§126.33. Computer Science I TEKS

(a) General requirements. Students shall be awarded one-half to one credit for successful completion of this course. The required prerequisite for this course is Algebra I. This course is recommended for students in Grades 9-12.

(b) Introduction.

(1) The technology applications curriculum has six strands based on the National Educational Technology Standards for Students (NETS•S) and performance indicators developed by the International Society for Technology in Education (ISTE): creativity and innovation; communication and collaboration; research and information fluency; critical thinking, problem solving, and decision making; digital citizenship; and technology operations and concepts.

(2) Computer Science I will foster students' creativity and innovation by presenting opportunities to design, implement, and present meaningful programs through a variety of media. Students will collaborate with one another, their instructor, and various electronic communities to solve the problems presented throughout the course. Through data analysis, students will identify task requirements, plan search strategies, and use computer science concepts to access, analyze, and evaluate information needed to solve problems. By using computer science knowledge and skills that support the work of individuals and groups in solving problems, students will select the technology appropriate for the task, synthesize knowledge, create solutions, and evaluate the results. Students will learn digital citizenship by researching current laws and regulations and by practicing integrity and respect. Students will gain an understanding of the principles of computer science through the study of technology operations, systems, and concepts.

(3) Statements that contain the word "including" reference content that must be mastered, while those containing the phrase "such as" are intended as possible illustrative examples.

(c) Knowledge and skills.

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

(A) participate with electronic communities as a learner, initiator, contributor, and teacher/mentor;

(B) extend the learning environment beyond the school walls with digital products created to increase teaching and learning in the other subject areas; and

(C) participate in relevant, meaningful activities in the larger community and society to create electronic projects.

(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:

(A) create and properly display meaningful output;

(B) create interactive console display interfaces, with appropriate user prompts, to acquire data from a user;

(C) use Graphical User Interfaces (GUIs) to create interactive interfaces to acquire data from a user and display program results;

(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;

(E) improve numeric display by optimizing data visualization;

(F) display simple vector graphics using lines, circles, and rectangles;

(G) display simple bitmap images; and

(H) seek and respond to advice from peers and professionals in evaluating quality and accuracy.

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

(A) use a variety of resources, including foundation and enrichment curricula, to gather authentic data as a basis for individual and group programming projects; and

(B) use various productivity tools to gather authentic data as a basis for individual and group programming projects.

(4) Critical thinking, problem solving, and decision making. The student uses appropriate strategies to analyze problems and design algorithms. The student is expected to:

(A) use program design problem-solving strategies to create program solutions;

(B) define and specify the purpose and goals of solving a problem;

(C) identify the subtasks needed to solve a problem;

(D) identify the data types and objects needed to solve a problem;

(E) identify reusable components from existing code;

(F) design a solution to a problem;

(G) code a solution from a program design;

(H) identify and debug errors;

(I) test program solutions with appropriate valid and invalid test data for correctness;

(J) debug and solve problems using error messages, reference materials, language documentation, and effective strategies;

(K) explore common algorithms, including finding greatest common divisor, finding the biggest number out of three, finding primes, making change, and finding the average;

(L) analyze and modify existing code to improve the underlying algorithm;

(M) create program solutions that exhibit robust behavior by understanding, avoiding, and preventing runtime errors, including division by zero and type mismatch;

(N) select the most appropriate algorithm for a defined problem;

(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;

(P) create program solutions to problems using available mathematics libraries, including absolute value, round, power, square, and square root;

(Q) develop program solutions that use assignment;

(R) develop sequential algorithms to solve non-branching and non-iterative problems;

(S) develop algorithms to decision-making problems using branching control statements;

(T) develop iterative algorithms and code programs to solve practical problems;

(U) demonstrate proficiency in the use of the relational operators;

(V) demonstrate proficiency in the use of the logical operators; and

(W) generate and use random numbers.

(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:

(A) discuss intellectual property, privacy, sharing of information, copyright laws, and software licensing agreements;

(B) model ethical acquisition and use of digital information;

(C) demonstrate proper digital etiquette, responsible use of software, and knowledge of acceptable use policies;

(D) investigate measures, including passwords and virus detection/prevention, to protect computer systems and databases from unauthorized use and tampering; and

(E) investigate how technology has changed and the social and ethical ramifications of computer usage.

(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:

(A) compare and contrast types of operating systems, software applications, and programming languages;

(B) demonstrate knowledge of major hardware components, including primary and secondary memory, a central processing unit (CPU), and peripherals;

(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;

(D) differentiate between a high-level compiled language and an interpreted language;

(E) understand concepts of object-oriented design;

(F) use local and global scope access variable declarations;

(G) encapsulate data and associated subroutines into an abstract data type;

(H) create subroutines that do not return values with and without the use of arguments and parameters;

(I) create subroutines that return typed values with and without the use of arguments and parameters;

(J) understand and identify the data-binding process between arguments and parameters;

(K) compare objects using reference values and a comparison routine;

(L) understand the binary representation of numeric and nonnumeric data in computer systems;

(M) understand the finite limits of numeric data;

(N) perform numerical conversions between the decimal and binary number systems and count in the binary number system;

(O) choose, identify, and use the appropriate data types for integer, real, and Boolean data when writing program solutions;

(P) demonstrate an understanding of the concept of a variable;

(Q) demonstrate an understanding of and use reference variables for objects;

(R) demonstrate an understanding of how to represent and manipulate text data, including concatenation and other string functions;

(S) demonstrate an understanding of the concept of scope;

(T) identify and use the structured data type of one-dimensional arrays to traverse, search, and modify data;

(U) choose, identify, and use the appropriate data type and structure to properly represent the data in a program problem solution; and

(V) compare and contrast strongly typed and un-typed programming languages.

Source: The provisions of this §126.33 adopted to be effective September 26, 2011, 36 TexReg 6263.