CompuScholar, Inc.

Correlations to the Texas Essential Knowledge and Skills (TEKS): Tech Apps "Computer Science I"

Texas Course Details:

Chapter	Chapter 126. Texas Essential Knowledge and Skills for Tech. Apps
Subchapter	Subchapter C. High School
Course	§126.33. Computer Science I
TEKS Coverage	100%

CompuScholar Course Details:

Course Title:	Java Programming (Abridged)
Course ISBN:	9780988707047
Course Year:	2018

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) Creativity and innovation. The student develops products and generates new understandings by extending existing knowledge. The student is expected to:

Student Expectation	**Citation(s)**
(1) (A) participate with electronic communities as a learner, initiator, contributor, and teacher/mentor	Chapter 21 (Team Project)
(1) (B) extend the learning environment beyond the school walls with digital products created to increase teaching and learning in the other subject areas	Chapter 21 (Team Project) Supplemental Chapters 1, 2, 3
(1) (C) participate in relevant, meaningful activities in the larger community and society to create electronic projects	Chapter 21 (Team Project) Supplemental Chapters 1, 2, 3

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) create and properly display meaningful output	Chapter 5, Lesson 4 Chapter 12, Lessons 1-4 Chapter 13, Lessons 1-3
(2) (B) create interactive console display interfaces, with appropriate user prompts, to acquire data from a user	Chapter 6, Lesson 2
2) (C) use Graphical User Interfaces (GUIs) to create interactive interfaces to acquire data from a user and display program results	Chapter 12, Lessons 1-4 Chapter 13, Lessons 1-3
(2) (D) write programs with proper programming style to enhance the readability and functionality of the code by using meaningful descriptive identifiers, internal comments, white space, spacing, indentation, and a standardized program style	Chapter 2, Lesson 2 Chapter 4, Lesson 2
(2) (E) improve numeric display by optimizing data visualization	Chapter 5, Lesson 4 Chapter 17, Lesson 1
(2) (F) display simple vector graphics using lines, circles and rectangles	Chapter 20, Lesson 2
(2) (G) display simple bit map images	Chapter 20, Lesson 3
(2) (H) seek and respond to advice from peers and professionals in evaluating quality and accuracy	Chapter 21, Lesson 4

Knowledge and Skills Statement: (3) Research and information fluency. The student locates, analyzes, processes, and organizes data. The student is expected to:

Student Expectation	Citation(s)
(3) (A) use a variety of resources, including foundation and enrichment curricula, to gather authentic data as a basis for individual and group programming projects	Chapter 9, Lesson 3 Chapter 21, Lesson 1 Supplemental Chapter 1, Lesson 5
(3) (B) use various productivity tools to gather authentic data as a basis for individual and group programming projects	Chapter 9, Lesson 3 Chapter 21, Lesson 1 Supplemental Chapter 1, Lesson 5

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 to create program solutions	Chapter 17, Lesson 4
(4) (B) define and specify the purpose and goals of solving a problem	Chapter 17, Lesson 4
(4) (C) identify the subtasks needed to solve a problem	Chapter 8, Lesson 1

(4) (D) identify the data types and objects needed to solve a problem	Chapter 4, Lesson 2 Chapter 10, Lesson 2 Chapter 15, Lesson 2 Chapter 17, Lesson 2
(4) (E) identify reusable components from existing code	Chapter 8, Lesson 1
(4) (F) design a solution to a problem	Chapter 17, Lesson 4
(4) (G) code a solution from a program design	Chapter 16, Lesson 1 (Jail Break) Chapter 21, Lessons 2-3 (Team Project)
(4) (H) identify and debug errors	Chapter 9, Lessons 1 and 3 Chapter 21, Lesson 4
(4) (I) test program solutions with appropriate valid and invalid test data for correctness	Chapter 6, Lesson 3 Chapter 9, Lessons 1 and 3 Chapter 21, Lesson 4
(4) (J) debug and solve problems using error messages, reference materials, language documentation, and effective strategies	Chapter 9, Lesson 3
(4) (K) explore common algorithms, including greatest common divisor, finding the biggest number out of three, finding primes, making change, and finding the average	Chapter 17, Lesson 4
(4) (L) analyze and modify existing code to improve the underlying algorithm	Chapter 17, Lesson 4
(4) (M) create program solutions that exhibit robust behavior by understanding, avoiding, and preventing runtime errors, including division by zero and type mismatch	Chapter 9, Lesson 1
(4) (N) select the most appropriate algorithm for a defined problem	Chapter 19, Lesson 2
(4) (O) demonstrate proficiency in the use of the arithmetic operators to create mathematical expressions, including addition, subtraction, multiplication, real division, integer division and modulus division	Chapter 4, Lesson 2 Chapter 7, Lesson 1
(4) (P) create program solutions to problems using available mathematics libraries, including absolute value, round, power, square, and square root	Chapter 17, Lesson 1
(4) (Q) develop program solutions that use assignment	Chapter 4, Lesson 2 (Plus most subsequent lessons involve assignment statements).
(4) (R) develop sequential algorithms to solve non- branching and non-iterative problems	Chapter 17, Lesson 3
(4) (S) develop algorithms to decision-making problems using branching control statements	Chapter 17, Lesson 3
(4) (T) develop iterative algorithms and code programs to solve practical problems	Chapter 19, Lessons 2, 3

(4) (U) demonstrate proficiency in the use of the relational operators	Chapter 7, Lessons 1, 2
(4) (V) demonstrate proficiency in the use of the logical operators	Chapter 7, Lessons 1, 2
(4) (W) generate and use random numbers	Chapter 17, Lesson 1 Chapter 20 Activity

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 intellectual property, privacy, sharing of information, copyright laws, and software licensing agreements	Chapter 1, Lesson 4
(5) (B) model 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, Lesson 4
(5) (D) investigate measures, including passwords and virus detection/prevention, to protect computer systems and databases from unauthorized use and tampering	Chapter 1, Lesson 4 Supplemental Chapter 3, Lesson 1
(5) (E) investigate how technology has changed and the social and ethical ramifications of computer usage	Chapter 1, Lesson 1 Supplemental Chapter 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) compare and contrast types of operating systems, software applications, and programming languages	Chapter 1, Lesson 2
(6) (B) demonstrate knowledge of major hardware components, including primary and secondary memory, a central processing unit (CPU), and peripherals	Chapter 1, Lesson 1
(6) (C) differentiate among current programming languages, discuss the use of those languages in other fields of study, and demonstrate knowledge of specific programming terminology and concepts	Chapter 1, Lesson 3 Terms, keywords and programming concepts are introduced and used throughout the course
(6) (D) differentiate between a high-level compiled language and an interpreted language	Chapter 1, Lesson 3
(6) (E) understand concepts of object-oriented design	Chapters 10, 11, 15, and 16
(6) (F) use local and global scope access variable declarations	Chapter 10, Lesson 2 Chapter 4, Lesson 2
(6) (G) encapsulate data and associated subroutines into an abstract data type	Chapter 15, Lesson 2

(6) (H) create subroutines that do not return values with and without the use of arguments and parameters	Chapter 8, Lessons 1, 2
(6) (I) create subroutines that return typed values with and without the use of arguments and parameters	Chapter 8, Lesson 2
(6) (J) understand and identify the data-binding process between arguments and parameters	Chapter 8, Lesson 3
(6) (K) compare objects using reference values and a comparison routine	Chapter 5, Lesson 2 Chapter 7, Lesson 1 Chapter 15, Lesson 5
(6) (L) understand the binary representation of numeric and nonnumeric data in computer systems	Chapter 5, Lesson 3 Chapter 17, Lesson 2
(6) (M) understand the finite limits of numeric data	Chapter 17, Lesson 2
(6) (N) perform numerical conversions between the decimal and binary number systems and count in the binary number system	Chapter 17, Lesson 2
(6) (O) choose, identify, and use the appropriate data types for integer, real, and Boolean data when writing program solutions	Chapter 4, Lesson 1
(6) (P) demonstrate an understanding of the concept of a variable	Chapter 4, Lesson 2
(6) (Q) demonstrate an understanding of and use reference variables for objects	Chapter 5, Lesson 1
(6) (R) demonstrate an understanding of how to represent and manipulate text data, including concatenation and other string functions	Chapter 5, Lesson 3 Chapter 5, Lesson 4
(6) (S) demonstrate an understanding of the concept of scope	Chapter 10, Lesson 2
(6) (T) identify and use the structured data type of one- dimensional arrays to traverse, search, and modify data	Chapter 14, Lesson 1
(6) (U) choose, identify, and use the appropriate data type and structure to properly represent the data in a program problem solution	Chapter 4, Lesson 1 Chapter 10, Lesson 2 Chapter 14, Lesson 1
(6) (V) compare and contrast strongly typed and un-typed programming languages	Chapter 1, Lesson 3