CompuScholar, Inc.

Alignment to Arkansas **Essentials of Computer Programming** Standards

Course Title: **TeenCoder: Java Programming (Abridged)**
Course ISBN: **978-0-9887070-4-7**

Note 1: Arkansas standards were derived from this document:
www.arkansased.org/public/userfiles/Learning_Services/Curriculum%20and%20Instruction/Frameworks/Computer%20Science/Essentials_of_Computer_Programming.pdf

Note 2: Citation(s) listed may represent a subset of the actual instances where objectives are met throughout the course.

Computer Science Curriculum Framework 2015	
Course Title:	**Essentials of Computer Programming**
Course/Unit Credit:	1
Course Number:	460020
Grades:	9 - 12
Prerequisite:	N/A

Essentials of Computer Programming is a course for students with no formal computer programming experience. Students will spend the majority of time coding to solve real-world problems in a collaborative environment. The course focuses on developing computational thinking, refining problem-solving skills, and applying key programming concepts. Throughout the course, students will use developmentally appropriate and accurate terminology when communicating about technology. Essentials of Computer Programming does not require additional Arkansas Department of Education approval.

Strand: Computational Thinking **Content Standard 1: Students will use relevant program design and problem-solving strategies.**		**CITATION(S)**
CT.1.ECP.1	Discuss approaches to problem solving used in programming	Chapter 10, Lesson 1 Chapter 10, Lesson 2 Chapter 17, Lesson 4
CT.1.ECP.2	Represent problem-solving logic (e.g., flowcharts, pseudocode, role-play)	Chapter 17, Lesson 4
CT.1.ECP.3	Interpret logical expressions using Boolean operators (e.g., AND, NOT, OR)	Chapter 7, Lesson 1
CT.1.ECP.4	Demonstrate operator precedence in expressions (e.g., inside-out, left-to-right, order of operations, right-to-left)	Chapter 7, Lesson 1

CT.1.ECP.5	Identify the appropriate use of control flow constructs	Chapter 7, Lessons 2, 3, 4, 5
CT.1.ECP.6	Demonstrate how large problems can be deconstructed into a sequence of smaller problems	Chapter 10, Lesson 1
CT.1.ECP.7	Analyze and implement collaborative methods in problem solving when appropriate	Chapter 21 Supplemental Lesson 1

Strand: Computational Thinking **Content Standard 2: Students will evaluate different data representations for storing information.**		**CITATION(S)**
CT.2.ECP.1	Classify the types of information that can be stored as variables (e.g., integers, floating points, characters, Booleans, strings)	Chapter 4, Lesson 1 Chapter 4, Lesson 2 Chapter 5, Lesson 1
CT.2.ECP.2	Compare and contrast how variables of different data types are represented in computer memory (e.g., ASCII codes, binary numbers)	Chapter 4, Lesson 1 Chapter 17, Lesson 2
CT.2.ECP.3	Compare techniques for manipulating and using data (e.g., searching, sorting)	Chapter 19, Lesson 2 Chapter 19, Lesson 3
CT.2.ECP.4	Analyze data organizational structures (e.g., arrays, databases, files)	Chapter 14, Lesson 1 Chapter 14, Lesson 2 Chapter 18, Lesson 1

Strand: Computational Thinking **Content Standard 3: Students will create and evaluate algorithms to solve problems.**		**CITATION(S)**
CT.3.ECP.1	Define the term algorithm as used in programming	Chapter 17, Lesson 4
CT.3.ECP.2	Apply algorithmic reasoning to connect problem-solving processes to programming	Chapter 17, Lesson 4
CT.3.ECP.3	Investigate ways in which a sequence of algorithms can be combined to create new algorithms	Chapter 17, Lesson 4
CT.3.ECP.4	Compare and contrast the functional characteristics (e.g., clarity, efficiency, understandability) of algorithms that solve the same problem	Chapter 19, Lesson 2 Chapter 19, Lesson 3
CT.3.ECP.5	Develop a procedure to test an algorithm	Chapter 9, Lesson 3
CT.3.ECP.6	Validate the effectiveness of an algorithm in solving a defined problem	Chapter 17, Activity 2

Strand: Computing Practice and Programming Content Standard 4: Students will evaluate the use of programming languages to solve problems.		CITATION(S)
CPP.4.ECP.1	Compare and contrast computer programming paradigms and languages (e.g., machine code, object oriented languages, procedural languages, visual programming languages)	Chapter 1, Lesson 3 Chapter 2, Lesson 1 Chapter 10, Lesson 1
CPP.4.ECP.2	Describe the sequence of steps necessary to create and execute a program on a computer	Chapter 2, Lesson 1 Chapter 2, Lesson 2 Chapter 3, Lesson 3
CPP.4.ECP.3	Manually trace the execution path of code to understand flow of control	Chapter 7, Lessons 2, 3, 4, 5
CPP.4.ECP.4	Manually trace the execution path of code to determine the output for a sample input	Chapter 19, Lesson 1

Strand: Computing Practice and Programming Content Standard 5: Students will analyze the capabilities of a programming language.		CITATION(S)
CPP.5.ECP.1	Explain how variables are created and initialized	Chapter 4, Lesson 2
CPP.5.ECP.2	Explain how arrays/lists are created and initialized	Chapter 14, Lesson 1 Chapter 14, Lesson 2
CPP.5.ECP.3	Compare and contrast capabilities of a language for conditional branching (e.g., if, if-else, multi-branch)	Chapter 7, Lesson 2
CPP.5.ECP.4	Compare and contrast capabilities of a language for iteration (e.g., counter controlled, event controlled)	Chapter 7, Lesson 4 Chapter 7, Lesson 5
CPP.5.ECP.5	Investigate various capabilities of a language for abstraction including functions, parameters, and return values	Chapter 8, Lessons 1, 2, 3
CPP.5.ECP.6	Recognize potential problems or limitations with conditional branching, iteration, and abstraction features (e.g., infinite loops, range checking on parameters, unreachable branches)	Chapter 7, Lesson 5
CPP.5.ECP.7	Analyze traditional programming algorithms (e.g., data management, searching, sorting)	Chapter 19, Lesson 2 Chapter 19, Lesson 3
CPP.5.ECP.8	Examine methods used for the input and output of data	Chapter 2, Lesson 2 Chapter 5, Lesson 4 Chapter 6, Lesson 2

Strand: Computing Practice and Programming Content Standard 6: Students will create, test, and use computer programs to solve problems.		CITATION(S)
CPP.6.ECP.1	Design source code that is efficient and easy to read	Chapter 2, Lesson 2

CPP.6.ECP.2	Integrate clear and appropriate documentation within student designed source code	Chapter 2, Lesson 2
CPP.6.ECP.3	Use an editor or visual interface to create and execute programs	Chapter 3
CPP.6.ECP.4	Implement computing applications by integrating software development tools and techniques	See individual bullets below
	• variables of different data types	Chapter 4, Lesson 2
	• arrays/lists of different sizes	Chapter 14, Lesson 1 Chapter 14, Lesson 2
	• conditional branching	Chapter 7, Lesson 2 Chapter 7, Lesson 3
	• iteration	Chapter 14, Lesson 3
	• combinations of conditional branching and iteration	Chapter 17, Lesson 4
	• functions (e.g., input/output libraries, graphical display libraries, math function libraries, user-defined functions)	Chapter 2, Lesson 4 Chapter 8 Chapters 12 and 13 Chapter 17, Lesson 1
CPP.6.ECP.5	Ensure program correctness by utilizing debugging output statements and test cases	Chapter 9, Lesson 3
CPP.6.ECP.6	Compare and contrast the implementation of two or more programming solutions for the same problem	Chapter 19, Lesson 2 Chapter 19, Lesson 3

Strand: Elements of Computing **Content Standard 7: Students will classify different types of computers and systems.**		**CITATION(S)**
EC.7.ECP.1	Identify a variety of electronic devices (e.g., computers, Global Positioning Systems [GPS], smart devices, vehicles) that contain computational processors and execute programs	Chapter 1, Lesson 1
EC.7.ECP.2	Discuss current trends (e.g., improved user interaction, increasing processor speed and data storage, parallel and distributed systems) in computing	Chapter 1, Lesson 1

Strand: Elements of Computing **Content Standard 8: Students will describe the major components of a computer system.**		**CITATION(S)**
EC.8.ECP.1	Describe the hardware components of a computer	Chapter 1, Lesson 1
EC.8.ECP.2	Identify unique components (e.g., accelerometers, GPS, touch screens) in mobile computing devices	Chapter 1, Lesson 1
EC.8.ECP.3	Identify the main software components (e.g., operating system, programming tools, user applications) in a computer	Chapter 1, Lesson 2

Strand: Elements of Computing **Content Standard 9: Students will analyze the relationships within systems and components.**		**CITATION(S)**
EC.9.ECP.1	Demonstrate an understanding of the relationship of input and output among systems and components	Chapter 6, Lessons 1, 2, 3
EC.9.ECP.2	Apply strategies to solve routine hardware problems	n/a
EC.9.ECP.3	Evaluate tools (e.g., compilers, computer networks, interpreters, operating systems, program editors) necessary to support program execution	Chapter 1, Lesson 3 Chapter 2, Lesson 1 Chapter 3, Lesson 1

Strand: Social and Ethical Aspects of Programming **Content Standard 10: Students will analyze ethical and unethical uses of technology and programming.**		**CITATION(S)**
SEA.10.ECP.1	Differentiate among open source, freeware, and proprietary software licenses	Chapter 1, Lesson 4
SEA.10.ECP.2	Discuss consequences of misuse of information and technology	Chapter 1, Lesson 4
SEA.10.ECP.3	Discuss social and economic implications (e.g., local, national, global) associated with ethical and unethical hacking and software piracy	Chapter 1, Lesson 4
SEA.10.ECP.4	Demonstrate legal and ethical behaviors when using information and technology	Chapter 1, Lesson 4
SEA.10.ECP.5	Apply academic integrity in programming	Chapter 1, Lesson 4