# Verona Public School District
# Curriculum Overview

## AP Computer Science Principles

**Curriculum Committee Members:**
Danielle Mutovic

**Supervisor:**
Glen Stevenson

**Curriculum Developed:**
Summer 2016
Summer 2017

**Board Approval Date:**
August 30, 2016
August 29, 2017

Verona Public Schools
121 Fairview Ave., Verona, NJ 07044
www.veronaschools.org

**Verona Public Schools Mission Statement:**

The mission of the Verona Public Schools, the center of an engaged and supportive community, is to empower students to achieve their potential as active learners and productive citizens through rigorous curricula and meaningful, enriching experiences.

**Course Description:**

AP Computer Science Principles introduces students to the foundational concepts of computer science and challenges them to explore how computing and technology can impact the world. With a unique focus on creative problem solving and real world applications, AP Computer Science Principles prepares students for college and career. This course may precede or follow AP Computer Science A. Student must take the AP exam to earn AP course credit.

**Prerequisite(s): Algebra I**

# Verona High School AP Computer Science Principles

| Standard 8: Technology Standards | |
|---|---|
| **8.1: Educational Technology:** ***All students will use digital tools to access, manage, evaluate, and synthesize information in order to solve problems individually and collaborate and to create and communicate knowledge.*** | **8.2: Technology Education, Engineering, Design, and Computational Thinking - Programming:** ***All students will develop an understanding of the nature and impact of technolog engineering, technological design, computational thinking and the designed world as they relate individual, global society, and the environment.*** |
| x A. Technology Operations and Concepts<br>x B. Creativity and Innovation<br>x C. Communication and Collaboration<br>x D. Digital Citizenship<br>x E. Research and Information Fluency<br>x F. Critical thinking, problem solving, and decision making | x A. The Nature of Technology: Creativity and Innovation<br>x B. Technology and Society<br>x C. Design<br>x D. Abilities for a Technological World<br>x E. Computational Thinking: Programming |

| SEL Competencies and Career Ready Practices | |
|---|---|
| **Social and Emotional Learning Core Competencies:** ***These competencies are identified as five interrelated sets of cognitive, affective, and behavioral capabilities*** | **Career Ready Practices:** ***These practices outline the skills that all individuals need to have to truly be adaptable, reflective, and proactive in life and careers. These are researched practices that are essential to career readiness.*** |
| **Self-awareness:** The ability to accurately recognize one's emotions and thoughts and their influence on behavior. This includes accurately assessing one's strengths and limitations and possessing a well-grounded sense of confidence and optimism. | x CRP2. Apply appropriate academic and technical skills.<br>x CRP9. Model integrity, ethical leadership, and effective management.<br>x CRP10. Plan education and career paths aligned to personal goals. |
| **Self-management:** The ability to regulate one's emotions, thoughts, and behaviors effectively in different situations. This includes managing stress, controlling impulses, motivating oneself, and setting and working toward achieving personal and academic goals. | x CRP3. Attend to personal health and financial well-being.<br>x CRP6. Demonstrate creativity and innovation.<br>x CRP8. Utilize critical thinking to make sense of problems and persevere in solving them.<br>x CRP11. Use technology to enhance productivity. |
| **Social awareness:** The ability to take the perspective of and empathize with others from diverse backgrounds and cultures, to understand social and ethical norms for behavior, and to recognize family, school, and community resources and supports. | x CRP1. Act as a responsible and contributing citizen and employee.<br>x CRP9. Model integrity, ethical leadership, and effective management. |
| **Relationship skills:** The ability to establish and maintain healthy and rewarding relationships with diverse individuals and groups. This includes communicating clearly, listening actively, cooperating, resisting inappropriate social pressure, negotiating conflict constructively, and seeking and offering help when needed. | x CRP4. Communicate clearly and effectively and with reason.<br>x CRP9. Model integrity, ethical leadership, and effective management.<br>x CRP12. Work productively in teams while using cultural global competence. |
| **Responsible decision making:** The ability to make constructive and respectful choices about personal behavior and social interactions based on consideration of ethical standards, safety concerns, social norms, the realistic evaluation of consequences of various actions, and the well-being of self and others. | x CRP5. Consider the environmental, social, and economic impact of decisions.<br>x CRP7. Employ valid and reliable research strategies.<br>x CRP8. Utilize critical thinking to make sense of problems and persevere in solving them.<br>x CRP9. Model integrity, ethical leadership, and effective management. |

| Standard 9: 21st Century Life and Careers | | |
|---|---|---|
| **9.1: Personal Financial Literacy:** ***This standard outlines the important fiscal knowledge, habits, and skills that must mastered in order for students to make informed decisions about personal finance. Financial literacy is an integral component of a student's college and career readiness, enabling students to achieve fulfilling, financially-secure, a successful careers.*** | **9.2: Career Awareness, Exploration & Preparation:** ***This standard outlines the importance of being knowledgeab about one's interests and talents, and being well informed about postsecondary and career options, career planning, a career requirements.*** | **9.3: Career and Technical Education:** ***This standard outlines what students should know and be able to do upon completion of a CTE Program of Study.*** |
| A. Income and Careers<br>B. Money Management<br>C. Credit and Debt Management<br>D. Planning, Saving, and Investing<br>E. Becoming a Critical Consumer<br>F. Civic Financial Responsibility<br>G. Insuring and Protecting | A. Career Awareness (K-4)<br>B. Career Exploration (5-8)<br>x C. Career Preparation (9-12) | A. Agriculture, Food & Natural Res.<br>B. Architecture & Construction<br>C. Arts, A/V Technology & Comm.<br>D. Business Management & Admin.<br>E. Education & Training<br>F. Finance<br>G. Government & Public Admin.<br>H. Health Science<br>I. Hospital & Tourism<br>J. Human Services<br>K. Information Technology<br>L. Law, Public, Safety, Corrections & Security<br>M. Manufacturing<br>N. Marketing<br>x O. Science, Technology, Engineering & Math<br>P. Transportation, Distribution & Log. |

| Course Materials | |
|---|---|
| Core Instructional Materials: ***These are the board adopted and approved materials to support the curriculum, instruction, and assessment of this course.*** | Differentiated Resources: ***These are teacher and department found materials, and also approved support materials that facilitate differentiation of curriculum, instruction, and assessment of this course.*** |
| • CS50 AP Portal<br>• Edx and CS50.io<br>• MIT AppInventor | • CS50 AP Portal<br>• Writing Assignments<br>• Programming assignments |

# Verona High School AP Computer Science Principles

| Unit 0: Computers and Computing | Unit Duration: 2 - 3 weeks |
|---|---|

## Stage 1: Desired Results

**Established Goals:**

### AP CSP Learning Objectives:

LO 1.2.5 Analyze the correctness, usability, functionality, and suitability of computational artifacts. [P4]
LO 7.1.1 Explain how computing innovations affect communication, interaction, and cognition. [P4]
LO 7.2.1 Explain how computing has impacted innovations in other fields. [P1]
LO 7.3.1 Analyze the beneficial and harmful effects of computing. [P4]
LO 4.1.1 Develop an algorithm for implementation in a program. [P2]
LO 4.1.2 Express an algorithm in a language. [P5]
LO 7.5.2 Evaluate online and print sources for appropriateness and credibility. [P5]

### AP CSP Curricular Requirements:

CR1a - Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P1: Connected Computing.
CR1b - Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P2: Creating Computational Artifacts.
CR1d - Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P4: Analyzing Problems and Artifacts.
CR1e - Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P5: Communicating.
CR2a - Students are provided with opportunities to meet learning objectives within Big Idea 1: Creativity. Such opportunities must occur in addition to the AP Computer Science Principles Performance Tasks.
CR2d - Students are provided with opportunities to meet learning objectives within Big Idea 4: Algorithms. Such opportunities must occur in addition to the AP Computer Science Principles Performance Tasks.
CR2g - Students are provided with opportunities to meet learning objectives connected within Big Idea 7: Global Impact. Such opportunities must occur in addition to the AP Computer Science Principles Performance Task.

### Transfer Goal:

Students will be able to independently use their learning to … understand how information is used by a computer so that humans can understand it and begin to explore ways in which computers process information

### Students will understand that:

- Many current devices can be considered computers.
- Computers perform a multitude of tasks in small amounts of time.
- Computers only do what they are told to do.
- There are many pieces that go into a current computer.
- Data and Storage on a computer all boils down to a fundamental unit of measure - a bit.
- All the parts of a computer work together.
- Memory is a broad term for information holders in a computer.
- Number systems can have different bases than 10.
- Binary numbers allow us to have the ASCII encoding scheme.
- Computers only can do what a human tells them to do.

### Essential Questions:

- What is a computer?
- Is computing exclusive to computers?
- Are computers always correct?
- What is under the hood of a computer?
- What happens when the power button is pressed?
- What is memory?
- What if there were only two digits, 0 and 1?
- Does a computer know the English alphabet?
- Are computer's intelligent?

### Students will know:

- How to define a computer
- Computers can be used to do a variety of things.
- How to define an algorithm.
- The technical terms and acronyms used in computing language.
- Bits have a value of yes or no.
- Computers have a lot of moving parts.
- Memory covers 5 different parts of a computer.
- How computers use the binary number system.
- How computers understand information from a keyboard.
- Definition of an algorithm.

### Students will be able to:

- Identify when a computer is a valuable tool.
- Recognize processes we will later call algorithms.
- Speak competently about computers in conversations with others.
- Compare old data formats to current ones.
- Rationalize the amount of data current devices can hold today.
- Think more about the technological implications of hardware.
- Use memory, with respect to computers, correctly.
- Use the binary number system.
- Detect patterns in the ASCII encoding scheme and see its meaning.
- Write and understand algorithms and their meanings.

## Stage 2: Acceptable Evidence

### Performance Task & Unit Assessments:

**Assessments -**

- Unit 0 Quiz 1
- Unit 0 Test 1

**Transfer Tasks:**

- Writing Problem 0-0: Around the House
  In no more than 400 words, describe a device in your home that you believe is a computer and answer questions pertaining to its possible definition.
  Teacher Notes
- Writing Problem 0-1: Tech Spotlight
  In no more than 600 words, describe and expand upon a piece of technology that you are familiar with providing and unbiased summary to a “non-tech” person.
  Teacher Notes
- Writing Problem 0-2: Everyday Algorithms
  Students will read and watch videos to become more familiar with algorithms and pseudocode. Then, students will write algorithms in sentence and pseudocode.
  Teacher Notes

### Other Evidence:
### Students will show that they have achieved Stage 1 goals by:

Formal:
- Providing written/oral response to the EQs
- Passing all quizzes on basic concepts in unit.

Informal:
- Identifying and explaining the purpose of major parts of a computer
- Identifying computing objects
- Beginning to write technologically
- Understanding different number/text systems
- Students should also troubleshoot examples and explain misapplications of the conventions or properties.

## Reference Materials

# Verona High School AP Computer Science Principles

- Verona AP CS50 - Unit 0
- 0-0 Teacher Notes - Computers and Computing
- 0-1 Teacher Notes - How Computers Work
    - Module 0-1 Practice Problems
- 0-2 Teacher Notes - Bits and Bytes
    - Bits and Bytes Activity
- 0-3 Teacher Notes - Hardware
- 0-4 Teacher Notes - Memory
- 0-5 Teacher Notes - Binary
    - Binary HW
- 0-6 Teacher Notes - ASCII
    - ASCII HW
- 0-7 Teacher Notes - Algorithms

# Verona High School AP Computer Science Principles

| Unit 1: Building Blocks of Programming | Unit Duration: 3 - 4 weeks |
|---|---|

## Stage 1: Desired Results

### Established Goals:

**AP CSP Learning Objectives:**

LO 1.1.1 Apply a creative development process when creating computational artifacts. [P2]
LO 1.2.1 Create a computational artifact for creative expression. [P2]
LO 1.2.2 Create a computational artifact using computing tools and techniques to solve a problem. [P2]
LO 1.2.3 Create a new computational artifact by combining or modifying existing artifacts. [P2]
LO 1.3.1 Use computing tools and techniques for creative expression. [P2]
LO 2.1.2 Explain how binary sequences are used to represent digital data. [P5]
LO 2.2.2 Use multiple levels of abstraction to write programs. [P3]
LO 4.1.1 Develop an algorithm for implementation in a program. [P2]
LO 4.1.2 Express an algorithm in a language. [P5]
LO 5.1.1 Develop a program for creative expression, to satisfy personal curiosity, or to create new knowledge. [P2]
LO 5.1.2 Develop a correct program to solve problems. [P2]
LO 5.4.1 Evaluate the correctness of a program. [P4]
LO 5.5.1 Employ appropriate mathematical and logical concepts in programming. [P1]

**AP CSP Curricular Requirements:**

CR1a - Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P1: Connecting Computing.
CR1b - Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P2: Creating Computational Artifacts.
CR1c - Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P3: Abstracting.
CR1d - Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P4: Analyzing Problems and Artifacts.
CR1e - Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P5: Communicating.
CR2a - Students are provided with opportunities to meet learning objectives within Big Idea 1: Creativity. Such opportunities must occur in addition to the AP Computer Science Principles Performance Tasks.
CR2b - Students are provided with opportunities to meet learning objectives within Big Idea 2: Abstraction. Such opportunities must occur in addition to the AP Computer Science Principles Performance Tasks.
CR2d - Students are provided with opportunities to meet learning objectives within Big Idea 4: Algorithms. Such opportunities must occur in addition to the AP Computer Science Principles Performance Tasks.
CR2e - Students are provided with opportunities to meet learning objectives within Big Idea 5: Programming. Such opportunities must occur in addition to the AP Computer Science Principles Performance Tasks.

### Transfer Goal:

Students will be able to <u>independently</u> use their learning to… code in different languages, independently solving problems, manipulating data and command a computer to perform tasks.

### Students will understand that:

- Pseudocode is a launching point for writing in different languages.
- Scratch was designed to eliminate the “grammar” issues of programming.
- A variable is a storage container capable of holding different values.
- C is a statically-typed language; it requires that every time you declare a variable, you specify the datatype of that variable.
- Loops are how programmers do the same thing repeatedly.
- Conditional branching using Boolean expressions are how we as programmers make decisions in our programs.
- Statements are among the simplest of programming elements; simply tell the computer to do something.
- Though tricky and annoying, syntax is extremely important to modern programming.

### Essential Questions:

- Can every algorithm be expressed as pseudocode?
- Can Scratch be used to create intensively complex programs?
- Why do we need variables?
- What is the benefit of old and new programming languages?
- What is clunky?
- Is a computer capable of making decisions?
- How do programming languages compare to the syntax of spoken language?
- How would spoken language be different if grammar did not exist?

### Students will know:

- The semantics of pseudocode and why it is important.
- How to program in Scratch.
- Variables can be read from, written to, updated, or have their contents displayed.
- Modern languages, figure out the type of all the variables in question, and may convert those variables on the fly to more appropriate types if the situation requires.
- The benefits of using loops in all programming.
- The standard programming conditionals and Boolean expression syntax.
- That statements are the bulk of what fills up the body of loops or branches.
- That syntax will become second nature.

### Students will be able to:

- Write pseudocode with comments.
- Translate Scratch programs into pseudocode and vice versa.
- How to create, name and manipulate variables.
- Distinguish between data types.
- Declare and use  appropriate variables..
- Choose appropriate looping methods.
- Write programs that have different pathways using conditionals.
- Code using proper and meaningful syntax.

## Stage 2: Acceptable Evidence

### Performance Task & Unit Assessments:

**Assessments -**

- Unit 1 Exam

**Transfer Tasks:**

- Problem 1-0 - App Inventor
  Students will create their own design or animation game using App Inventor.
  Teacher Notes

### Other Evidence:
### Students will show that they have achieved Stage 1 goals by:

Formal:
- Providing written/oral response to the EQs
- Passing all quizzes on basic concepts in unit.
- Receiving passing grades on Transfer Tasks

Informal:
- Writing effective pseudocode
- Using pseudocode to write code in App Inventor and C

| | |
|---|---|
| • Problem 1-1 - App Inventor Problem Set<br>Students will modify their own work, be introduced to distribution code for modification, and practice good design in Scratch.<br>Teacher Notes<br>• Problem 1-2 - Hello<br>Students will be introduced to Cloud9 IDE, Linux and begin programming in C.<br>Teacher Notes<br>• Problem 1-3 - Fahrenheit<br>Students will use variables and create utility programs in C.<br>Teacher Notes<br>• Problem 1-4 - Skittles<br>Students will investigate random number generators and use them to create a game in C.<br>Teacher Notes<br>• Problem 1-5 - Greedy<br>Students will use algorithms to solve problems while investigating design differences.<br>Teacher Notes<br>• Problem 1-6 - Pennies<br>Students will use algorithms to solve problems and being investigating header files.<br>Teacher Notes<br>• Problem 1-7 - ISBN<br>Students will use algorithms to solve problems, be introduced to check sums and start to see how computers input large pieces of information and break it down into smaller units.<br>Teacher Notes<br>• Problem 1-8 - Mario<br>Students will continue to use loops and be introduced to terminal printing.<br>Teacher Notes<br>• Problem 1-9 - Credit<br>Students will use algorithms to solve real world problems and complete contrasting design analysis.<br>Teacher Notes | • Using correct Syntax<br>• Understanding Boolean Algebra<br>• Students should also troubleshoot examples and explain misapplications of the conventions or properties. |

## Reference Materials

- Verona AP CS50 - Unit 1
- 1-0 Pseudocode Teacher Notes
  - Pseudocode Activity
- 1-1 App Inventor Teacher Notes
- 1-2 Syntax Teacher Notes
  - Syntax Error Worksheet
- 1-3 Variables Teacher Notes
- 1-4 Data Types Teacher Notes
- 1-5 Operators Teacher Notes
- 1-6 Boolean Expressions Teacher Notes
  - Boolean Algebra Worksheet
- 1-7 Loops Teacher Notes
- AP Exam Reference Sheet
- CS50 Reference Sheet

# Verona High School AP Computer Science Principles

| Unit 2: Putting the Blocks Together | Unit Duration: 4 - 5 Weeks |
|---|---|

## Stage 1: Desired Results

### Established Goals:

### AP CSP Learning Objectives:

LO 1.1.1 Apply a creative development process when creating computational artifacts. [P2]
LO 1.2.2 Create a computational artifact using computing tools and techniques to solve a problem. [P2]
LO 1.2.3 Create a new computational artifact by combining or modifying existing artifacts. [P2]
LO 2.1.1 Describe the variety of abstractions used to represent data. [P3]
LO 2.1.2 Explain how binary sequences are used to represent digital data. [P5]
LO 2.2.2 Use multiple levels of abstraction to write programs. [P3]
LO 2.2.3 Identify multiple levels of abstractions that are used when writing programs. [P3]
LO 4.1.1 Develop an algorithm for implementation in a program. [P2]
LO 4.1.2 Express an algorithm in a language. [P5]
LO 5.1.1 Develop a program for creative expression, to satisfy personal curiosity, or to create new knowledge. [P2]
LO 5.1.2 Develop a correct program to solve problems. [P2]
LO 5.3.1 Use abstraction to manage complexity in programs. [P3]
LO 5.4.1 Evaluate the correctness of a program. [P4]
LO 5.5.1 Employ appropriate mathematical and logical concepts in programming. [P1]
LO 7.1.1 Explain how computing innovations affect communication, interaction, and cognition. [P4]
LO 7.2.1 Explain how computing has impacted innovations in other fields. [P1]

### AP CSP Curricular Requirements:

CR1a - Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P1: Connecting Computing.
CR1b - Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P2: Creating Computational Artifacts.
CR1c - Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P3: Abstracting.
CR1d - Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P4: Analyzing Problems and Artifacts.
CR1e - Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P5: Communicating.
CR2a - Students are provided with opportunities to meet learning objectives within Big Idea 1: Creativity. Such opportunities must occur in addition to the AP Computer Science Principles Performance Tasks.
CR2b - Students are provided with opportunities to meet learning objectives within Big Idea 2: Abstraction. Such opportunities must occur in addition to the AP Computer Science Principles Performance Tasks.
CR2d - Students are provided with opportunities to meet learning objectives within Big Idea 4: Algorithms. Such opportunities must occur in addition to the AP Computer Science Principles Performance Tasks.
CR2e - Students are provided with opportunities to meet learning objectives within Big Idea 5: Programming. Such opportunities must occur in addition to the AP Computer Science Principles Performance Tasks.
CR2g - Students are provided with opportunities to meet learning objectives within Big Idea 7: Global Impact. Such opportunities must occur in addition to the AP Computer Science Principles Performance Tasks.

### Transfer Goal:

Students will be able to <u>independently</u> use their learning to… create more complex code and continuously make it better.

### Students will understand that:

- Compiling correctly is essential to programming.
- Functions are an amazing tool that we can use to organize, simplify, and reuse the code that we write.
- It is easier to debug 10 lines of code rather that 100 or 1000.
- Arrays are a fundamental data structure that hold the same value type at contiguous memory locations.
- Arrays are the reason C is instrumental to this course due to the ability to access memory locations.
- Information input at the command line can have a program take different paths depending on what is input.
- Exit codes tell the computer what to expect next, or what to do next.
- Libraries are how we as programmers share important information and work with one another.
- There are times what the defined type of a variable should be changed.
- A bug is an error in our code, logical or otherwise, such that the behavior is not quite what we expect, that we allow certain inputs from users that cause our program to fail, or even worse that our program suffers a segmentation fault or otherwise crashes every single time we run it.
- The introductory programming in this course will not be what is done in modern jobs.
- Templates and frameworks that are built upon by people in industry are the most common coding jobs today.

### Essential Questions:

- What does “make” do?
- How is code understood by the computer?
- What is the optimal way to write code?
- When would it be useful to use an array?
- What is the benefit of command line interaction?
- Why would we need different word/codes to tell us how to exit a building or in our case, a function?
- Is it appropriate to use code already written?
- What are the benefits and shortcomings of using someone else’s code? (open source)
- What are the benefits and shortcomings of typecasting?
- What are the good and bad qualities of bugs?

### Students will know:

- The purpose of the make command.
- What the make command does behind the scenes.
- The process of passing information within a program.
- Why we count from 0 instead of 1.
- The syntax of arrays and strings.
- The definitions of argv and argc and their usage.
- The difference between exit codes.
- The difference between a .c and a .h file.
- Variables can be converted to difference types that have the same or less precision.
- Typecasting is temporary.
- How to use tools to spot and fix bugs in a program using CS50 IDE.
- The pros and cons of filling out existing code frameworks.

### Students will be able to:

- Use clang.
- Write and use functions.
- Create and manipulate arrays and strings.
- Create and manipulate programs with command line interaction.
- Understand what specific exit codes mean in upcoming programs.
- Use libraries outside the realm of this course to advance their own programming abilities.
- Change variable types to suit the needs of the program.
- See real-world implications of “buggy” code.
- Purposefully debug and bug code.
- Read distribution code and understand its meaning and what is needed in order to solve a problem.

# Verona High School AP Computer Science Principles

## Stage 2: Acceptable Evidence

### Performance Task & Unit Assessments:

**Assessments -**

- Unit 2 Pop Quizzes
- Unit 2 Exam
  - Unit 2 Exam - Student Answer Sheet

**Transfer Tasks:**

- Writing Problem 2-0: Me, Myself and UI
  Compare and contrast different technologies that are relevant now and research the way they came to be what they are today. Then predict what society will see as the next generation of these technologies.
  Teacher Notes

- Problem 2-1: Reverse Engineer
  GIven output and a different compiler, create code that generates this output.
  Teacher Notes

- Problem 2-2: Initials
  Using outside sources to research commands,, students will get input for a user and manipulate the specific letters of an array/string and generate output.
  Teacher Notes

- Problem 2-3: Old Friends
  Students will modify old programs to use new topics like command line arguments and exit codes.
  Teacher Notes

- Problem 2-4: Calc 1.0
  Create a command line calculator that require in depth use of arrays.
  Teacher Notes

- Problem 2-5: Caesar
  Write a simple Caesar cipher as an introduction to cryptography.
  Teacher Notes

- Problem 2-6: Vigenere
  Write a more complex cryptographic program that uses arrays, keys, functions and more.
  Teacher Notes

- Problem 2-7: Crack
  Write a hacker program that decodes a list of passwords encoded with an unknown cipher.
  Teacher Notes

### Other Evidence:
### Students will show that they have achieved Stage 1 goals by:

Formal:
- Providing written/oral response to the EQs
- Passing all quizzes on basic concepts in unit.
- Receiving passing grades on Transfer Tasks

Informal:
- Understanding Source/Assembly Code
- Using cyphers
- Exclusively using Command line
- Students should also troubleshoot examples and explain misapplications of the conventions or properties.

## Reference Materials

- Verona AP CS50 - Unit 2
- 2-0 Compiling Teacher Notes
- 2-1 Functions Teacher Notes
- 2-2 Arrays and Strings Teacher Notes
- 2-3 Command Line Teacher Notes
- 2-4 Exit Codes Teacher Notes
- 2-5 Libraries Teacher Notes
  - Libraries Activity
- 2-6 Typecasting Teacher Notes
- 2-7 Bugs and Debugging Teacher Notes
  - Debugging Problem Set

- CS50 Reference Sheet 2

# Verona High School AP Computer Science Principles

| Unit 3: Thinking Computationally | Unit Duration: 4 weeks |
|---|---|

## Stage 1: Desired Results

### Established Goals:

### AP CSP Learning Objectives:

LO 1.1.1 Apply a creative development process when creating computational artifacts.
LO 1.2.3 Create a new computational artifact by combining or modifying existing artifacts.
LO 1.2.4 Collaborate in the creation of computational artifacts.
LO 2.2.1 Develop an abstraction when writing a program or creating other computational artifacts.
LO 2.2.2 Use multiple levels of abstraction to write programs.
LO 2.2.3 Identify multiple levels of abstractions that are used when writing programs.
LO 3.1.1 Use computers to process information, find patterns, and test hypotheses about digitally processed information to gain insight and knowledge.
LO 3.1.2 Collaborate when processing information to gain insight and knowledge.
LO 3.1.3 Explain the insight and knowledge gained from digitally processed data by using appropriate visualizations, notations, and precise language.
LO 3.2.2 Use large data sets to explore and discover information and knowledge.
LO 4.1.1 Develop an algorithm for implementation in a program.
LO 4.1.2 Express an algorithm in a language.
LO 4.2.1 Explain the difference between algorithms that run in a reasonable time and those that do not run in a reasonable time.
LO 5.1.2 Develop a correct program to solve problems.
LO 5.1.3 Collaborate to develop a program.
LO 5.2.1 Explain how programs implement algorithms.
LO 5.3.1 Use abstraction to manage complexity in programs.
LO 5.4.1 Evaluate the correctness of a program.
LO 5.5.1 Employ appropriate mathematical and logical concepts in programming.
LO 7.1.1 Explain how computing innovations affect communication, interaction, and cognition.

### AP CSP Curricular Requirements:

CR1a – Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P1: Connecting Computing.
CR1b – Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P2: Creating Computational Artifacts.
CR1c – Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P3: Abstracting.
CR1d – Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P4: Analyzing Problems and Artifacts.
CR1e – Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P5: Communicating.
CR1f – Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P6: Collaborating.
CR2a – Students are provided with opportunities to meet learning objectives within Big Idea 1: Creativity. Such opportunities must occur in addition to the AP Computer Science Principles Performance Tasks.
CR2b – Students are provided with opportunities to meet learning objectives within Big Idea 2: Abstraction. Such opportunities must occur in addition to the AP Computer Science Principles Performance Tasks.
CR2c – Students are provided with opportunities to meet learning objectives within Big Idea 3: Data and Information. Such opportunities must occur in addition to the AP Computer Science Principles Performance Tasks.
CR2d – Students are provided with opportunities to meet learning objectives within Big Idea 4: Algorithms. Such opportunities must occur in addition to the AP Computer Science Principles Performance Tasks.
CR2e – Students are provided with opportunities to meet learning objectives within Big Idea 5: Programming. Such opportunities must occur in addition to the AP Computer Science Principles Performance Tasks.
CR2g – Students are provided with opportunities to meet learning objectives within Big Idea 7: Global Impact. Such opportunities must occur in addition to the AP Computer Science Principles Performance Tasks.

### Transfer Goal:

Students will be able to <u>independently</u> use their learning to… think like a computer, analyzing differences in algorithms and their merits and disadvantages while doing so with refine computational vocabulary.

### Students will understand that:

- Having sorted data makes analysis and changes much easier.
- Algorithms cost memory usage and CPU cycles.
- Bubble sort compares two subsequent terms at a time and swaps their order, if necessary, through the array, however many times it will take to put array in order.
- Selection sort looks for the smallest value and swaps it with the appropriate place in the array on each pass.
- Insertion sort sorts the specific element in the proper place at each iteration, shifting when necessary.
- Linear search is the most basic algorithm for searching that students will encounter in the course.
- Binary search eliminates half the problem at every iteration.
- Mergesort uses recursion to sort arrays.
- It is important to analyze how long a program takes to run to minimize processing time and RAM usage.
- Computers, though very “smart” cannot do EVERYTHING and will never be able to do EVERYTHING.

### Essential Questions:

- Is order important?
- Is the way we order data important?
- What makes a program “cost” a lot?
- How can we define cost time in terms of mathematics and variables?
- How can an algorithm be O(log n)?
- Is the fastest always the best?
- How is time measured?
- Can computers do anything?
- What can computers NOT do?
- Will there be a time in the future where computers will be able to do anything?
- Can computers work WITHOUT humans?

### Students will know:

- The different sorting algorithms and their costs.
- Bubble sort is inefficient.
- Bubble sort bubbles higher numbers to the end of the array.
- Selection sort has the same cost as bubble sort in the worst case scenario.
- Insertion Sort has the same cost as selection and bubble sort in the worst case scenario.

### Students will be able to:

- Write code that sorts in four different ways.
- Use the sorted data to extract information.
- Analyze how many comparisons and how many swaps are made in each type of sort in best case and worst case scenarios.
- Search for a value linearly.

| | |
|---|---|
| ● Linear cost is the worst possibility, with respect to memory and CPU consumption.<br>● In order to use binary search, the array must be sorted.<br>● Big O is worst case scenario.<br>● Big Ω is best case scenario.<br>● The difference between algorithms that run in a reasonable time and those that do not run in a reasonable time.<br>● There are some problems a computer is logically incapable of solving. | ● Analyze algorithms to find their best case and worst case timing.<br>● Recursively analyze previous programs.<br>● Understand the mathematics behind asymptotic notation.<br>● Understand the implications of the halting problem. |

## Stage 2: Acceptable Evidence

### Performance Task & Unit Assessments:

**Assessments -**

- Unit 3 Pop Quiz
- Unit 3 Cumulative Quiz
- Unit 3 Exam

**Transfer Tasks:**

- Problem 3-0: RNG
  Create a random number generator to be used later in the unit that adds to the specifications learned about before.
  Teacher Notes

- Problem 3-1: Fifteen (Part 1)
  Using distributing code, initialize and draw the game board for the game of 15.
  Teacher Notes

- Problem 3-2: Fifteen (Part 2)
  Add functions move and won to the previous program to play a user-friendly, electronic game of 15.
  Teacher Notes

- Problem 3-3: Fifteen (Part 3)
  Add a RNG to the previous program to have the board start in a pseudorandom, solvable order where the function: init, draw, move and won all play the game properly.
  Teacher Notes

- Problem 3-4: Sort Race
  Using distribution code and helper files, modify to your specifications to analyze the time it takes to run algorithms on different types of arrays.
  Teacher Notes

- Writing Problem 3-5: Analyze This
  In 500-1,000 words, prepare an essay on your experience, expectations, growths, triumphs, and struggles up through this point in the course
  Teacher Notes

- Problem 3-6: Seek
  Using distribution helper files, create search and sorting algorithms.
  Teacher Notes

- Problem 3-7: Scramble (Part 1)
  After playing the Scramble game, finish the distribution code to draw the board and lookup words in the dictionary.
  Teacher Notes

- Problem 3-8: Breakout
  Use distribution code and a new GUI to implement the Bricks game.
  Teacher Notes

### Other Evidence:
### Students will show that they have achieved Stage 1 goals by:

Formal:
- Providing written/oral response to the EQs
- Passing all quizzes on basic concepts in unit.
- Receiving passing grades on Transfer Tasks

Informal:
- Understanding time complexity with respect to sorting algorithms
- Choosing their favorite sorting algorithm
- Migrating knowledge to JavaScript
- Being able to adapt to adding to other's code

## Reference Materials

- Verona AP CS50 - Unit 3
- 3-0 Linear Search Teacher Notes
- 3-1 Bubble Sort Teacher Notes
- 3-2 Selection Sort Teacher Notes
- 3-3 Insertion Sort Teacher Notes
- 3-4 Binary Search Teacher Notes
- 3-5 Time Complexity Teacher Notes
- 3-6 Unsolvable Problems Teacher Notes
- 3-7 Simulation Teacher Notes

# Verona High School AP Computer Science Principles

| Unit 4: Design, Elegance, and Efficiency | Unit Duration: 4 weeks |
|---|---|

## Stage 1: Desired Results

### Established Goals:

### AP CSP Learning Objectives:

LO 1.1.1 Apply a creative development process when creating computational artifacts.
LO 1.2.1 Create a computational artifact for creative expression.
LO 1.2.3 Create a new computational artifact by combining or modifying existing artifacts.
LO 1.2.4 Collaborate in the creation of computational artifacts.
LO 2.2.1 Develop an abstraction when writing a program or creating other computational artifacts.
LO 2.2.2 Use multiple levels of abstraction to write programs.
LO 2.2.3 Identify multiple levels of abstractions that are used when writing programs.
LO 3.1.2 Collaborate when processing information to gain insight and knowledge.
LO 3.1.3 Explain the insight and knowledge gained from digitally processed data by using appropriate visualizations, notations, and precise language.
LO 3.2.1 Extract information from data to discover and explain connections, patterns, or trends.
LO 3.2.2 Use large data sets to explore and discover information and knowledge.
LO 3.3.1 Analyze how data representation, storage, security, and transmission of data involve computational manipulation of information.
LO 4.1.1 Develop an algorithm for implementation in a program.
LO 4.1.2 Express an algorithm in a language.
LO 4.2.4 Evaluate algorithms analytically and empirically for efficiency, correctness, and clarity.
LO 5.1.1 Develop a program for creative expression, to satisfy personal curiosity, or to create new knowledge.
LO 5.1.2 Develop a correct program to solve problems.
LO 5.1.3 Collaborate to develop a program.
LO 5.3.1 Use abstraction to manage complexity in programs.
LO 5.4.1 Evaluate the correctness of a program.
LO 5.5.1 Employ appropriate mathematical and logical concepts in programming.

### AP CSP Curricular Requirements:

CR1a – Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P1: Connecting Computing.
CR1b – Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P2: Creating Computational Artifacts.
CR1c – Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P3: Abstracting.
CR1d – Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P4: Analyzing Problems and Artifacts.
CR1e – Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P5: Communicating.
CR1f – Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P6: Collaborating.
CR2a – Students are provided with opportunities to meet learning objectives within Big Idea 1: Creativity. Such opportunities must occur in addition to the AP Computer Science Principles Performance Tasks.
CR2b – Students are provided with opportunities to meet learning objectives within Big Idea 2: Abstraction. Such opportunities must occur in addition to the AP Computer Science Principles Performance Tasks.
CR2c– Students are provided with opportunities to meet learning objectives within Big Idea 3: Data and Information. Such opportunities must occur in addition to the AP Computer Science Principles Performance Tasks.
CR2d – Students are provided with opportunities to meet learning objectives within Big Idea 4: Algorithms. Such opportunities must occur in addition to the AP Computer Science Principles Performance Tasks.
CR2e – Students are provided with opportunities to meet learning objectives within Big Idea 5: Programming. Such opportunities must occur in addition to the AP Computer Science Principles Performance Tasks.

### Transfer Goal:

Students will be able to <u>independently</u> use their learning to… consider not just right and wrong, but how and why, discussing design trade-offs and why sometimes indeed the discipline of computer science boils down to determining what trade-off is appropriate under a given set of circumstances.

### Students will understand that:

- Good design is an important aspect of programming and product development in general.
- Good design is what differentiates between a program that works and a program that works well.
- We can create objects that store multiple variable of different data types.
- In a recursive solution, a function (or set of functions) repeatedly invokes slightly modified instance of itself, with each subsequent instance tending closer and closer to a base case.
- Merge sort leverages recursion to "pass the buck" of sorting.
- To represent values in hex and how to convert between hex, binary, and decimal using basic algorithms.
- C has the capability of file I/O as a means of storing persistent data that exists after our programs have finished running and to read information from a file during the course of the program's execution.
- Each type of image file has its advantages and disadvantages, limitations, and more.
- Collaboration is so important in the field of computer science; rarely if ever does a programmer find herself on an island when working on a project.

### Essential Questions:

- What is good design?
- Why is good design important?
- How are arrays inefficient?
- What is the optimal way to write code?
- What's the most significant trade off with merge sort?
- Why do we have different number systems? Isn't decimal enough?
- When might it be useful for our programs to have persistent data?
- Why do different types of the same things exist?
- What is the best way to collaborate?

### Students will know:

- Good design includes, but is not limited to, a program that has a fast run-time, is modularized for easy debugging, robust, consistent, and not repetitive in code.
- ncurses is a simple graphics library that can be used to create rudimentary graphical user interfaces for games.
- The situations in programming when structures are useful.
- Recursive procedures, when contrasted with iterative ones, can sometimes lead to incredibly efficient, elegant, and beautiful solutions.

### Students will be able to:

- Create programs that are portable, scalable, and reusable.
- Discuss ncurses and some of the features existing therein.
- Create, analyze and use structures.
- Be impressed and amazed at the simplicity of these elegant solutions.
- Successfully use recursion and mergesort.
- Convert between basic number systems used by computers.
- Some powerful I/O programs.

| | |
|---|---|
| ● Mergesort accomplishes O(n log n) run time.<br>● Hexadecimal numbers provide a shorthand mechanism for binary numbers.<br>● The basic file I/O functions they have at their disposal through stdio.h<br>● Images are stored as a number of file types, including but not limited to bitmaps (.bmp), JPGs (.jpg), PNGs (.png), and GIFs (.gif).<br>● More often than not, a programmer will have peers with whom she is collaborating or sharing when designing something, and all parties will need to share information | ● Converse accurately about images and what we are capable of doing with them.<br>● Collaborate effectively and ethically. |

# Verona High School AP Computer Science Principles

## Stage 2: Acceptable Evidence

### Performance Task & Unit Assessments:

**Assessments -**

- 

**Transfer Tasks:**

- Problem 4-0: Sudoku (Part 1)
  Using the new GUI, ncurses, analyze the components that are necessary for a GUI program and write code for those components.
  Teacher Notes

- Problem 4-1: Sudoku (Part 2)
  Using student or staff implementation of distribution code, program the sudoku board to work correctly.
  Teacher Notes

- Problem 4-2: Scramble (Part 2)
  Extend on the Scramble program to implement a Help feature and variable scoring.
  Teacher Notes

- Problem 4-3: Finder
  Students will implement a finder program that will search for strings and output all files that contain the string in an output file.
  Teacher Notes

- Problem 4-4: Whodunit
  Use the characteristics of bitmap images to uncover the identity of a blurred image.
  Teacher Notes

- Problem 4-5: Scale
  Resize bitmap images while keeping all of the attributes.
  Teacher Notes

- Problem 4-6: Shrink
  Expand on the Scale program to shrink a bitmap by a user defined factor.
  Teacher Notes

- Problem 4-7: Recover
  Knowing information about image file headers, create a program that can recover data that has been corrupted.
  Teacher Notes

**Performance Task 1: Explore – Impact of Computing Innovations** (Completion Time: 2 weeks/8 hours of in class time)
16% of overall AP Score
Computing innovations impact our lives in ways that require considerable study and re ection for us to fully understand them. In this performance task, students will explore a computing innovation of their choice. The close examination of a computing innovation will deepen the students' understanding of computer science principles.

In this task, students select and investigate a computing innovation that has had or has the potential to have signi cant bene cial and harmful effects on society, economy, or culture; consumes,
produces, and/or transforms data; and raises at least one data storage concern, data privacy concern, or data security concern. Students will produce a computational artifact that illustrates,
represents, or explains the computing innovation's intended purpose, its function, or its effect, and provide written responses to each of the given prompts.

There are a number of widely available computational tools students can use to create computational artifacts for this task. A computational artifact is a visualization, a graphic, a video, a program, or an audio recording that students create using a computer. The students' creations could solve a problem, show creative expression, or provide the viewer with new insight or knowledge.

E***ffective artifacts include:***

▸ visual, graphical, and/or audio content to help a reader understand the purpose of a computing innovation; and

▸ the use of communications media, such as animations, comic strips, infographics, and/ or public service announcements, to illustrate the purpose of a computing innovation.

***Ineffective artifacts include:***

▸ artifacts that repeat information supplied in the written responses;

### Other Evidence:
### Students will show that they have achieved Stage 1 goals by:

Formal:
- Providing written/oral response to the EQs
- Passing all quizzes on basic concepts in unit.
- Receiving passing grades on Transfer Tasks

Informal:
- Creating and using structures
- Seamlessly starting to use upper level programming languages
- Integrating files into C programs
- Understanding the functionality of the C programming language with respect to Images and other files.

- ▸ multi-slide presentations with paragraphs of text or bullets; and
- ▸ artifacts that have not been created by the student.

Students will provide written responses to a series of prompts. Their responses must provide evidence of the extensive knowledge they have developed about their chosen computing innovation and its impact(s). Their responses should be understandable to someone who is unfamiliar with the computing innovation.

Written responses must be based on relevant, credible, and easily accessible sources. Students are required to cite at least three sources that helped them create their computational artifact and/or formulate their written responses. At least two of the sources must be available online or in print; the third source may be either online, in print, or a personal interview with an expert on the computing innovation. At least two of the sources must have been created after the end of the previous academic year. Students must avoid plagiarism by acknowledging, attributing, and/or citing sources throughout their responses and including a bibliography. Sources that should be cited include text, images, graphs, and program code that are used in the creation of their computational artifacts.

## Reference Materials

- Verona AP CS50 - Unit 4
- ncurses Lab
- 4-0 Principles of Good Design Teacher Notes
  - Principles of Good Design Writing Assignment
- 4-1 ncurses Teacher Notes
- 4-2 Structures and Encapsulation Teacher Notes
- 4-3 Recursion Teacher Notes
  - Recursion Activity
- 4-4 Merge Sort Teacher Notes
- 4-5 Hexadecimal Teacher Notes
- 4-6 File I/O Teacher Notes
- 4-7 Images Teacher Notes
- 4-8 Version Control Teacher Notes

# Verona High School AP Computer Science Principles

| Unit 5: Networking and the Internet | Unit Duration: 3 weeks |
|---|---|

## Stage 1: Desired Results

### Established Goals:

### AP CSP Learning Objectives:

LO 1.1.1 Apply a creative development process when creating computational artifacts.
LO 1.2.1 Create a computational artifact for creative expression.
LO 1.2.2 Create a computational artifact using computing tools and techniques to solve a problem.
LO 1.2.3 Create a new computational artifact by combining or modifying existing artifacts.
LO 1.2.4 Collaborate in the creation of computational artifacts.
LO 1.3.1 Use computing tools and techniques for creative expression.
LO 5.1.1 Develop a program for creative expression, to satisfy personal curiosity, or to create new knowledge.
LO 5.1.3 Collaborate to develop a program.
LO 5.3.1 Use abstraction to manage complexity in programs.
LO 6.1.1 Explain the abstractions in the Internet and how the Internet functions.
LO 6.2.1 Explain characteristics of the Internet and the systems built on it.
LO 6.2.2 Explain how the characteristics of the Internet influence the systems built on it.
LO 6.3.1 Identify existing cybersecurity concerns and potential options to address these issues with the Internet and the systems built on it.
LO 7.1.1 Explain how computing innovations affect communication, interaction, and cognition.
LO 7.1.2 Explain how people participate in a problem-solving process that scales.
LO 7.2.1 Explain how computing has impacted innovations in other fields.
LO 7.3.1 Analyze the beneficial and harmful effects of computing.
LO 7.4.1 Explain the connections between computing and economic, social, and cultural contexts.

### AP CSP Curricular Requirements:

CR1a – Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P1: Connecting Computing.
CR1b – Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P2: Creating Computational Artifacts.
CR1c – Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P3: Abstracting.
CR1d – Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P4: Analyzing Problems and Artifacts.
CR1e – Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P5: Communicating.
CR1f – Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P6: Collaborating.
CR2a – Students are provided with opportunities to meet learning objectives within Big Idea 1: Creativity. Such opportunities must occur in addition to the AP Computer Science Principles Performance Tasks.
CR2e – Students are provided with opportunities to meet learning objectives within Big Idea 5: Programming. Such opportunities must occur in addition to the AP Computer Science Principles Performance Tasks.
CR2f – Students are provided with opportunities to meet learning objectives within Big Idea 6: The Internet. Such opportunities must occur in addition to the AP Computer Science Principles Performance Tasks.
CR2g – Students are provided with opportunities to meet learning objectives within Big Idea 7: Global Impact. Such opportunities must occur in addition to the AP Computer Science Principles Performance Tasks.

### Transfer Goal:

Students will be able to <u>independently</u> use their learning to… build simple web pages and broadcast them to the world.

### Students will understand that:

- The Internet is a set of rules that defines how networks communicate through different types of protocols.
- A device or web page can be identified by its IP addresses, as all are unique - like a postal address.
- Routers are the components of the Internet that direct packages of data across various networks.
- TCP is a system of "guaranteed" delivery, wherein the destination system knows if not all packets of data have arrived successfully at the destination.
- There are two major types of HTTP requests used by browsers: GET and POST. The server receives the request and either successfully executes the action (by rendering a page or submitting a form, for example) or returns an error code.
- Every open source piece of software is held to an "open standard" that the software will work in the way intended and not do anything malicious, though the standard is not owned by any particular group of people.
- The Internet is a wonderful resource, but can also occasionally make us vulnerable if we are not following good practices.
- Even though we can protect ourselves from many cybersecurity threats by being wary and creating strong passwords, there are many security threats we cannot control and may not even be aware of.
- It is not actually a programming language (hence the markup language title), but it is used to make any web page you see by formatting all text and images.
- CSS is capable of manipulating colors, positioning, size, alignment, fonts, borders, background shading, and others.

### Essential Questions:

- How does the Internet work?
- Can we run out of IP addresses?
- What happens when two things have the same address?
- What does a router do?
- Why do you think data will seemingly travel farther away than is necessary only to return back to a desired location?
- Can webpages talk? What language do they speak?
- Why do we trust specific programs? What makes us believe that they're safe?
- How can we try to protect ourselves from the most blatant internet security problems?
- How can we as users defend against cyber attacks?
- What are the assets and limitations of HTML?
- What does CSS allow us to do?

### Students will know:

- The intricacies of IP addresses, access points, switches, routers, the server/client paradigm, TCP/IP, HTTP, and cybersecurity.
- IP (Internet Protocol) addresses are assigned to each device connected to a network.
- Routers follow a set of rules to direct packets based upon IP address and port.

### Students will be able to:

- Converse competently and accurately about the Internet and how it works.
- Distinguish between public and private IP addresses.
- Communicate with other computers via IP address.
- Rationalize why routers are necessary in home networks.
- TCP is also responsible for separating data by type and sending them via separate ports.

| | |
|---|---|
| ● TCP ensures data is properly marked when it is split into pieces, so if one packet does not arrive, the sender is notified and can resend.<br>● Hypertext Transfer Protocol, or HTTP, is what web browsers use to speak to web servers.<br>● The vast majority of the software that you run on your computer every day came from an open source.<br>● Having a safe password, keeping track of file permissions, and only visiting safe websites are just a few examples of ways to protect yourself on the web.<br>● The most common types of cyber attacks that take place on the Internet today.<br>● HTML, or HyperText Markup Language, forms the backbone of web pages.<br>● CSS, Cascading Style Sheets, is used to style web pages. | ● Make informed decisions about trusting websites and software.<br>● See how easy it is for non-secure websites to extract personal information from users.<br>● Rationalize recent cyberattacks and how they happened.<br>● Code webpages using HTML.<br>● Incorporate CSS into HTML code. |

## Stage 2: Acceptable Evidence

### Performance Task & Unit Assessments:

**Assessments -**

- Unit 5 Exam

**Transfer Tasks:**

- Writing Problem 7-0: Be the Teacher
  The technologies of the internet can be complex, so students are challenged to explain things concisely to a lay audience, cementing their understanding of these technologies by having to discuss them more casually.
  Teacher Notes

- Writing Problem 7-1: Defender of the Web
  Students explore the notions of cyberattacks and cyber security, and investigate in more detail some of the common types of attacks that impact websites today.
  Teacher Notes

- Problem 7-2: </Unit 7>
  Students create their own web pages, learn about permissions schemes, and make their creations accessible to the world.
  Teacher Notes

### Other Evidence:
### Students will show that they have achieved Stage 1 goals by:

Formal:
- Providing written/oral response to the EQs
- Passing all quizzes on basic concepts in unit.
- Receiving passing grades on Transfer Tasks

Informal:
- Create web pages
- Explain how the internet works in non-technical terms
- Thoroughly discuss computer/internet security issues.

## Reference Materials

- Verona AP CS50 - Unit 5
- 5-0 Internet Basics Teacher Notes
- 5-1 IP Addresses Teacher Notes
  - IP Address Activity
- 5-2 DCHP and DNS TEacher Notes
- 5-3 Routers Teacher Notes
- 5-4 TCP and IP Teacher Notes
- 5-5 HTTP Teacher Notes
  - Request/Status Code Activity
- 5-6 Trust Models Teacher Notes
  - Trust Model Writing Assignment
- 5-7 Cybersecurity Teacher Notes
- 5-8 and 5-9 HTML and CSS Teacher Notes

# Verona High School AP Computer Science Principles

| Unit 6: Problem Solving in an Interconnected World | Unit Duration: 5 weeks |
|---|---|

## Stage 1: Desired Results

### Established Goals:

### AP CSP Learning Objectives:

LO 1.1.1 Apply a creative development process when creating computational artifacts.
LO 1.2.1 Create a computational artifact for creative expression.
LO 1.2.2 Create a computational artifact using computing tools and techniques to solve a problem.
LO 1.2.3 Create a new computational artifact by combining or modifying existing artifacts.
LO 1.2.4 Collaborate in the creation of computational artifacts.
LO 1.3.1 Use computing tools and techniques for creative expression.
LO 2.1.1 Describe the variety of abstractions used to represent data.
LO 2.2.2 Use multiple levels of abstraction to write programs.
LO 2.2.3 Identify multiple levels of abstractions that are used when writing programs.
LO 3.1.2 Collaborate when processing information to gain insight and knowledge.
LO 3.1.3 Explain the insight and knowledge gained from digitally processed data by using appropriate visualizations, notations, and precise language.
LO 3.2.1 Extract information from data to discover and explain connections, patterns, or trends.
LO 3.2.2 Use large data sets to explore and discover information and knowledge.
LO 3.3.1 Analyze how data representation, storage, security, and transmission of data involve computational manipulation of information.
LO 4.1.1 Develop an algorithm for implementation in a program.
LO 4.1.2 Express an algorithm in a language.
LO 5.1.1 Develop a program for creative expression, to satisfy personal curiosity, or to create new knowledge.
LO 5.1.3 Collaborate to develop a program.
LO 5.3.1 Use abstraction to manage complexity in programs.
LO 5.5.1 Employ appropriate mathematical and logical concepts in programming.
LO 6.1.1 Explain the abstractions in the Internet and how the Internet functions.
LO 6.2.2 Explain how the characteristics of the Internet influence the systems built on it.
LO 6.3.1 Identify existing cybersecurity concerns and potential options to address these issues with the Internet and the systems built on it.
LO 7.3.1 Analyze the beneficial and harmful effects of computing.
LO 7.4.1 Explain the connections between computing and economic, social, and cultural contexts.

### AP CSP Curricular Requirements:

CR1a – Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P1: Connecting Computing.
CR1b – Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P2: Creating Computational Artifacts.
CR1c – Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P3: Abstracting.
CR1d – Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P4: Analyzing Problems and Artifacts.
CR1e – Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P5: Communicating.
CR1f – Students are provided with opportunities to meet learning objectives connected to Computational Thinking Practice P6: Collaborating.
CR2a – Students are provided with opportunities to meet learning objectives within Big Idea 1: Creativity. Such opportunities must occur in addition to the AP Computer Science Principles Performance Tasks.
CR2b – Students are provided with opportunities to meet learning objectives within Big Idea 2: Abstraction. Such opportunities must occur in addition to the AP Computer Science Principles Performance Tasks.
CR2c – Students are provided with opportunities to meet learning objectives within Big Idea 3: Data and Information. Such opportunities must occur in addition to the AP Computer Science Principles Performance Tasks.
CR2d – Students are provided with opportunities to meet learning objectives within Big Idea 4: Algorithms. Such opportunities must occur in addition to the AP Computer Science Principles Performance Tasks.
CR2e – Students are provided with opportunities to meet learning objectives within Big Idea 5: Programming. Such opportunities must occur in addition to the AP Computer Science Principles Performance Tasks.
CR2f – Students are provided with opportunities to meet learning objectives within Big Idea 6: The Internet. Such opportunities must occur in addition to the AP Computer Science Principles Performance Tasks.
CR2g – Students are provided with opportunities to meet learning objectives within Big Idea 7: Global Impact. Such opportunities must occur in addition to the AP Computer Science Principles Performance Tasks.

### Transfer Goal:

Students will be able to <u>independently</u> use their learning to… solve more complex problems that require processing large amounts of data and dealing with processes that scale, and see how these techniques can be applied to confront the challenges computer scientists will be contending with in the future.

### Students will understand that:

- PHP is an interpreted language--it is reduced to binary at runtime, not ahead of time--which means that it can suffer some performance degradation when run alongside similar programs in C.
- In addition to PHP's power as a language operating on the command line, that power is only multiplied when we leverage what PHP was originally designed to do -- permit the implementation of information exchange via the web.
- Structured Query Language is a language that we can use to create, add to, select, modify, and delete information in connection with a database.
- MVC, or the Model-View-Controller paradigm, is a software design practice that separates code for a website into three parts: the model code, the view code, and the controller code.

### Essential Questions:

- What is the point of studying C for so long if we now take the time to learn PHP?
- Why are there so many programming languages?
- What is the point of studying C for so long if we now take the time to learn PHP?
- What are some of the advantages of SQL? What are some of the disadvantages?
- Why is MVC is a useful tool for developing websites?
- What languages might be necessary in order to use MVC effectively?
- Why do so many programming languages look similar?
- What common websites seem to be using Ajax, and how can you tell?
- How should ethical dilemmas be handled with artificial intelligence?
- Why do humans playing board games not have to consider as many possibilities as AIs do? How do they limit which possibilities they consider?

| | |
|---|---|
| ● PHP gave us the opportunity to build dynamic websites, but JavaScript lets us take things even further, permitting client-side modifications to our web pages, making them more dynamic without requiring contact with a database or off-site server, which can greatly improve the user experience.<br>● Ajax (which formerly stood for Asynchronous JavaScript and XML) is a web programming technique that lets us dynamically update the content of the web page.<br>● Artificial intelligence is an example of how Computer Science concepts can be used to create global impact.<br>● Though many people believe that computers can solve all problems, as we've seen in Unit 3, Module 7: Unsolvable problems, not all problems are solvable by computers, e.g. the halting problem. | ● What other modern-day computing problems can you think of?<br>● What problems would you think that computers can solve but in reality can't? |
| **Students will know:**<br><br>● PHP, like many modern programming languages, counts C as an ancestor and so it is syntactically extremely similar; this should make the transition to using PHP instead of C a little bit easier.<br>● Using PHP's web programming powers, students will quickly be able to implement a multi-layered website with a login, forms, and more.<br>● PHP is not compiled due to step constraints, but will not give error code feedback and takes longer.<br>● By learning how to use SQL we can create a so-called "backend" for our websites, allowing us to store persistent user data that we can access when necessary to improve the user experience on a page.<br>● C is used to learn the fundamentals which translate into more recent, usable languages.<br>● MVC deals with the data of the website, often performing operations that interact with a database.<br>● The controller code is the logic of the website: it contains code like loops and conditions. The view code is the aesthetic part of the website: it takes information provided by the controller and displays the page that the user ultimately sees when they visit the website.<br>● Most websites use MySQL.<br>● The basic power of JavaScript and the Document Object Model (DOM), which JavaScript is able to manipulate to literally change the contents of a website, albeit temporarily, in response to user interactions.<br>● We can make snappier web pages that do not reload every time additional content is requested.<br>● CS50's artificial intelligence material covers natural language processing, speech recognition, and game playing systems.<br>● Even in modern times, there still exists many computing problems that remain unsolved. | **Students will be able to:**<br><br>● Research and use needed PHP functions.<br>● Write and use web servers.<br>● Use PHP to create login, forms have users interact with a database.<br>● Use phpMyAdmin used a GUI to manage data in a database.<br>● Create web based applications.<br>● Query information from a database.<br>● Code in JavaScript.<br>● Utilize Ajax to make optimize the use of HTML and JavaScript<br>● See and manipulate code that controls an AI.<br>● Realize the limitations of computers. |

## Stage 2: Acceptable Evidence

| | |
|---|---|
| **Performance Task & Unit Assessments:**<br><br>**Assessments -**<br>● Unit 6 Exam<br><br>**Transfer Tasks:**<br>● Problem 6-0: Birthday Party<br>Students will see how PHP provides abstractions that make it easy to process huge amounts of data quickly, and can incorporate PHP to create their own search engine.<br>Teacher Notes<br><br>● Problem 6-1: Address Book 1.0<br>Students will use PHP to collect information from users and, pending correctness, write the information to a csv file.<br>Teacher Notes<br><br>● Problem 6-2: C$50 Finance (Part 1)<br>Using PHP and SQL, write code for a website to accept and record usernames and passwords.<br><br>● Problem 6-3: C$50 Finance (Part 2)<br>After successfully accepting users, execute code that looks up real time stock information for users to buy and sell, recording the progress with currency given to users by the student.<br><br>● Problem 6-4: Mashup (Part 1)<br>Set is to implement "mashup" that integrates Google Maps with Google News with a MySQL database containing thousands of postal codes, GPS coordinates, and more.<br><br>**Performance Task 2: Create – Applications From Ideas** (Completion Time: 3 weeks/12 hours of in class time)<br>24% of overall AP Score<br>Programming is a collaborative and creative process that brings ideas to life through the development of software. Programs can help solve problems, enable innovations, or express personal interests. In this performance task, students will be developing a program of their choice. The students' development process should include iteratively designing, implementing, and testing their program. Students are strongly encouraged to work with another student in their class.<br><br>Students completing the AP Computer Science Principles course in a nontraditional classroom situation (e.g., online, homeschool, independent study) are permitted to | **Other Evidence:**<br>**Students will show that they have achieved Stage 1 goals by:**<br><br>Formal:<br>● Providing written/oral response to the EQs<br>● Passing all quizzes on basic concepts in unit.<br>● Receiving passing grades on Transfer Tasks<br><br>Informal:<br>● Understanding the Impact of technology on society<br>● Successfully implementing databases<br>● Seamlessly transitioning to PHP and/or Python<br>● Using knowledge from the course to create their own project for the AP Task |

collaborate with another
secondary level student peer.

There is no designated programming language for AP Computer Science Principles. Students may choose a programming language learned while taking this course to complete the task, or they may select a different programming language — one they are familiar with from outside of class. When selecting a programming language and their program focus, students should ensure that their program will be sophisticated enough to integrate mathematical and logical concepts, develop abstractions, and implement algorithms.

The following components are formally assessed:

▸ A video of your program running (V)

▸ Individual written responses about your program and development process (IWR)

▸ Program code (PC)

This performance task requires students to develop a program on a topic that interests them or solves a problem. It is strongly recommended that a portion of the program involve some form of collaboration with another student in their class. The program development must also involve a signi cant amount of independent work in the planning and designing parts of the process.

Students are required to:

▸ iteratively design, implement, and test their program;

▸ independently create at least one signi cant part of their program;

▸ create a video that displays the running of their program and demonstrates its functionality;

▸ write responses to questions about their program; and

▸ include their entire program code in their written submission.

In this task, students develop a program that demonstrates a variety of capabilities and implements several different language features that, when combined, produce a result that cannot easily be accomplished without computing tools and techniques. The program should draw upon mathematical and logical concepts, such as use of numbers, variables, mathematical expressions with
arithmetic operators, logical and Boolean operators and expressions, decision statements, iteration, and/or collections.

The program must demonstrate:

▸ use of several effectively integrated mathematical and logical concepts, from the language being used;

▸ implementation of algorithms that integrate other algorithms and mathematical and/or logical concepts; and

▸ development and use of abstractions to manage the complexity of their program (e.g., procedures, abstractions provided by the programming language, APIs).

Students are encouraged to collaborate with another student in their class when planning, developing, or implementing their program. Collaboration can take different forms and can occur at different times in the program development process. For example, students can begin working independently, or they can start by collaborating with a partner in the initial phase of the task.

The following are examples of different forms of collaboration:

▸ Collaboration may involve soliciting feedback from a collaborative partner at various points in their program development and providing feedback to a collaborative partner(s) at various points in their program development.

▸ Collaboration may involve each collaborative partner developing pieces of the program, combining those pieces, and providing feedback during the development process.

▸ Collaboration can take the form of brainstorming and sharing ideas before the process of writing program code begins. Collaborative partners can then choose to work together or independently at selected times during the programming process.

▸ Collaboration can take the form of working together to develop an idea, beginning the programming process together, and then working independently to add different features to the collaboratively developed portion of the program.

▸ Collaboration can employ pair programming, in which one collaborative partner “drives” (enters code) while the other “navigates” (recommends and reviews program code entered by driver), with the collaborative partners changing roles after designated time intervals.

| | |
|---|---|
| ▸ Collaboration can blend any or all of the above techniques and may include an iterative process in which one or more of these techniques, or other collaboration techniques, are employed several times in the program design, implementation, and testing phases. | |

## Reference Materials

- Verona AP CS50 - Unit 6
- Albert.io Global Impact Questions
- 6-0 PHP Teacher Notes
- 6-1 PHP for Web Programming Teacher Notes
  - Python Assignment
- 6-2 SQL Teacher Notes
  - SQL Assignment
- 6-3 MVC Teacher Notes
- 6-4 JavaScript Teacher Notes
- 6-5 Ajax Teacher Notes
- 6-6 Artificial Intelligence Teacher Notes
- 6-7 VR and AR Teacher Notes