CompuScholar, Inc.

Correlations to the Texas Essential Knowledge and Skills (TEKS):

Computer Science I

Texas Course Details:

Chapter	Chapter 127. Texas Essential Knowledge and Skills for CTE
Subchapter	Subchapter O. STEM
Course	§127.789. Computer Science I
Standards	Subchapter O (STEM)
TEKS Coverage	100%

CompuScholar Course Details:

Course Title:	Java Programming
Course ISBN:	978-1-946113-99-3
Course Year:	2023

Note 1: Citation(s) listed may represent a subset of the instances where objectives are met throughout the course.

Note 2: Citation(s) for a "Lesson" refer to the "Lesson Text" elements and associated "Activities" within the course, unless otherwise noted. The "Instructional Video" components are supplements designed to introduce or re-enforce the main lesson concepts, and the Lesson Text contains full details.

Course Standards

Knowledge and Skills Statement: (1) Employability. The student identifies various employment opportunities in the computer science field. The student is expected to:

Student Expectation	Citation(s)
(1.A) identify job opportunities and accompanying job duties and tasks;	Supplemental Chapter 2, Lesson 2
(1.B) examine the role of certifications, resumes, and portfolios in the computer science profession;	Supplemental Chapter 2, Lesson 2
(1.C) employ effective technical reading and writing skills;	Chapter 2, Lesson 4 Chapter 27, Lesson 2 Supplemental Chapter 2, Lesson 4
(1.D) employ effective verbal and non-verbal communication skills;	Supplemental Chapter 2, Lesson 2
(1.E) solve problems and think critically;	Chapter 11, Lesson 1 Chapter 13, Lesson 3

(1.F) demonstrate leadership skills and function effectively as a team member;	Chapter 27, Lessons 1, 2 Supplemental Chapter 2, Lesson 2
(1.G) communicate an understanding of legal and ethical responsibilities in relation to the field of computer science;	Chapter 1, Lessons 4, 5 Supplemental Chapter 3, Lesson 2
(1.H) demonstrate planning and time-management skills; and	Chapter 27, Lessons 1, 2
(1.I) compare university computer science programs.	Supplemental Chapter 2, Lesson 2

Knowledge and Skills Statement: (2) Communication and collaboration. The student communicates and collaborates with peers to contribute to his or her own learning and the learning of others. The student is expected to:	
Student Expectation	**Citation(s)**
(2.A) participate in learning communities as a learner, initiator, contributor, and teacher/mentor; and	Chapter 27, Lesson 1 Supplemental Chapter 2, Lesson 3
(2.B) seek and respond to advice from peers, educators, or professionals when evaluating quality and accuracy of the student's product	Chapter 27, Lesson 1 Supplemental Chapter 2, Lesson 4

Knowledge and Skills Statement: (3) Programming style and presentation. The student utilizes proper programming style and develops appropriate visual presentation of data, input, and output. The student is expected to:	
Student Expectation	**Citation(s)**
(3.A) create and properly label and display output;	Chapter 3, Lesson 3 Chapter 6, Lesson 3
(3.B) create interactive input interfaces, with relevant user prompts, to acquire data from a user such as console displays or Graphical User Interfaces (GUIs);	Chapter 5, Lesson 6
(3.C) write programs with proper programming style to enhance the readability and functionality of a code by using descriptive identifiers, internal comments, white space, spacing, indentation, and a standardized program style;	Chapter 2, Lesson 3 Chapter 3, Lesson 2 Chapter 14, Lesson 2
(3.D) format data displays using standard formatting styles; and	Chapter 3, Lesson 3 Chapter 6, Lesson 3
(3.E) display simple vector graphics using lines, circles, and rectangles.	Chapter 32, Lessons 1, 2

Knowledge and Skills Statement: (4) Critical thinking, problem solving, and decision making. The student uses appropriate strategies to analyze problems and design algorithms. The student is expected to:	
Student Expectation	**Citation(s)**
(4.A) use program design problem-solving strategies such as flowchart or pseudocode to create program solutions;	Chapter 13, Lessons 1, 2 Chapter 27, Lesson 2
(4.B) create a high-level program plan using a visual tool such as a flowchart or graphic organizer;	Chapter 13, Lesson 1 Chapter 27, Lesson 2

(4.C) identify the tasks and subtasks needed to solve a problem;	Chapter 13, Lessons 1, 2 Chapter 14, Lesson 5 Chapter 27, Lessons 1, 2
(4.D) identify the data types and objects needed to solve a problem;	Chapter 3, Lesson 1 Chapter 13, Lesson 1 Chapter 27, Lesson 2
(4.E) identify reusable components from existing code;	Chapter 14, Lesson 1
(4.F) design a solution to a problem;	Chapter 13, Lessons 1, 2 Chapter 27, Lesson 2
(4.G) code a solution from a program design;	Chapter 13, Lessons 2, 3 Chapter 17, Lesson 1 Chapter 27, Lesson 2
(4.H) identify error types, including syntax, lexical, run time, and logic;	Chapter 11, Lesson 1
(4.I) test program solutions with valid and invalid test data and analyze resulting behavior;	Chapter 10, Lesson 3 Chapter 11, Lesson 1 Chapter 27, Lesson 3
(4.J) debug and solve problems using error messages, reference materials, language documentation, and effective strategies;	Chapter 2, Lesson 4 Chapter 10, Lesson 1 Chapter 11, Lessons 1, 2
(4.K) create and implement common algorithms such as finding greatest common divisor, finding the biggest number out of three, finding primes, making change, and finding the average;	Chapter 13, Lesson 3 Chapter 18, Lesson 4 Chapter 19, Lesson 4
(4.L) create program solutions that address basic error handling such as preventing division by zero and type mismatch;	Chapter 10, Lessons 2, 3
(4.M) select the most appropriate construct for a defined problem;	Chapter 12, Lesson 2 Chapter 13, Lesson 1 Chapter 27, Lesson 1
(4.N) create program solutions by using the arithmetic operators to create mathematical expressions, including addition, subtraction, multiplication, real division, integer division, and modulus division;	Chapter 4, Lessons 1, 3
(4.O) create program solutions to problems using available mathematics library functions or operators, including absolute value, round, power, square, and square root;	Chapter 7, Lesson 3
(4.P) develop program solutions that use assignment;	Chapter 3, Lesson 2 Chapter 4, Lessons 1, 2
(4.Q) develop sequential algorithms to solve non-branching and non-iterative problems;	Chapter 2, Lesson 3 Chapter 13, Lessons 1, 3
(4.R) develop algorithms to decision-making problems using branching control statements;	Chapter 8, Lessons 2, 3 Chapter 13, Lessons 1, 3
(4.S) develop iterative algorithms and code programs to solve practical problems;	Chapter 12, Lessons 1, 2 Chapter 13, Lessons 3, 4

(4.T) demonstrate the appropriate use of the relational operators;	Chapter 8, Lesson 1
(4.U) demonstrate the appropriate use of the logical operators; and	Chapter 9, Lesson 2
(4.V) generate and use random numbers	Chapter 7, Lesson 3

Knowledge and Skills Statement: (5) Digital citizenship. The student explores and understands safety, legal, cultural, and societal issues relating to the use of technology and information. The student is expected to:	
Student Expectation	**Citation(s)**
(5.A) discuss and explain intellectual property, privacy, sharing of information, copyright laws, and software licensing agreements;	Chapter 1, Lessons 4, 5
(5.B) practice ethical acquisition and use of digital information;	Chapter 1, Lesson 4
(5.C) demonstrate proper digital etiquette, responsible use of software, and knowledge of acceptable use policies;	Chapter 1, Lessons 4, 5
(5.D) investigate privacy and security measures, including strong passwords, pass phrases, and other methods of authentication and virus detection and prevention; and	Chapter 1, Lesson 5
(5.E) investigate computing and computing-related advancements and the social and ethical ramifications of computer usage.	Supplemental Chapter 3, Lessons 2, 3

Knowledge and Skills Statement: (6) Technology operations, systems, and concepts. The student understands technology concepts, systems, and operations as they apply to computer science. The student is expected to:	
Student Expectation	**Citation(s)**
(6.A) identify and describe the function of major hardware components, including primary and secondary memory, a central processing unit (CPU), and peripherals;	Chapter 1, Lesson 2
(6.B) differentiate between current programming languages, discuss the general purpose for each language, and demonstrate knowledge of specific programming terminology and concepts and types of software development applications;	Chapter 2, Lessons 1, 2, 3 Chapter 3, Lesson 2 Chapter 8, Lessons 2, 3
(6.C) differentiate between a high-level compiled language and an interpreted language;	Chapter 2, Lesson 1
(6.D) identify and use concepts of object-oriented design;	Chapter 5, Lesson 1 Chapter 14, Lessons 1, 2
(6.E) differentiate between local and global scope access variable declarations;	Chapter 15, Lesson 2
(6.F) encapsulate data and associated subroutines into an abstract data type;	Chapter 5, Lesson 4 Chapter 14, Lessons 1, 5 Chapter 15, Lesson 3
(6.G) create subroutines that do not return values with and without the use of arguments and parameters;	Chapter 5, Lesson 3 Chapter 14, Lesson 5

(6.H) create subroutines that return typed values with and without the use of arguments and parameters;	Chapter 5, Lesson 3 Chapter 14, Lesson 5
(6.I) create calls to processes passing arguments that match parameters by number, type, and position;	Chapter 5, Lesson 3 Chapter 6, Lesson 3 Chapter 15, Lesson 4
(6.J) compare data elements using logical and relational operators;	Chapter 8, Lesson 1 Chapter 9, Lesson 2
(6.K) identify and convert binary representation of numeric and nonnumeric data in computer systems using American Standard Code for Information Interchange (ASCII) or Unicode;	Chapter 6, Lesson 1 Supplemental Chapter 1, Lesson 1
(6.L) identify finite limits of numeric data such as integer wrap around and floating point precision;	Chapter 3, Lesson 1 Chapter 4, Lesson 3
(6.M) perform numerical conversions between the decimal and binary number systems and count in the binary number system;	Chapter 7, Lesson 2
(6.N) choose, identify, and use the appropriate data types for integer, real, and Boolean data when writing program solutions;	Chapter 3, Lesson 1 Chapter 4, Lessons 1, 3 Chapter 8, Lesson 1
(6.O) analyze the concept of a variable, including primitives and objects;	Chapter 3, Lesson 2 Chapter 5, Lesson2 Chapter 14, Lesson 4
(6.P) represent and manipulate text data, including concatenation and other string functions;	Chapter 5, Lesson 2 Chapter 6, Lessons 1, 2
(6.Q) identify and use the structured data type of one-dimensional arrays to traverse, search, and modify data;	Chapter 18, Lessons 1 - 4 Chapter 20, Lesson 4
(6.R) choose, identify, and use the appropriate data type or structure to properly represent the data in a program problem solution; and	Chapter 19, Lessons 1, 2 Chapter 27, Lesson 2
(6.S) compare strongly typed and un-typed programming languages	Chapter 2, Lesson 1