer use.

COURSE SYLLABUS

Course Name

Advanced Placement Computer Science A

Level

Advanced Placement

Prerequisites

Successful completion of ACP Algebra II, or concurrent enrollment in ACP Algebra II with Department Chair permission.

Materials Required

Laptop or desktop with a Java IDE installed (preferably BlueJ), and Internet access

General Description of the Course

AP Computer Science A (CSA) uses the Java programming language to introduce students to computer science. Through the creation of Java programs, students will cover fundamental topics including the design of solutions to problems, the use of data structures to organize large sets of data, the development and implementation of algorithms to process data and discover new information, the analysis of potential solutions, and the ethical and social implications of computing systems. The course emphasizes object-oriented programming and iterative problem solving and design. If students plan on taking both AP Computer Science Principles and AP Computer Science A, it is suggested that AP Computer Science Principles be taken first.

Assured Assessments

Formative Assessments:

Formative assessments can include, but are not limited to:

- In-class assignments/activities based on the topics in the unit (Units 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 12)
- Textbook review questions (Units 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 12)
- In-class assignments/activities based on prior, released AP CSA College Board Examinations (Unit 11)

Summative Assessments:

- Common end-of-unit assessment scored via common scoring guide (Units 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 12)
- Programming projects based on topics in the unit (Units 2, 3, 4, 7, 8, 9, 10, 12)
- Full released AP CSA College Board Examination (Unit 11)

Core Resources

- *CodeHS*. https://codehs.com/. Web.
- *CodingBat*. https://codingbat.com/java. Web.
- College Board. *AP Computer Science Elevens Lab Student Guide*. https://secure-media.collegeboard.org/digitalServices/pdf/ap/elevens-lab-student-guide.pdf. Web.
- College Board. *AP Computer Science A Magpie Chatbot Lab Student Guide*. https://secure-media.collegeboard.org/digitalServices/pdf/ap/magpie-lab-student-guide.pdf. Web.

- College Board. *AP Computer Science Picture Lab Student Guide*. https://secure-media.collegeboard.org/digitalServices/pdf/ap/picture-lab-studentguide.pdf. Web.
- *CS Awesome*. https://runestone.academy/runestone/books/published/csawesome/index.html. Web.
- Lewis, John, William Loftus, and Cara Cocking. *Java Software Solutions for AP Computer Science*. 3rd ed. Upper Saddle River, NJ: Prentice Hall, 2010. Print.

UNIT 1
Computer Systems

Before beginning Unit 1, students will be introduced to College Board's requirements for this course, and the resources that are available to them. After that, Unit 1 includes a brief overview of computer systems, programming, and programming languages, laying the foundation for the study of software development.

Unit Goals

At the completion of this unit, students will:

The following Unit Goals align with the 2019 College Board Curriculum Framework for Advanced Placement Computer Science A.

MOD-1.A	Call `System` class methods to generate output to the console.
VAR-1.A	Create string literals.

The following Unit Goals align with the 2017 Computer Science Teachers' Association (CSTA) Computer Science Standards.

3A-CS-01	Explain how abstractions hide the underlying implementation details of computing systems embedded in everyday objects.
3A-CS-02	Compare levels of abstraction and interactions between application software, system software, and hardware layers.
3A-DA-09	Translate between different bit representations of real-world phenomena, such as characters, numbers, and images.
3A-DA-10	Evaluate the tradeoffs in how data elements are organized and where data is stored.
3B-CS-01	Categorize the roles of operating system software.
3B-CS-02	Illustrate ways computing systems implement logic, input, and output through hardware components.

The following Unit Goals align with the 2016 International Society for Technology in Education (ISTE) Standards.

ISTE Empowered Learner (Standard 1)	Students leverage technology to take an active role in choosing, achieving and demonstrating competency in their learning goals, informed by the learning sciences.
ISTE Knowledge Constructor (Standard 3)	Students critically curate a variety of resources using digital tools to construct knowledge, produce creative artifacts and

make meaningful learning experiences for themselves and others.

Unit Essential Questions

- What are the Advanced Placement requirements of the College Board?
- What is the relationship between hardware and software?
- How do hardware components execute programs and manage data?
- How are numbers translated to and from the binary, decimal, and hexadecimal number systems?
- What is the Java programming language?
- What are the steps involved in program compilation and execution?

Scope and Sequence

- Overview of Advanced Placement Computer Science A curriculum and requirements
- Identifying software and hardware components
- Number conversion between binary, decimal, and hexadecimal number systems
- Introduction to the Java programming language
- Creation and execution of first Java program

Assured Assessments

Formative Assessments:
- Students will complete several in-class assignments/activities based on the topics in the unit.
- Students will create their first executable program: “printing” text on a terminal.
- Students will complete review questions (multiple-choice, true/false) from the textbook.

Summative Assessment:
- Students will take a brief, common assessment based on number conversions, scored via a common scoring guide.

Resources

Core
- Lewis, John, William Loftus, and Cara Cocking. *Java Software Solutions for AP Computer Science*. 3rd ed. Upper Saddle River, NJ: Prentice Hall, 2010. Print. Chp. 1: 1.0, 1.1, 1.3, 1.4.
- Laptop or desktop with a Java IDE installed (preferably BlueJ), and Internet access

Time Allotment

- Approximately 7 school days

UNIT 2
Objects and Primitive Data

Exploring the key elements used in a program – objects and primitive data – in Unit 2 students learn to create and use objects to work with character strings, get information from the user, do complex calculations, and format output.

Unit Goals

At the completion of this unit, students will:

The following Unit Goals align with the 2019 College Board Curriculum Framework for Advanced Placement Computer Science A.

MOD-1.A	Call `System` class methods to generate output to the console.
MOD-1.B	Explain the relationship between a class and an object.
MOD-1.C	Identify, using its signature, the correct constructor being called.
MOD-1.D	For creating objects: a. Create objects by calling constructors without parameters. b. Create objects by calling constructors with parameters.
MOD-1.H	Call static methods.
VAR-1.A	Create string literals.
VAR-1.B	Identify the most appropriate data type category for a particular specification.
VAR-1.C	Declare variables of the correct types to represent primitive data.
VAR-1.E	For `String` class: a. Create `String` objects. b. Call `String` methods.
VAR-1.F	For wrapper classes: a. Create `Integer` objects. b. Call `Integer` methods. c. Create `Double` objects. d. Call `Double` methods.
CON-1.A	Evaluate arithmetic expressions in a program code.
CON-1.B	Evaluate what is stored in a variable as a result of an expression with an assignment statement.
CON-1.C	Evaluate arithmetic expressions that use casting.

CON-1.D	Evaluate expressions that use the `Math` class methods.

The following Unit Goals align with the 2017 Computer Science Teachers' Association (CSTA) Computer Science Standards.

3A-DA-09	Translate between different bit representations of real-world phenomena, such as characters, numbers, and images.
3A-DA-10	Evaluate the tradeoffs in how data elements are organized and where data is stored.
3A-AP-13	Create prototypes that use algorithms to solve computational problems by leveraging prior student knowledge and personal interests.
3A-AP-16	Design and iteratively develop computational artifacts for practical intent, personal expression, or to address a societal issue by using events to initiate instructions.
3A-AP-17	Decompose problems into smaller components through systematic analysis, using constructs such as procedures, modules, and/or objects.
3A-AP-18	Create artifacts by using procedures within a program, combinations of data and procedures, or independent but interrelated programs.
3A-AP-21	Evaluate and refine computational artifacts to make them more usable and accessible.
3A-IC-26	Demonstrate ways a given algorithm applies to problems across disciplines.
3B-DA-07	Evaluate the ability of models and simulations to test and support the refinement of hypotheses.
3B-AP-10	Use and adapt classic algorithms to solve computational problems.
3B-AP-12	Compare and contrast fundamental data structures and their uses.
3B-AP-14	Construct solutions to problems using student-created components, such as procedures, modules and/or objects.
3B-AP-15	Analyze a large-scale computational problem and identify generalizable patterns that can be applied to a solution.
3B-AP-16	Demonstrate code reuse by creating programming solutions using libraries and APIs.
3B-AP-21	Develop and use a series of test cases to verify that a program performs according to its design specifications.

3B-AP-22	Modify an existing program to add additional functionality and discuss intended and unintended implications (e.g., breaking other functionality).
3B-AP-23	Evaluate key qualities of a program through a process such as a code review.

The following Unit Goals align with the 2016 International Society for Technology in Education (ISTE) Standards.

ISTE Empowered Learner (Standard 1)	Students leverage technology to take an active role in choosing, achieving and demonstrating competency in their learning goals, informed by the learning sciences.
ISTE Knowledge Constructor (Standard 3)	Students critically curate a variety of resources using digital tools to construct knowledge, produce creative artifacts and make meaningful learning experiences for themselves and others.
ISTE Innovative Designer (Standard 4)	Students use a variety of technologies within a design process to identify and solve problems by creating new, useful or imaginative solutions.
ISTE Computational Thinker (Standard 5)	Students develop and employ strategies for understanding and solving problems in ways that leverage the power of technological methods to develop and test solutions.

Unit Essential Questions

- What are the different primitive data types?
- How are variables declared and used?
- How are mathematical computations performed in Java?
- What are objects?
- How are objects created and used?
- What are the differences between primitive data and objects?

Scope and Sequence

- Student completion of a project used as an example for the rest of the course: creation of a face on the terminal by creating shape objects and manipulating them, learning how to create and use objects before formally understanding what each line of their program is doing
- Exploration of the different ways to print/display text on the terminal, including using the two different print commands, as well as escape sequences. Strings are also introduced, while stressing the importance of understanding that they are objects, not primitive types.
- Introduction of four primitive data types: `int`, `double`, `Boolean`, and `char`.

- Learning of the particular way that Java completes mathematical computations, along with variables, primitive data types, and how to create random numbers using the *Math.random()* method.
- The basics about objects, revisiting the face project from the beginning of the unit
- Learning how to make programs that interact with the user, through the use of a Scanner, a vital skill for the rest of the course, as the user can now input information into programs

Assured Assessments

Formative Assessments:

- Students will complete several in-class assignments/activities based on the topics in the unit, including programming tasks on *CodingBat*.
- Students will complete review questions (multiple-choice, true/false) from the textbook.

Summative Assessments:

- Students will complete programming projects covering object creation and manipulation, formatting text, primitive data creation and manipulation, random numbers, and user input.
- Students will take a common end-of-unit assessment scored via a common scoring guide.

Resources

Core

- Lewis, John, William Loftus, and Cara Cocking. *Java Software Solutions for AP Computer Science*. 3rd ed. Upper Saddle River, NJ: Prentice Hall, 2010. Print. Chp. 2: 2.0, 2.1, 2.2, 2.3, 2.3, 2.5, 2.7, 2.8, 2.9.
- Laptop or desktop with a Java IDE installed (preferably BlueJ), and Internet access
- *CodingBat*. https://codingbat.com/java. Web.

Supplemental

- *CodeHS*. https://codehs.com/. Web.
- *CS Awesome*. https://runestone.academy/runestone/books/published/csawesome/index.html. Web.

Time Allotment

- Approximately 13 school days

UNIT 3
Program Statements – Conditional

Unit 3 examines several programming statements, as well as some additional operators, that allow one to perform basic operations, including interaction with objects and the definition of the services those objects provides; these are the first step toward a disciplined software development process.

Unit Goals

At the completion of this unit, students will:

The following Unit Goals align with the 2019 College Board Curriculum Framework for Advanced Placement Computer Science A.

CON-1.E	Evaluate Boolean expressions that use relational operators in program code.
CON-1.F	Evaluate compound Boolean expressions in program code.
CON-1.G	Compare and contrast equivalent Boolean expressions.
CON-2.A	Represent branching logical processes by using conditional statements.
CON-2.B	Represent branching logical processes by using nested conditional statements.

The following Unit Goals align with the 2017 Computer Science Teachers' Association (CSTA) Computer Science Standards.

3A-DA-10	Evaluate the tradeoffs in how data elements are organized and where data is stored.
3A-AP-13	Create prototypes that use algorithms to solve computational problems by leveraging prior student knowledge and personal interests.
3A-AP-15	Justify the selection of specific control structures when tradeoffs involve implementation, readability, and program performance, and explain the benefits and drawbacks of choices made.
3A-AP-16	Design and iteratively develop computational artifacts for practical intent, personal expression, or to address a societal issue by using events to initiate instructions.
3A-AP-21	Evaluate and refine computational artifacts to make them more usable and accessible.
3B-AP-10	Use and adapt classic algorithms to solve computational problems.
3B-AP-11	Evaluate algorithms in terms of their efficiency, correctness, and clarity.

3B-AP-21	Develop and use a series of test cases to verify that a program performs according to its design specifications.
3B-AP-22	Modify an existing program to add additional functionality and discuss intended and unintended implications (e.g., breaking other functionality).
3B-AP-23	Evaluate key qualities of a program through a process such as a code review.

The following Unit Goals align with the 2016 International Society for Technology in Education (ISTE) Standards.

ISTE Empowered Learner (Standard 1)	Students leverage technology to take an active role in choosing, achieving and demonstrating competency in their learning goals, informed by the learning sciences.
ISTE Knowledge Constructor (Standard 3)	Students critically curate a variety of resources using digital tools to construct knowledge, produce creative artifacts and make meaningful learning experiences for themselves and others.
ISTE Innovative Designer (Standard 4)	Students use a variety of technologies within a design process to identify and solve problems by creating new, useful or imaginative solutions.
ISTE Computational Thinker (Standard 5)	Students develop and employ strategies for understanding and solving problems in ways that leverage the power of technological methods to develop and test solutions.

Unit Essential Questions

- What are the basic steps of program development?
- How can programmers control the flow of a program?
- What are conditional statements?
- How can conditional statements be used to make complex decisions within a program?

Scope and Sequence

- The four basic programming steps in developing software: establishing the requirements, creating a design, implementing the code, and testing the implementation
- Basic control structures: loops and conditional statements. These are used to create more complex programs which make decisions based on pre-determined information, or user-entered data. The *if* statement and *if-else* statements are the main focus; *nested if* statements (also refactored as *else-if* statements) are also examined.
- Boolean expressions used to further control the flow of programs, with the three basic logical operators: AND, OR, and NOT. Truth tables are completed in order to understand how these operators work.

- Programming “shortcuts”: increment, decrement, and assignment operators as shorter ways to program mathematical expressions already studied

Assured Assessments

Formative Assessments:

- Students will complete several in-class assignments/activities based on the topics in the unit, including programming tasks on CodingBat.
- Students will complete review questions (multiple-choice, true/false) from the textbook.

Summative Assessments:

- Students will complete programming projects covering conditional statements and simple control structures (forced `while` loop).
- Students will take a common end-of-unit assessment scored via a common scoring guide.

Resources

Core

- Lewis, John, William Loftus, and Cara Cocking. *Java Software Solutions for AP Computer Science*. 3rd ed. Upper Saddle River, NJ: Prentice Hall, 2010. Print. Chp. 3: 3.0, 3.1, 3.2, 3.3, 3.4.
- Laptop or desktop with a Java IDE installed (preferably BlueJ), and Internet access
- *CodingBat*. https://codingbat.com/java. Web.

Supplemental

- *CodeHS*. https://codehs.com/. Web.
- *CS Awesome*. https://runestone.academy/runestone/books/published/csawesome/index.html. Web.

Time Allotment

- Approximately 15 school days

UNIT 4
Program Statements – Iteration

Unit 4 builds on Boolean operators by focusing on iteration using `while` and `for` loops, including several standard algorithms that use iteration; this forms the basis for future concepts such as searching, sorting, arrays, `ArrayList`s, and 2D arrays.

Unit Goals

At the completion of this unit, students will:

The following Unit Goals align with the 2019 College Board Curriculum Framework for Advanced Placement Computer Science A.

CON-1.F	Evaluate compound Boolean expressions in program code.
CON-1.G	Compare and contrast equivalent Boolean expressions.
CON-2.C	Represent iterative processes using a `while` loop.
CON-2.D	For algorithms in the context of a particular specification that does not require the use of traversals: a. Identify standard algorithms. b. Modify standard algorithms. c. Develop an algorithm.
CON-2.E	Represent iterative processes using a `for` loop.
CON-2.F	For algorithms in the context of a particular specification that involves `String` objects: a. Identify standard algorithms. b. Modify standard algorithms. c. Develop an algorithm.
CON-2.G	Represent nested iterative processes.
CON-2.H	Compute statement execution counts and informal run-time comparison of iterative statements.
IOC-1.A	Explain the ethical and social implications of computing systems.

The following Unit Goals align with the 2017 Computer Science Teachers' Association (CSTA) Computer Science Standards.

3A-DA-12	Create computational models that represent the relationships among different elements of data collected from a phenomenon or process.
3A-AP-13	Create prototypes that use algorithms to solve computational problems by

	leveraging prior student knowledge and personal interests.
3A-AP-15	Justify the selection of specific control structures when tradeoffs involve implementation, readability, and program performance, and explain the benefits and drawbacks of choices made.
3A-AP-16	Design and iteratively develop computational artifacts for practical intent, personal expression, or to address a societal issue by using events to initiate instructions.
3A-AP-17	Decompose problems into smaller components through systematic analysis, using constructs such as procedures, modules, and/or objects.
3A-AP-18	Create artifacts by using procedures within a program, combinations of data and procedures, or independent but interrelated programs.
3A-AP-19	Systematically design and develop programs for broad audiences by incorporating feedback from users.
3A-IC-24	Evaluate the ways computing impacts personal, ethical, social, economic, and cultural practices.
3A-IC-25	Test and refine computational artifacts to reduce bias and equity deficits.
3B-DA-07	Evaluate the ability of models and simulations to test and support the refinement of hypotheses.
3B-AP-08	Describe how artificial intelligence drives many software and physical systems.
3B-AP-09	Implement an artificial intelligence algorithm to play a game against a human opponent or solve a problem.
3B-AP-10	Use and adapt classic algorithms to solve computational problems.
3B-AP-11	Evaluate algorithms in terms of their efficiency, correctness, and clarity.
3B-AP-12	Compare and contrast fundamental data structures and their uses.
3B-AP-14	Construct solutions to problems using student-created components, such as procedures, modules and/or objects.
3B-AP-15	Analyze a large-scale computational problem and identify generalizable patterns that can be applied to a solution.
3B-AP-16	Demonstrate code reuse by creating programming solutions using libraries and APIs.
3B-AP-21	Develop and use a series of test cases to verify that a program performs according to its design specifications.

3B-AP-22	Modify an existing program to add additional functionality and discuss intended and unintended implications (e.g., breaking other functionality).
3B-AP-23	Evaluate key qualities of a program through a process such as a code review.
3B-IC-27	Predict how computational innovations that have revolutionized aspects of our culture might evolve.

The following Unit Goals align with the 2016 International Society for Technology in Education (ISTE) Standards.

ISTE Empowered Learner (Standard 1)	Students leverage technology to take an active role in choosing, achieving and demonstrating competency in their learning goals, informed by the learning sciences.
ISTE Digital Citizen (Standard 2)	Students recognize the rights, responsibilities and opportunities of living, learning and working in an interconnected digital world, and they act and model in ways that are safe, legal and ethical.
ISTE Knowledge Constructor (Standard 3)	Students critically curate a variety of resources using digital tools to construct knowledge, produce creative artifacts and make meaningful learning experiences for themselves and others.
ISTE Innovative Designer (Standard 4)	Students use a variety of technologies within a design process to identify and solve problems by creating new, useful or imaginative solutions.
ISTE Computational Thinker (Standard 5)	Students develop and employ strategies for understanding and solving problems in ways that leverage the power of technological methods to develop and test solutions.
ISTE Creative Communicator (Standard 6)	Students communicate clearly and express themselves creatively for a variety of purposes using the platforms, tools, styles, formats and digital media appropriate to their goals.

Unit Essential Questions

- Besides the programming elements contained within them, what are the three basic components of any loop?
- What are the similarities and differences between `while` and `for` loops?
- When should `while` loops be used instead of `for` loops?
- When should `for` loops be used instead of `while` loops?

Scope and Sequence

- Completion of Activities 1-4 of College Board's *Magpie Chatbot Lab*, creating a basic chatbot program that interacts with the user; based on the user-entered question/answer, a response is selected from a set of predetermined responses within the program and displayed to the user, and the program continues until the user enters a specific command to stop.
- The details of `while` and `for` loops, including the basic structure of each, and the situations that dictate using one over another; extensive practice with loops, including nested loops, occurs, as well as techniques to avoid creating infinite loops.

Assured Assessments

Formative Assessments:

- Students will complete several in-class assignments/activities based on the topics in the unit.
- Students will complete review questions (multiple-choice, true/false) from the textbook.

Summative Assessments:

- Students will complete programming projects covering `while` loops, `for` loops, and nested loops.
- Students will take a common end-of-unit assessment scored via a common scoring guide.

Resources

Core

- Lewis, John, William Loftus, and Cara Cocking. *Java Software Solutions for AP Computer Science*. 3rd ed. Upper Saddle River, NJ: Prentice Hall, 2010. Print. Chp. 3: 3.5, 3.7.
- Laptop or desktop with a Java IDE installed (preferably BlueJ), and Internet access
- College Board. *AP Computer Science A Magpie Chatbot Lab Student Guide*. https://secure-media.collegeboard.org/digitalServices/pdf/ap/magpie-lab-student-guide.pdf. Web.

Supplemental

- *CodeHS*. https://codehs.com/. Web.
- *CodingBat*. https://codingbat.com/java. Web.
- *CS Awesome*. https://runestone.academy/runestone/books/published/csawesome/index.html. Web.

Time Allotment

- Approximately 15 school days

UNIT 5
Writing Classes

With the prior foundation of objects, classes, and several fundamental programing statements, students in Unit 5 begin designing more complex software by creating their own classes to define objects that perform whatever services we define; the unit explores the details of class definitions, including the structure and semantics of methods and the scope and encapsulation of data.

Unit Goals

At the completion of this unit, students will:

The following Unit Goals align with the 2019 College Board Curriculum Framework for Advanced Placement Computer Science A.

CON-1.H	Compare object references using Boolean expressions in program code.
MOD-2.A	Designate access and visibility constraints to classes, data, constructors, and methods.
MOD-2.B	Define instance variables for the attributes to be initialized through the constructors of a class.
MOD-2.C	Describe the functionality and use of program code through comments.
MOD-2.D	Define behaviors of an object through non-void methods without parameters written in a class.
MOD-2.E	Define behaviors of an object through void methods with or without parameters written in a class.
MOD-2.F	Define behaviors of an object through non-void methods with parameters written in a class.
MOD-3.A	Designate private visibility of instance variables to encapsulate the attributes of an object.

The following Unit Goals align with the 2017 Computer Science Teachers' Association (CSTA) Computer Science Standards.

3A-DA-12	Create computational models that represent the relationships among different elements of data collected from a phenomenon or process.
3A-AP-13	Create prototypes that use algorithms to solve computational problems by leveraging prior student knowledge and personal interests.
3A-AP-15	Justify the selection of specific control structures when tradeoffs involve implementation, readability, and program performance, and explain the benefits

	and drawbacks of choices made.
3A-AP-16	Design and iteratively develop computational artifacts for practical intent, personal expression, or to address a societal issue by using events to initiate instructions.
3A-AP-17	Decompose problems into smaller components through systematic analysis, using constructs such as procedures, modules, and/or objects.
3A-AP-18	Create artifacts by using procedures within a program, combinations of data and procedures, or independent but interrelated programs.
3A-AP-19	Systematically design and develop programs for broad audiences by incorporating feedback from users.
3A-IC-24	Evaluate the ways computing impacts personal, ethical, social, economic, and cultural practices.
3B-DA-07	Evaluate the ability of models and simulations to test and support the refinement of hypotheses.
3B-AP-08	Describe how artificial intelligence drives many software and physical systems.
3B-AP-09	Implement an artificial intelligence algorithm to play a game against a human opponent or solve a problem.
3B-AP-10	Use and adapt classic algorithms to solve computational problems.
3B-AP-11	Evaluate algorithms in terms of their efficiency, correctness, and clarity.
3B-AP-12	Compare and contrast fundamental data structures and their uses.
3B-AP-14	Construct solutions to problems using student-created components, such as procedures, modules and/or objects.
3B-AP-15	Analyze a large-scale computational problem and identify generalizable patterns that can be applied to a solution.
3B-AP-16	Demonstrate code reuse by creating programming solutions using libraries and APIs.
3B-AP-21	Develop and use a series of test cases to verify that a program performs according to its design specifications.
3B-AP-22	Modify an existing program to add additional functionality and discuss intended and unintended implications (e.g., breaking other functionality).
3B-AP-23	Evaluate key qualities of a program through a process such as a code review.

The following Unit Goals align with the 2016 International Society for Technology in Education (ISTE) Standards.

ISTE Empowered Learner (Standard 1)	Students leverage technology to take an active role in choosing, achieving and demonstrating competency in their learning goals, informed by the learning sciences.
ISTE Knowledge Constructor (Standard 3)	Students critically curate a variety of resources using digital tools to construct knowledge, produce creative artifacts and make meaningful learning experiences for themselves and others.
ISTE Innovative Designer (Standard 4)	Students use a variety of technologies within a design process to identify and solve problems by creating new, useful or imaginative solutions.
ISTE Computational Thinker (Standard 5)	Students develop and employ strategies for understanding and solving problems in ways that leverage the power of technological methods to develop and test solutions.

Unit Essential Questions

- What are classes, and why are they so important in Java?
- What are methods, and how are they declared?
- How are methods invoked?
- What are parameters, and how are they passed to, and handled by, methods?
- What is the main difference between accessor methods and mutator methods?
- What is method overloading?

Scope and Sequence

- Creation in Java of classes, which act as blueprints for new objects, made of variables and methods; students learn how to declare variables, how to create objects using constructors, and how methods are used to manipulate objects once they are created.
- The specifics of methods: basic structure, how they are invoked, and how data can be sent to them and sent from them; specific attention is paid to accessor and mutator methods, and how parameters are passed. Understanding the flow of data to and from methods is vital.
- Completion of Activity 1 of College Board's *Elevens Lab*, creating a Card class that will ultimately be used, later in the course, in a program that plays the card game "Elevens" with the user

Assured Assessments

Formative Assessments:

- Students will complete several in-class assignments/activities based on the topics in the unit, as well as explorations in *CSAwesome* and *CodeHS*.

- Students will complete review questions (multiple-choice, true/false) from the textbook.

Summative Assessment:

- Students will take a common end-of-unit assessment scored via a common scoring guide.

Resources

Core

- Lewis, John, William Loftus, and Cara Cocking. *Java Software Solutions for AP Computer Science*. 3rd ed. Upper Saddle River, NJ: Prentice Hall, 2010. Print. Chp. 4: 4.0, 4.1, 4.2, 4.3, 4.4, 4.5.
- Laptop or desktop with a Java IDE installed (preferably BlueJ), and Internet access
- *CodeHS*. https://codehs.com/. Web.
- *CS Awesome*. https://runestone.academy/runestone/books/published/csawesome/index.html. Web.
- College Board. *AP Computer Science Elevens Lab Student Guide*. https://secure-media.collegeboard.org/digitalServices/pdf/ap/elevens-lab-student-guide.pdf. Web.

Supplemental

- *CodingBat*. https://codingbat.com/java. Web.

Time Allotment

- Approximately 15 school days

UNIT 6
Enhancing Classes

Unit 6 explores a variety of issues related to the design and implementation of classes, including the concept of an object reference, the `static` modifier, a wrapper class to help process input, the ability to nest one class definition within another, and finally the use of an interface construct to formalize the interaction between classes.

Unit Goals

At the completion of this unit, students will:

The following Unit Goals align with the 2019 College Board Curriculum Framework for Advanced Placement Computer Science A.

CON-1.H	Compare object references using Boolean expressions in program code.
MOD-2.G	Define behaviors of a class through static methods.
MOD-2.H	Define the static variables that belong to the class.
VAR-1.G	Explain where variables can be used in the program code.
VAR-1.H	Evaluate object reference expressions that use the keyword `this`.

The following Unit Goals align with the 2017 Computer Science Teachers' Association (CSTA) Computer Science Standards.

3A-DA-12	Create computational models that represent the relationships among different elements of data collected from a phenomenon or process.
3A-AP-13	Create prototypes that use algorithms to solve computational problems by leveraging prior student knowledge and personal interests.
3A-AP-15	Justify the selection of specific control structures when tradeoffs involve implementation, readability, and program performance, and explain the benefits and drawbacks of choices made.
3A-AP-16	Design and iteratively develop computational artifacts for practical intent, personal expression, or to address a societal issue by using events to initiate instructions.
3A-AP-17	Decompose problems into smaller components through systematic analysis, using constructs such as procedures, modules, and/or objects.
3A-AP-18	Create artifacts by using procedures within a program, combinations of data and procedures, or independent but interrelated programs.
3A-AP-19	Systematically design and develop programs for broad audiences by incorporating

	feedback from users.
3A-IC-24	Evaluate the ways computing impacts personal, ethical, social, economic, and cultural practices.
3B-DA-07	Evaluate the ability of models and simulations to test and support the refinement of hypotheses.
3B-AP-08	Describe how artificial intelligence drives many software and physical systems.
3B-AP-09	Implement an artificial intelligence algorithm to play a game against a human opponent or solve a problem.
3B-AP-10	Use and adapt classic algorithms to solve computational problems.
3B-AP-11	Evaluate algorithms in terms of their efficiency, correctness, and clarity.
3B-AP-12	Compare and contrast fundamental data structures and their uses.
3B-AP-14	Construct solutions to problems using student-created components, such as procedures, modules and/or objects.
3B-AP-15	Analyze a large-scale computational problem and identify generalizable patterns that can be applied to a solution.
3B-AP-16	Demonstrate code reuse by creating programming solutions using libraries and APIs.
3B-AP-21	Develop and use a series of test cases to verify that a program performs according to its design specifications.
3B-AP-22	Modify an existing program to add additional functionality and discuss intended and unintended implications (e.g., breaking other functionality).
3B-AP-23	Evaluate key qualities of a program through a process such as a code review.

The following Unit Goals align with the 2016 International Society for Technology in Education (ISTE) Standards.

ISTE Empowered Learner (Standard 1)	Students leverage technology to take an active role in choosing, achieving and demonstrating competency in their learning goals, informed by the learning sciences.
ISTE Knowledge Constructor (Standard 3)	Students critically curate a variety of resources using digital tools to construct knowledge, produce creative artifacts and make meaningful learning experiences for themselves and others.
ISTE Innovative Designer	Students use a variety of technologies within a design

(Standard 4)	process to identify and solve problems by creating new, useful or imaginative solutions.
ISTE Computational Thinker (Standard 5)	Students develop and employ strategies for understanding and solving problems in ways that leverage the power of technological methods to develop and test solutions.

Unit Essential Questions

- What are object reference aliases, and how are they created?
- How are object references passed as parameters?
- What is the `this` reference?
- What is the `static` modifier, and why is it needed in Java?
- What is an exception, and what are the basic types of exceptions that Java displays?
- What are interfaces, and how do they help interact with classes?

Scope and Sequence

- Further exploration of object references, including the `null` reference and the `this` reference; aliases are also introduced. Students are led to understand the difference between making "copies" of an object and manipulating the copy, or getting access to the original object and manipulating it from a different method. The difference between using the = = operator and the `equals` method on objects is also covered.
- Introduction of the `static` modifier, which allows Java to associate a variable or method with its class rather than with an object; sample programs are explored, where `static` variables and/or `static` methods are used.
- Exceptions as a type of program error
- How interfaces can be used in Java, including abstract methods, which also leads to abstract classes; supplemental materials help students understand the differences between abstract classes and interfaces.

Assured Assessments

Formative Assessments:

- Students will complete several in-class assignments/activities based on the topics in the unit, as well as explorations in *CSAwesome* and *CodeHS*.
- Students will complete review questions (multiple-choice, true/false) from the textbook.

Summative Assessment:

- Students will take a common end-of-unit assessment scored via a common scoring guide.

Resources

Core

- Lewis, John, William Loftus, and Cara Cocking. *Java Software Solutions for AP Computer Science*. 3rd ed. Upper Saddle River, NJ: Prentice Hall, 2010. Print. Chp. 5: 5.0, 5.1, 5.2, 5.3.
- Laptop or desktop with a Java IDE installed (preferably BlueJ), and Internet access
- *CodeHS*. https://codehs.com/. Web.
- *CS Awesome*. https://runestone.academy/runestone/books/published/csawesome/index.html. Web.

Supplemental

- *CodingBat*. https://codingbat.com/java. Web.

Time Allotment

- Approximately 15 school days

UNIT 7
Arrays, 2D Arrays, and Searching

Unit 7 focuses on data structures, which are used to represent collections of related data using a single variable rather than multiple variables. Arrays – programming constructs that group data into lists – and 2D arrays – most suitable to represent a table – use nested iterative statements to traverse and access all elements.

Unit Goals

At the completion of this unit, students will:

The following Unit Goals align with the 2019 College Board Curriculum Framework for Advanced Placement Computer Science A.

VAR-2.A	Represent collections of related primitive or object reference data using one-dimensional (1D) array objects.
VAR-2.B	Traverse the elements in a 1D array.
VAR-2.C	Traverse the elements in a 1D array object using an enhanced `for` loop.
VAR-2.F	Represent collections of related primitive or object reference data using two-dimensional (2D) array objects.
VAR-2.G	For 2D array objects: a. Traverse using nested for loops. b. Traverse using nested enhanced `for` loops.
CON-2.I	For algorithms in the context of a particular specification that requires the use of array traversals: a. Identify standard algorithms. b. Modify standard algorithms. c. Develop an algorithm.
CON-2.K	Apply sequential/linear search algorithms to search for specific information in array or `ArrayList` objects.
CON-2.N	For algorithms in the context of a particular specification that requires the use of 2D array traversals: a. Identify standard algorithms. b. Modify standard algorithms. c. Develop an algorithm.
IOC-1.A	Explain the ethical and social implications of computing systems.
IOC-1.B	Explain the risks to privacy from collecting and storing personal data on computer systems.

The following Unit Goals align with the 2017 Computer Science Teachers' Association (CSTA) Computer Science Standards.

3A-DA-09	Translate between different bit representations of real-world phenomena, such as characters, numbers, and images.
3A-DA-10	Evaluate the tradeoffs in how data elements are organized and where data is stored.
3A-DA-12	Create computational models that represent the relationships among different elements of data collected from a phenomenon or process.
3A-AP-13	Create prototypes that use algorithms to solve computational problems by leveraging prior student knowledge and personal interests.
3A-AP-14	Use lists to simplify solutions, generalizing computational problems instead of repeatedly using simple variables.
3A-AP-15	Justify the selection of specific control structures when tradeoffs involve implementation, readability, and program performance, and explain the benefits and drawbacks of choices made.
3A-AP-16	Design and iteratively develop computational artifacts for practical intent, personal expression, or to address a societal issue by using events to initiate instructions.
3A-AP-17	Decompose problems into smaller components through systematic analysis, using constructs such as procedures, modules, and/or objects.
3A-AP-18	Create artifacts by using procedures within a program, combinations of data and procedures, or independent but interrelated programs.
3A-AP-19	Systematically design and develop programs for broad audiences by incorporating feedback from users.
3A-IC-24	Evaluate the ways computing impacts personal, ethical, social, economic, and cultural practices.
3A-IC-25	Test and refine computational artifacts to reduce bias and equity deficits.
3B-DA-07	Evaluate the ability of models and simulations to test and support the refinement of hypotheses.
3B-AP-08	Describe how artificial intelligence drives many software and physical systems.
3B-AP-10	Use and adapt classic algorithms to solve computational problems.
3B-AP-11	Evaluate algorithms in terms of their efficiency, correctness, and clarity.

3B-AP-12	Compare and contrast fundamental data structures and their uses.
3B-AP-14	Construct solutions to problems using student-created components, such as procedures, modules and/or objects.
3B-AP-15	Analyze a large-scale computational problem and identify generalizable patterns that can be applied to a solution.
3B-AP-16	Demonstrate code reuse by creating programming solutions using libraries and APIs.
3B-AP-21	Develop and use a series of test cases to verify that a program performs according to its design specifications.
3B-AP-22	Modify an existing program to add additional functionality and discuss intended and unintended implications (e.g., breaking other functionality).
3B-AP-23	Evaluate key qualities of a program through a process such as a code review.
3B-IC-27	Predict how computational innovations that have revolutionized aspects of our culture might evolve.

The following Unit Goals align with the 2016 International Society for Technology in Education (ISTE) Standards.

ISTE Empowered Learner (Standard 1)	Students leverage technology to take an active role in choosing, achieving and demonstrating competency in their learning goals, informed by the learning sciences.
ISTE Digital Citizen (Standard 2)	Students recognize the rights, responsibilities and opportunities of living, learning and working in an interconnected digital world, and they act and model in ways that are safe, legal and ethical.
ISTE Knowledge Constructor (Standard 3)	Students critically curate a variety of resources using digital tools to construct knowledge, produce creative artifacts and make meaningful learning experiences for themselves and others.
ISTE Innovative Designer (Standard 4)	Students use a variety of technologies within a design process to identify and solve problems by creating new, useful or imaginative solutions.
ISTE Computational Thinker (Standard 5)	Students develop and employ strategies for understanding and solving problems in ways that leverage the power of technological methods to develop and test solutions.
ISTE Creative Communicator (Standard 6)	Students communicate clearly and express themselves creatively for a variety of purposes using the platforms, tools, styles, formats and digital media appropriate to their goals.

Unit Essential Questions

- What are arrays and 2D arrays, and why do we use them?
- How are elements of arrays and 2D arrays accessed and/or manipulated?
- How are arrays and elements of arrays passed as parameters?
- How can knowing standard algorithms be useful when solving new problems?
- What algorithms are used to search through the elements of an array or 2D array?

Scope and Sequence

- Introduction of arrays: creation of them, storing data in them, and how to access/manipulate them.
- The algorithms for a sequential/linear search and a binary search, with emphasis on using binary search when possible
- 2D arrays (tables, grids, or any data structures that organize elements in rows and columns); the modification of search algorithms to work with 2D arrays is included.
- Completion of Activities 1-9 of College Board's *Picture Lab*, tinting images to a certain color, creating filters to enhance different aspects of images, mirroring images over different lines of symmetry, and using the technique of chromakey (merging pictures using a "green screen")

Assured Assessments

Formative Assessments:
- Students will complete several in-class assignments/activities based on the topics in the unit, including programming tasks on *CodingBat*.
- Students will complete review questions (multiple-choice, true/false) from the textbook.

Summative Assessments:
- Students will complete programming projects covering arrays and 2D arrays.
- Students will complete Activities 1-9 of College Board's *Picture Lab*.
- Students will take a common end-of-unit assessment scored via a common scoring guide.

Resources

Core
- Lewis, John, William Loftus, and Cara Cocking. *Java Software Solutions for AP Computer Science*. 3rd ed. Upper Saddle River, NJ: Prentice Hall, 2010. Print. Chp. 6: 6.0, 6.1, 6.2, 6.6.
- Laptop or desktop with a Java IDE installed (preferably BlueJ), and Internet access
- *CodingBat*. https://codingbat.com/java. Web.
- College Board. *AP Computer Science Picture Lab Student Guide*. https://secure-media.collegeboard.org/digitalServices/pdf/ap/picture-lab-studentguide.pdf. Web.

Supplemental
- *CodeHS*. https://codehs.com/. Web.

- *CS Awesome*. https://runestone.academy/runestone/books/published/csawesome/index.html. Web.

Time Allotment

- Approximately 18 school days

UNIT 8
`List`, `ArrayLists`, Selection and Insert Sorts

As composed to the arrays studied in Unit 7, the `ArrayList` object has a dynamic size, and the class contains methods for insertion and deletion of elements; Unit 8 introduces students to two methods of sorting data: selection and insertion both allow binary searching to locate elements.

Unit Goals

At the completion of this unit, students will:

The following Unit Goals align with the 2019 College Board Curriculum Framework for Advanced Placement Computer Science A.

VAR-2.D Represent collections of related object reference data using `ArrayList` objects.

VAR-2.E For `ArrayList` objects:
a. Traverse using a `for` or `while` loop.
b. Traverse using an enhanced `for` loop.

CON-2.J For algorithms in the context of a particular specification that requires the use of `ArrayList` traversals:
a. Identify standard algorithms.
b. Modify standard algorithms.
c. Develop an algorithm.

CON-2.L Apply selection sort and insertion sort algorithms to sort the elements of array or `ArrayList` objects.

CON-2.M Compute statement execution counts and informal run-time comparison of sorting algorithms.

The following Unit Goals align with the 2017 Computer Science Teachers' Association (CSTA) Computer Science Standards.

3A-DA-09 Translate between different bit representations of real-world phenomena, such as characters, numbers, and images.

3A-DA-10 Evaluate the tradeoffs in how data elements are organized and where data is stored.

3A-DA-12 Create computational models that represent the relationships among different elements of data collected from a phenomenon or process.

3A-AP-13 Create prototypes that use algorithms to solve computational problems by leveraging prior student knowledge and personal interests.

3A-AP-14 Use lists to simplify solutions, generalizing computational problems instead of

	repeatedly using simple variables.
3A-AP-15	Justify the selection of specific control structures when tradeoffs involve implementation, readability, and program performance, and explain the benefits and drawbacks of choices made.
3A-AP-16	Design and iteratively develop computational artifacts for practical intent, personal expression, or to address a societal issue by using events to initiate instructions.
3A-AP-17	Decompose problems into smaller components through systematic analysis, using constructs such as procedures, modules, and/or objects.
3A-AP-18	Create artifacts by using procedures within a program, combinations of data and procedures, or independent but interrelated programs.
3A-AP-19	Systematically design and develop programs for broad audiences by incorporating feedback from users.
3B-DA-07	Evaluate the ability of models and simulations to test and support the refinement of hypotheses.
3B-AP-08	Describe how artificial intelligence drives many software and physical systems.
3B-AP-09	Implement an artificial intelligence algorithm to play a game against a human opponent or solve a problem.
3B-AP-10	Use and adapt classic algorithms to solve computational problems.
3B-AP-11	Evaluate algorithms in terms of their efficiency, correctness, and clarity.
3B-AP-12	Compare and contrast fundamental data structures and their uses.
3B-AP-14	Construct solutions to problems using student-created components, such as procedures, modules and/or objects.
3B-AP-15	Analyze a large-scale computational problem and identify generalizable patterns that can be applied to a solution.
3B-AP-16	Demonstrate code reuse by creating programming solutions using libraries and APIs.
3B-AP-21	Develop and use a series of test cases to verify that a program performs according to its design specifications.
3B-AP-22	Modify an existing program to add additional functionality and discuss intended and unintended implications (e.g., breaking other functionality).
3B-AP-23	Evaluate key qualities of a program through a process such as a code review.

The following Unit Goals align with the 2016 International Society for Technology in Education (ISTE) Standards.

ISTE Empowered Learner (Standard 1)	Students leverage technology to take an active role in choosing, achieving and demonstrating competency in their learning goals, informed by the learning sciences.
ISTE Digital Citizen (Standard 2)	Students recognize the rights, responsibilities and opportunities of living, learning and working in an interconnected digital world, and they act and model in ways that are safe, legal and ethical.
ISTE Knowledge Constructor (Standard 3)	Students critically curate a variety of resources using digital tools to construct knowledge, produce creative artifacts and make meaningful learning experiences for themselves and others.
ISTE Innovative Designer (Standard 4)	Students use a variety of technologies within a design process to identify and solve problems by creating new, useful or imaginative solutions.
ISTE Computational Thinker (Standard 5)	Students develop and employ strategies for understanding and solving problems in ways that leverage the power of technological methods to develop and test solutions.
ISTE Creative Communicator (Standard 6)	Students communicate clearly and express themselves creatively for a variety of purposes using the platforms, tools, styles, formats and digital media appropriate to their goals.

Unit Essential Questions

- Why is it beneficial to have sorted data sets?
- What are the procedural differences between the selection and insertion sorts?
- What are the similarities and differences between arrays and `ArrayList`s?
- When should a programmer use an `ArrayList` instead of an array?

Scope and Sequence

- The algorithms for the selection and insertion sorts, which result in sorted data that will enable the binary search method, which is more efficient than the sequential/linear search method
- The `ArrayList` class, which contains methods allowing the management of `ArrayList` objects, which change size as needed, and with elements that can be inserted and removed; many in-class examples are explore to gain practice with `ArrayList`s.

- Completion of Activities 2-4 of College Board’s *Elevens Lab*, creating a deck of cards and shuffling the deck by removing and inserting Card objects within the Deck `ArrayList`

Assured Assessments

Formative Assessments:
- Students will complete several in-class assignments/activities based on the topics in the unit, as well as explorations in *CSAwesome* and *CodeHS*.
- Students will complete review questions (multiple-choice, true/false) from the textbook.

Summative Assessments:
- Students will complete programming projects covering `ArrayList`s.
- Students will complete Activities 2-4 of College Board’s *Elevens Lab*.
- Students will take a common end-of-unit assessment scored via a common scoring guide.

Resources

Core
- Lewis, John, William Loftus, and Cara Cocking. *Java Software Solutions for AP Computer Science*. 3rd ed. Upper Saddle River, NJ: Prentice Hall, 2010. Print. Chp. 6: 6.3, 6.4, 6.7.
- Laptop or desktop with a Java IDE installed (preferably BlueJ), and Internet access
- *CodeHS*. https://codehs.com/. Web.
- *CS Awesome*. https://runestone.academy/runestone/books/published/csawesome/index.html. Web.
- College Board. *AP Computer Science Elevens Lab Student Guide*. https://secure-media.collegeboard.org/digitalServices/pdf/ap/elevens-lab-student-guide.pdf. Web.

Supplemental
- *CodingBat*. https://codingbat.com/java. Web.

Time Allotment

- Approximately 15 school days

UNIT 9
Inheritance

Unit 9 explains inheritance, a fundamental technique for organizing and creating classes. Students explore how classes can be related to form inheritance hierarchies and how these relationships allow one to create polymorphic references; students also revisit the concept of a formal Java interface and abstract classes.

Unit Goals

At the completion of this unit, students will:

The following Unit Goals align with the 2019 College Board Curriculum Framework for Advanced Placement Computer Science A.

MOD-3.B	Create an inheritance relationship from a subclass to the superclass.
MOD-3.C	Define reference variables of a superclass to be assigned to an object of a subclass in the same hierarchy.
MOD-3.D	Call methods in an inheritance relationship.
MOD-3.E	Call `Object` class methods through inheritance.

The following Unit Goals align with the 2017 Computer Science Teachers' Association (CSTA) Computer Science Standards.

3A-DA-09	Translate between different bit representations of real-world phenomena, such as characters, numbers, and images.
3A-DA-10	Evaluate the tradeoffs in how data elements are organized and where data is stored.
3A-DA-12	Create computational models that represent the relationships among different elements of data collected from a phenomenon or process.
3A-AP-13	Create prototypes that use algorithms to solve computational problems by leveraging prior student knowledge and personal interests.
3A-AP-14	Use lists to simplify solutions, generalizing computational problems instead of repeatedly using simple variables.
3A-AP-15	Justify the selection of specific control structures when tradeoffs involve implementation, readability, and program performance, and explain the benefits and drawbacks of choices made.
3A-AP-16	Design and iteratively develop computational artifacts for practical intent, personal expression, or to address a societal issue by using events to initiate instructions.

3A-AP-17	Decompose problems into smaller components through systematic analysis, using constructs such as procedures, modules, and/or objects.
3A-AP-18	Create artifacts by using procedures within a program, combinations of data and procedures, or independent but interrelated programs.
3A-AP-19	Systematically design and develop programs for broad audiences by incorporating feedback from users.
3A-IC-24	Evaluate the ways computing impacts personal, ethical, social, economic, and cultural practices.
3A-IC-25	Test and refine computational artifacts to reduce bias and equity deficits.
3B-DA-07	Evaluate the ability of models and simulations to test and support the refinement of hypotheses.
3B-AP-08	Describe how artificial intelligence drives many software and physical systems.
3B-AP-09	Implement an artificial intelligence algorithm to play a game against a human opponent or solve a problem.
3B-AP-10	Use and adapt classic algorithms to solve computational problems.
3B-AP-11	Evaluate algorithms in terms of their efficiency, correctness, and clarity.
3B-AP-12	Compare and contrast fundamental data structures and their uses.
3B-AP-14	Construct solutions to problems using student-created components, such as procedures, modules and/or objects.
3B-AP-15	Analyze a large-scale computational problem and identify generalizable patterns that can be applied to a solution.
3B-AP-16	Demonstrate code reuse by creating programming solutions using libraries and APIs.
3B-AP-21	Develop and use a series of test cases to verify that a program performs according to its design specifications.
3B-AP-22	Modify an existing program to add additional functionality and discuss intended and unintended implications (e.g., breaking other functionality).
3B-AP-23	Evaluate key qualities of a program through a process such as a code review.
3B-IC-27	Predict how computational innovations that have revolutionized aspects of our culture might evolve.

The following Unit Goals align with the 2016 International Society for Technology in Education (ISTE) Standards.

ISTE Empowered Learner (Standard 1)	Students leverage technology to take an active role in choosing, achieving and demonstrating competency in their learning goals, informed by the learning sciences.
ISTE Digital Citizen (Standard 2)	Students recognize the rights, responsibilities and opportunities of living, learning and working in an interconnected digital world, and they act and model in ways that are safe, legal and ethical.
ISTE Knowledge Constructor (Standard 3)	Students critically curate a variety of resources using digital tools to construct knowledge, produce creative artifacts and make meaningful learning experiences for themselves and others.
ISTE Innovative Designer (Standard 4)	Students use a variety of technologies within a design process to identify and solve problems by creating new, useful or imaginative solutions.
ISTE Computational Thinker (Standard 5)	Students develop and employ strategies for understanding and solving problems in ways that leverage the power of technological methods to develop and test solutions.
ISTE Creative Communicator (Standard 6)	Students communicate clearly and express themselves creatively for a variety of purposes using the platforms, tools, styles, formats and digital media appropriate to their goals.

Unit Essential Questions

- How are new classes derived from existing ones?
- What is software reuse, and how is it connected to inheritance?
- How are methods in child classes added and/or modified?
- What are class hierarchies, and how do they help with creating classes?
- What is polymorphism, and how is it used?

Scope and Sequence

- Exploration of the topic of inheritance: creating a new class from existing ones. The proper vocabulary is stressed: “base class,” “superclass,” “parent class,” “derived class,” “subclass,” and “child class.” Students practice identifying and creating these classes using *CSAwesome* and other digital materials. Class hierarchies are also introduced to help students understand the concept of inheritance.
- Related to the study of inheritance, the tracing of many sample programs to show the extent of when and where to override methods

- The concept of polymorphism, which appears frequently when using inheritance; the focus is object references, and how to create polymorphic references that can refer to different types of objects at different times.
- Global strategies to help plan for the use of inheritance and polymorphism, for the most elegant programs possible
- Completion of Activities 6-9 of College Board's *Elevens Lab*, completing programming the game by using inheritance and polymorphism

Assured Assessments

Formative Assessments:

- Students will complete several in-class assignments/activities based on the topics in the unit, as well as explorations in *CSAwesome* and *CodeHS*.
- Students will complete Activities 6-9 of College Board's *Elevens Lab*.
- Students will complete review questions (multiple-choice, true/false) from the textbook.

Summative Assessments:

- Students will complete programming projects covering inheritance and polymorphism.
- Students will take a common end-of-unit assessment scored via a common scoring guide.

Resources

Core

- Lewis, John, William Loftus, and Cara Cocking. *Java Software Solutions for AP Computer Science*. 3rd ed. Upper Saddle River, NJ: Prentice Hall, 2010. Print. Chp. 7: 7.0, 7.1, 7.2, 7.3, 7.4, 7.5, 7.6, 7.7.
- Laptop or desktop with a Java IDE installed (preferably BlueJ), and Internet access
- *CodeHS*. https://codehs.com/. Web.
- *CS Awesome*. https://runestone.academy/runestone/books/published/csawesome/index.html. Web.
- College Board. *AP Computer Science Elevens Lab Student Guide*. https://secure-media.collegeboard.org/digitalServices/pdf/ap/elevens-lab-student-guide.pdf. Web.

Supplemental

- *CodingBat*. https://codingbat.com/java. Web.

Time Allotment

- Approximately 12 school days

UNIT 10
Recursion, Merge, and Quick Sorts

Unit 10 provides an introduction to recursive programming, which provides elegant solutions to certain problems.

Unit Goals

At the completion of this unit, students will:

The following Unit Goals align with the 2019 College Board Curriculum Framework for Advanced Placement Computer Science A.

CON-2.O	Determine the result of executing recursive methods
CON-2.P	Apply recursive search algorithms to information in `String`, 1D array, or `ArrayList` objects.
CON-2.Q	Apply recursive algorithms to sort elements of array or `ArrayList` objects.

The following Unit Goals align with the 2017 Computer Science Teachers' Association (CSTA) Computer Science Standards.

3A-DA-09	Translate between different bit representations of real-world phenomena, such as characters, numbers, and images.
3A-DA-10	Evaluate the tradeoffs in how data elements are organized and where data is stored.
3A-DA-12	Create computational models that represent the relationships among different elements of data collected from a phenomenon or process.
3A-AP-13	Create prototypes that use algorithms to solve computational problems by leveraging prior student knowledge and personal interests.
3A-AP-14	Use lists to simplify solutions, generalizing computational problems instead of repeatedly using simple variables.
3A-AP-15	Justify the selection of specific control structures when tradeoffs involve implementation, readability, and program performance, and explain the benefits and drawbacks of choices made.
3A-AP-16	Design and iteratively develop computational artifacts for practical intent, personal expression, or to address a societal issue by using events to initiate instructions.
3A-AP-17	Decompose problems into smaller components through systematic analysis, using constructs such as procedures, modules, and/or objects.

3A-AP-18	Create artifacts by using procedures within a program, combinations of data and procedures, or independent but interrelated programs.
3A-AP-19	Systematically design and develop programs for broad audiences by incorporating feedback from users.
3B-DA-07	Evaluate the ability of models and simulations to test and support the refinement of hypotheses.
3B-AP-08	Describe how artificial intelligence drives many software and physical systems.
3B-AP-09	Implement an artificial intelligence algorithm to play a game against a human opponent or solve a problem.
3B-AP-10	Use and adapt classic algorithms to solve computational problems.
3B-AP-11	Evaluate algorithms in terms of their efficiency, correctness, and clarity.
3B-AP-12	Compare and contrast fundamental data structures and their uses.
3B-AP-13	Illustrate the flow of execution of a recursive algorithm.
3B-AP-14	Construct solutions to problems using student-created components, such as procedures, modules and/or objects.
3B-AP-15	Analyze a large-scale computational problem and identify generalizable patterns that can be applied to a solution.
3B-AP-16	Demonstrate code reuse by creating programming solutions using libraries and APIs.
3B-AP-21	Develop and use a series of test cases to verify that a program performs according to its design specifications.
3B-AP-22	Modify an existing program to add additional functionality and discuss intended and unintended implications (e.g., breaking other functionality).
3B-AP-23	Evaluate key qualities of a program through a process such as a code review.
3B-IC-27	Predict how computational innovations that have revolutionized aspects of our culture might evolve.

The following Unit Goals align with the 2016 International Society for Technology in Education (ISTE) Standards.

ISTE Empowered Learner (Standard 1)	Students leverage technology to take an active role in choosing, achieving and demonstrating competency in their learning goals, informed by the learning sciences.

ISTE Knowledge Constructor (Standard 3)	Students critically curate a variety of resources using digital tools to construct knowledge, produce creative artifacts and make meaningful learning experiences for themselves and others.
ISTE Innovative Designer (Standard 4)	Students use a variety of technologies within a design process to identify and solve problems by creating new, useful or imaginative solutions.
ISTE Computational Thinker (Standard 5)	Students develop and employ strategies for understanding and solving problems in ways that leverage the power of technological methods to develop and test solutions.

Unit Essential Questions

- What is recursion?
- What are the main components of a recursive method?
- What is infinite recursion, and how can it be avoided?
- When should recursion be used or not used?
- How can recursion be used in sorting data?

Scope and Sequence

- Significant in-class tracing of examples of recursion using *CSAwesome* and *CodeHS*; subsequent time writing recursive methods in class and on *CodingBat*
- The merge and quick sort methods to sort data, each of which uses recursion to complete the same task as the previously-studied selection and insertion sorting methods; however, since they use recursion, the programming is more elegant.

Assured Assessments

Formative Assessments:
- Students will complete several in-class assignments/activities based on the topics in the unit, including programming tasks on *CodingBat*.
- Students will complete review questions (multiple-choice, true/false) from the textbook.

Summative Assessments:
- Students will complete programming projects covering recursion.
- Students will take a common end-of-unit assessment scored via a common scoring guide.

Resources

Core
- Lewis, John, William Loftus, and Cara Cocking. *Java Software Solutions for AP Computer Science*. 3rd ed. Upper Saddle River, NJ: Prentice Hall, 2010. Print. Chp. 8: 8.0, 8.1, 8.2, 8.3.
- Laptop or desktop with a Java IDE installed (preferably BlueJ), and Internet access
- *CodeHS*. https://codehs.com/. Web.

- *CodingBat*. https://codingbat.com/java. Web.
- *CS Awesome*. https://runestone.academy/runestone/books/published/csawesome/index.html. Web.
- College Board. *AP Computer Science Elevens Lab Student Guide*. https://secure-media.collegeboard.org/digitalServices/pdf/ap/elevens-lab-student-guide.pdf. Web.

Time Allotment

- Approximately 12 school days

UNIT 11
College Board Examination Preparation

Unit 11 focuses on the details of the College Board Examination in Advanced Placement Computer Science Principles; its goals are all the goals of the prior units.

Unit Essential Questions

- What is the format, style, and length of the College Board Examination in Advanced Placement Computer Science A?
- What is the AP CSA Java Quick Reference Sheet, and how can I successfully use it?
- How are free-response questions scored on the AP CSA College Board Examination?

Scope and Sequence

- Review of many examples from prior, released AP CSA College Board Examinations, including multiple-choice questions and free-response questions
- Scoring and discussion of sample responses from prior, released AP CSA College Board Examinations

Assured Assessments

Formative Assessments:
- Students will complete several in-class assignments/activities based on prior, released AP CSA College Board Examinations, including multiple-choice questions and free-response questions

Summative Assessments:
- Students will participate in a full released AP CSA College Board Examination.

Resources

Core
- Lewis, John, William Loftus, and Cara Cocking. *Java Software Solutions for AP Computer Science*. 3rd ed. Upper Saddle River, NJ: Prentice Hall, 2010. Print.
- Laptop or desktop with a Java IDE installed (preferably BlueJ), and Internet access

Supplemental
- *CodeHS*. https://codehs.com/. Web.
- *CodingBat*. https://codingbat.com/java. Web.
- *CS Awesome*. https://runestone.academy/runestone/books/published/csawesome/index.html. Web.
- Teukolsky, Roselyn. *Barron's AP Computer Science A*. 8th ed. Hauppauge, NY: Barron's, 2018. Print.

Time Allotment

- Approximately 10 school days

UNIT 12
Graphics

In Unit 12, students put all of the skills they have learned during the course to use with graphics; though this topic is not part of the College Board framework for AP Computer Science A, it allows students to branch out and apply and adapt their prior knowledge.

Unit Goals

At the completion of this unit, students will:

The following Unit Goals align with the 2017 Computer Science Teachers' Association (CSTA) Computer Science Standards.

3A-DA-12	Create computational models that represent the relationships among different elements of data collected from a phenomenon or process.
3A-AP-13	Create prototypes that use algorithms to solve computational problems by leveraging prior student knowledge and personal interests.
3A-AP-15	Justify the selection of specific control structures when tradeoffs involve implementation, readability, and program performance, and explain the benefits and drawbacks of choices made.
3A-AP-16	Design and iteratively develop computational artifacts for practical intent, personal expression, or to address a societal issue by using events to initiate instructions.
3A-AP-17	Decompose problems into smaller components through systematic analysis, using constructs such as procedures, modules, and/or objects.
3A-AP-18	Create artifacts by using procedures within a program, combinations of data and procedures, or independent but interrelated programs.
3A-AP-19	Systematically design and develop programs for broad audiences by incorporating feedback from users.
3A-IC-24	Evaluate the ways computing impacts personal, ethical, social, economic, and cultural practices.
3B-DA-07	Evaluate the ability of models and simulations to test and support the refinement of hypotheses.
3B-AP-08	Describe how artificial intelligence drives many software and physical systems.
3B-AP-09	Implement an artificial intelligence algorithm to play a game against a human opponent or solve a problem.
3B-AP-10	Use and adapt classic algorithms to solve computational problems.

3B-AP-11	Evaluate algorithms in terms of their efficiency, correctness, and clarity.
3B-AP-12	Compare and contrast fundamental data structures and their uses.
3B-AP-14	Construct solutions to problems using student-created components, such as procedures, modules and/or objects.
3B-AP-15	Analyze a large-scale computational problem and identify generalizable patterns that can be applied to a solution.
3B-AP-16	Demonstrate code reuse by creating programming solutions using libraries and APIs.
3B-AP-21	Develop and use a series of test cases to verify that a program performs according to its design specifications.
3B-AP-22	Modify an existing program to add additional functionality and discuss intended and unintended implications (e.g., breaking other functionality).
3B-AP-23	Evaluate key qualities of a program through a process such as a code review.

The following Unit Goals align with the 2016 International Society for Technology in Education (ISTE) Standards.

ISTE Empowered Learner (Standard 1)	Students leverage technology to take an active role in choosing, achieving and demonstrating competency in their learning goals, informed by the learning sciences.
ISTE Knowledge Constructor (Standard 3)	Students critically curate a variety of resources using digital tools to construct knowledge, produce creative artifacts and make meaningful learning experiences for themselves and others.
ISTE Innovative Designer (Standard 4)	Students use a variety of technologies within a design process to identify and solve problems by creating new, useful or imaginative solutions.
ISTE Computational Thinker (Standard 5)	Students develop and employ strategies for understanding and solving problems in ways that leverage the power of technological methods to develop and test solutions.
ISTE Creative Communicator (Standard 6)	Students communicate clearly and express themselves creatively for a variety of purposes using the platforms, tools, styles, formats and digital media appropriate to their goals.
ISTE Global Collaborator (Standard 7)	Students use digital tools to broaden their perspectives and enrich their learning by collaborating with others and

working effectively in teams locally and globally.

Unit Essential Questions

- What are applets?
- What is a GUI?
- How are algorithms adapted to create graphics and interactive objects on the screen?

Scope and Sequence

- Introduction to some of the graphical elements of Java; by examining programs from the textbook, students apply and adapt the Java skills learned in the course to create different graphical projects, and learn new programming skills as necessary.

Assured Assessments

Formative Assessments:

- Students will complete several in-class assignments/activities based on the topics in the unit.
- Students will complete review questions (multiple-choice, true/false) from the textbook.

Summative Assessments:

- Students will complete programming projects covering graphics, applets, dialog boxes, and different types of buttons.
- Students will take a common end-of-unit assessment scored via a common scoring guide.

Resources

Core

- Lewis, John, William Loftus, and Cara Cocking. *Java Software Solutions for AP Computer Science*. 3rd ed. Upper Saddle River, NJ: Prentice Hall, 2010. Print. Chps. 1.5, 2.11, 2.12, 3.9, 4.6, 4.7, 5.5, 5.6, 5.7, 6.8, 6.9.
- Laptop or desktop with a Java IDE installed (preferably BlueJ), and Internet access

Time Allotment

- Approximately 15 school days

COURSE CREDIT

One credit in STEM
One class period daily for a full year

PREREQUISITES

Successful completion of ACP Algebra II, or concurrent enrollment in ACP Algebra II with Department Chair permission.

SUPPLEMENTARY MATERIALS/RESOURCES/TECHNOLOGY

Department- and teacher-prepared materials

Laptop or desktop with a Java IDE installed (preferably BlueJ), and Internet access

College Board website including past Advanced Placement Computer Science A tests

CURRENT REFERENCE

College Board: Advanced Placement Computer Science A:

https://apcentral.collegeboard.org/courses/ap-computer-science-a?course=ap-computer-science-a

ASSURED STUDENT PERFORMANCE RUBRICS

- Trumbull High School School-Wide Writing Rubric (attached)
- Trumbull High School School-Wide Problem-Solving Rubric (attached)
- Trumbull High School School-Wide Independent Learning and Thinking Rubric (attached)

Trumbull High School School-Wide Writing Rubric

Category/ Weight	**Exemplary 4 Student work:**	**Goal 3 Student work:**	**Working Toward Goal 2 Student work:**	**Needs Support 1-0 Student work:**
Purpose X_______	• Establishes and maintains a clear purpose • Demonstrates an insightful understanding of audience and task	• Establishes and maintains a purpose • Demonstrates an accurate awareness of audience and task	• Establishes a purpose • Demonstrates an awareness of audience and task	• Does not establish a clear purpose • Demonstrates limited/no awareness of audience and task
Organization X_______	• Reflects sophisticated organization throughout • Demonstrates logical progression of ideas • Maintains a clear focus • Utilizes effective transitions	• Reflects organization throughout • Demonstrates logical progression of ideas • Maintains a focus • Utilizes transitions	• Reflects some organization throughout • Demonstrates logical progression of ideas at times • Maintains a vague focus • May utilize some ineffective transitions	• Reflects little/no organization • Lacks logical progression of ideas • Maintains little/no focus • Utilizes ineffective or no transitions
Content X_______	• Is accurate, explicit, and vivid • Exhibits ideas that are highly developed and enhanced by specific details and examples	• Is accurate and relevant • Exhibits ideas that are developed and supported by details and examples	• May contain some inaccuracies • Exhibits ideas that are partially supported by details and examples	• Is inaccurate and unclear • Exhibits limited/no ideas supported by specific details and examples
Use of Language X_______	• Demonstrates excellent use of language • Demonstrates a highly effective use of standard writing that enhances communication • Contains few or no errors. Errors do not detract from meaning	• Demonstrates competent use of language • Demonstrates effective use of standard writing conventions • Contains few errors Most errors do not detract from meaning	• Demonstrates use of language • Demonstrates use of standard writing conventions • Contains errors that detract from meaning	• Demonstrates limited competency in use of language • Demonstrates limited use of standard writing conventions • Contains errors that make it difficult to determine meaning

Trumbull High School School-Wide Problem-Solving Rubric

Category/ Weight	**Exemplary 4**	**Goal 3**	**Working Toward Goal 2**	**Needs Support 1-0**
Understanding X_______	• Student demonstrates clear understanding of the problem and the complexities of the task	• Student demonstrates sufficient understanding of the problem and most of the complexities of the task	• Student demonstrates some understanding of the problem but requires assistance to complete the task	• Student demonstrates limited or no understanding of the fundamental problem after assistance with the task
Research X_______	• Student gathers compelling information from multiple sources including digital, print, and interpersonal	• Student gathers sufficient information from multiple sources including digital, print, and interpersonal	• Student gathers some information from few sources including digital, print, and interpersonal	• Student gathers limited or no information
Reasoning and Strategies X_______	• Student demonstrates strong critical thinking skills to develop a comprehensive plan integrating multiple strategies	• Student demonstrates sufficient critical thinking skills to develop a cohesive plan integrating strategies	• Student demonstrates some critical thinking skills to develop a plan integrating some strategies	• Student demonstrates limited or no critical thinking skills and no plan
Final Product and/or Presentation X_______	• Solution shows deep understanding of the problem and its components • Solution shows extensive use of 21st-century technology skills	• Solution shows sufficient understanding of the problem and its components • Solution shows sufficient use of 21st-century technology skills	• Solution shows some understanding of the problem and its components • Solution shows some use of 21st-century technology skills	• Solution shows limited or no understanding of the problem and its components • Solution shows limited or no use of 21st-century technology skills

Trumbull High School School-Wide
Independent Learning and Thinking Rubric

Category/ Weight	**Exemplary 4**	**Goal 3**	**Working Toward Goal 2**	**Needs Support 1-0**
Proposal X_______	• Student demonstrates a strong sense of initiative by generating compelling questions, creating uniquely original projects/work	• Student demonstrates initiative by generating appropriate questions, creating original projects/work	• Student demonstrates some initiative by generating questions, creating appropriate projects/work	• Student demonstrates limited or no initiative by generating few questions and creating projects/work
Independent Research & Development X_______	• Student is analytical, insightful, and works independently to reach a solution	• Student is analytical, and works productively to reach a solution	• Student reaches a solution with direction	• Student is unable to reach a solution without consistent assistance
Presentation of Final Product X_______	• Presentation shows compelling evidence of an independent learner and thinker • Solution shows deep understanding of the problem and its components • Solution shows extensive and appropriate application of 21st-century skills	• Presentation shows clear evidence of an independent learner and thinker • Solution shows adequate understanding of the problem and its components • Solution shows adequate application of 21st-century skills	• Presentation shows some evidence of an independent learner and thinker • Solution shows some understanding of the problem and its components • Solution shows some application of 21st-century skills	• Presentation shows limited or no evidence of an independent learner and thinker • Solution shows limited or no understanding of the problem and its components • Solution shows limited or no application of 21st-century skills