eDynamicLearning
CAREER & ELECTIVE COURSES

eDynamic Learning Course Title: Coding 1a / 1b

State: Texas
State Course Title: Computer Science I
State Course Code: 126.33
State Standards: Computer Science I
Date of Standards: 26 Sept. 2011

TEKS	Unit Name(s)	Lesson(s) Numbers
(1) Creativity and innovation. The student develops products and generates new understandings by extending existing knowledge.		
(A) participate with electronic communities as a learner, initiator, contributor, and teacher/mentor	1b: Unit 7: Developing Web Solutions Part 1	1b: Unit 7: L: Activity
(B) extend the learning environment beyond the school walls with digital products created to increase teaching and learning in the other subject areas	1b: Unit 7: Developing Web Solutions Part 1	1b: Unit 7: L: 1-4
(C) participate in relevant, meaningful activities in the larger community and society to create electronic projects	1b: Unit 4: Modular Programming	1b: Unit 4: Lab
(2) Communication and collaboration. The student communicates and collaborates with peers to contribute to his or her own learning and the learning of others.		
(A) create and properly display meaningful output	1b: Unit 2: The Software-Human Interface	1b: Unit 2: L: 4
(B) create interactive console display interfaces, with appropriate user prompts, to acquire data from a user	1a: Unit 8: Capstone Project Part B; Unit 2: The Big Programming Picture	1a: Unit 8: L: 2; Unit 2: L: 6
(C) use Graphical User Interfaces (GUIs) to create interactive interfaces to acquire data from a user and display program results	1a: Unit 2: The Big Programming Picture; 1b: Unit 2: The Software-Human Interface	1a: Unit 2: L: 6; 1b: Unit 2: L: 4
(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	1b: Unit 2: The Software-Human Interface; Unit 3: Coding Structures	1b: Unit 2: Lab, Activity; Unit 3: L: 3
(E) improve numeric display by optimizing data visualization	1b: Unit 2: The Software-Human Interface	1b: Unit 2: Lab
(F) display simple vector graphics using lines, circles, and rectangles	1b: Unit 2: The Software-Human Interface	1b: Unit 2: L: 4, Activity 2
(G) display simple bitmap images	1b: Unit 2: The Software-Human Interface	1b: Unit 2: L: 4, Activity 2

(H) seek and respond to advice from peers and professionals in evaluating quality and accuracy	1b: Unit 6: Top-Notch Programming	1b: Unit 6: L: 4
(3) Research and information fluency. The student locates, analyzes, processes, and organizes data.		
(A) use a variety of resources, including foundation and enrichment curricula, to gather authentic data as a basis for individual and group programming projects	1b: Unit 6: Top-Notch Programming	1b: Unit 6: Lab
(B) use various productivity tools to gather authentic data as a basis for individual and group programming projects	1b: Unit 3: Coding Structures	1b: Unit 3: Activity
(4) Critical thinking, problem solving, and decision making. The student uses appropriate strategies toanalyze problems and design algorithms.		
(A) use program design problem-solving strategies to create program solutions	1b: Unit 3: Coding Structures	1b: Unit 3: Activity 1
(B) define and specify the purpose and goals of solving a problem	1b: Unit 3: Coding Structures	1b: Unit 3: Activity 2
(C) identify the subtasks needed to solve a problem	1b: Unit 6: Top-Notch Programming	1b: Unit 6: Activity
(D) identify the data types and objects needed to solve a problem	1b: Unit 5: Data Security	1b: Unit 5: Activity
(E) identify reusable components from existing code	1b: Unit 6: Top-Notch Programming	1b: Unit 6: Lab
(F) design a solution to a problem	1b: Unit 3: Coding Structures	1b: Unit 3: Activity 2
(G) code a solution from a program design	1b: Unit 6: Top-Notch Programming	1b: Unit 6: Activity
(H) identify and debug errors	1a: Unit 2: The Big Programming Picture; 1b: Unit 1: Get the IDEa	1a: Unit 2: L: 6; 1b: Unit 1: L: 4
(I) test program solutions with appropriate valid and invalid test data for correctness	1a: Unit 4: The Big Process Picture; 1b: Unit 2: The Software-Human Interface	1a: Unit 4: L: 1, 5; Ib: Unit 2: L: 3
(J) debug and solve problems using error messages, reference materials, language documentation, and effective strategies	1a: Unit 4: The Big Process Picture; 1b: Unit 7: Developing web Solutions	1a: Unit 4: L: 2, 5; 1b: Unit 7: L: 4
(K) explore common algorithms, including finding greatest common divisor, finding the biggest number out of three, finding primes, making change, and finding the average	1a: Unit 3: Let's Do Some Programming; Unit 7: Capstone Project Part 1	1a: Unit 3: L: 1; Unit 7: Lab
(L) analyze and modify existing code to improve the underlying algorithm	1b: Unit 6: Top-Notch Programming; Unit 8: Developing Web Solutions Part 2	1b: Unit 6: L: 3; Unit 8: L: 2, Lab
(M) create program solutions that exhibit robust behavior by understanding, avoiding, and preventing runtime errors, including division by zero and type mismatch	1b: Unit 4: Modular Programming	1b: Unit 4: L: 5
(N) select the most appropriate algorithm for a defined problem	1a: Unit 3: Let's Do Some Programming; Unit 7: Capstone Project Part 1	1a: Unit 3: L: 1; Unit 7: Lab

(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	1b: Unit 2: The Software-Human Interface	1b: Unit 2: L: 2
(P) create program solutions to problems using available mathematics libraries, including absolute value, round, power, square, and square root	1b: Unit 2: The Software-Human Interface	1b: Unit 2: L: 5
(Q) develop program solutions that use assignment	1b: Unit 6: Top-Notch Programming; Unit 8: Developing Web Solutions, Part 2	1b: Unit 6: L: 2; Unit 8: L: 2, Lab
(R) develop sequential algorithms to solve non-branching and non-iterative problems	1b: Unit 6: Top-Notch Programming; Unit 8: Developing Web Solutions, Part 2	1b: Unit 6: L: 2; Unit 8: L: 2, Lab
(S) develop algorithms to decision-making problems using branching control statements	1b: Unit 6: Top-Notch Programming; Unit 8: Developing Web Solutions, Part 2	1b: Unit 6: L: 2; Unit 8: L: 2, Lab
(T) develop iterative algorithms and code programs to solve practical problems	1a: Unit 3: Let's Do Some Programming	1a: Unit 3: Lab
(U) demonstrate proficiency in the use of the relational operators	1b: Unit 3: Coding Structures	1b: Unit 3: L: 1
(V) demonstrate proficiency in the use of the logical operators	1b: Unit 3: Coding Structures	1b: Unit 3: L: 1
(W) generate and use random numbers	1b: Unit 6: Top-Notch Programming	1b: Unit 6: L: 1
(5) Digital citizenship. The student explores and understands safety, legal, cultural, and societal issues relating to the use of technology and information.		
(A) discuss intellectual property, privacy, sharing of information, copyright laws, and software licensing agreements	1a: Unit 6: Security, Privacy, and Other Issues	1a: Unit 6: L: 4, 5
(B) model ethical acquisition and use of digital information	1a: Unit 6: Security, Privacy, and Other Issues	1a: Unit 6: L: 5
(C) demonstrate proper digital etiquette, responsible use of software, and knowledge of acceptable use policies	1a: Unit 6: Security, Privacy, and Other Issues	1a: Unit 6: L: 3
(D) investigate measures, including passwords and virus detection/prevention, to protect computer systems and databases from unauthorized use and tampering	1a: Unit 6: Security, Privacy, and Other Issues; 1b: Unit 5: Data Security	1a: Unit 6: L: 2; 1b: Unit 5: Lab
(E) investigate how technology has changed and the social and ethical ramifications of computer usage	1a: Unit 6: Security, Privacy, and Other Issues; Unit 2: The Big Programming Picture	1a: Unit 6: L: 1-5; Unit 2: L: 1
(6) Technology operations, systems, and concepts. The student understands technology concepts, systems, and operations as they apply to computer science.		
(A) compare and contrast types of operating systems, software applications, and programming languages	1b: Unit 1: Get the IDEa?	1b: Unit 1: L: 2
(B) demonstrate knowledge of major hardware components, including primary and secondary memory, a central processing unit (CPU), and peripherals	1b: Unit 1: Get the IDEa?	1b: Unit 1: L: 1

(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	1a: Unit 2: The Big Programming Picture	1b: Unit 2: L: 2
(D) differentiate between a high-level compiled language and an interpreted language	1a: Unit 2: The Big Programming Picture	1b: Unit 2: L: 2
(E) understand concepts of object-oriented design	1a: Unit 2: The Big Programming Picture	1b: Unit 2: L: 3
(F) use local and global scope access variable declarations	1b: Unit 4: Modular Programming	1b: Unit 4: L: 3
(G) encapsulate data and associated subroutines into an abstract data type	1b: Unit 1: Get the IDEa?	1b: Unit 1: L: 1
(H) create subroutines that do not return values with and without the use of arguments and parameters	1b: Unit 2: The Software-Human Interface; Unit 4: Modular Programming	1b: ; Unit 4: L: 2Unit 2: L: 2
(I) create subroutines that return typed values with and without the use of arguments and parameters	1b: Unit 2: The Software-Human Interface; Unit 4: Modular Programming	1b: ; Unit 4: L: 2Unit 2: L: 2
(J) understand and identify the data-binding process between arguments and parameters	1b: Unit 2: The Software-Human Interface; Unit 4: Modular Programming	1b: ; Unit 4: L: 2Unit 2: L: 2
(K) compare objects using reference values and a comparison routine	1a: Unit 2; The Big Programming Picture	1a: Unit 2: L: 3
(L) understand the binary representation of numeric and nonnumeric data in computer systems	1b: Unit 1: Get the IDEa?	1b: Unit 1: L: Lab
(M) understand the finite limits of numeric data	1b: Unit 6: Top-Notch Programming	1b: Unit 6: L: 1
(N) perform numerical conversions between the decimal and binary number systems and count in the binary number system	1b: Unit 1: Get the IDEa?	1b: Unit 1: L: Lab
(O) choose, identify, and use the appropriate data types for integer, real, and Boolean data when writing program solutions	1b: Unit 2: The Software-Human Interface	1b: Unit 2: L: 1
(P) demonstrate an understanding of the concept of a variable	1a: Unit 3: Let's Do Some Programming	1a: Unit 3: L: 4
(Q) demonstrate an understanding of and use reference variables for objects	1b: Unit 2: The Software-Human Interface	1b: Unit 2: L: 1
(R) demonstrate an understanding of how to represent and manipulate text data, including concatenation and other string functions	1b: Unit 2: The Software-Human Interface	1b: Unit 2: L: 2
(S) demonstrate an understanding of the concept of scope	1a: Unit 7: Capstone Project Part 1	1a: Unit 7: L: 2
(T) identify and use the structured data type of one-dimensional arrays to traverse, search, and modify data	1b: Unit 3: Coding Structures	1b: Unit 3: L: 5
(U) choose, identify, and use the appropriate data type and structure to properly represent the data in a program problem solution	1b: Unit 2: The Software-Human Interface	1b: Unit 2: L: 1
(V) compare and contrast strongly typed and un-typed programming languages	1a: Unit 2: The Big Programming Picture	1a: Unit 2: L: 4