eDynamicLearning
CAREER & ELECTIVE COURSES

eDynamic Learning Course Title: Programming 1a/1b

State: TX
State Course Title: Computer Science 1
State Course Code: 127.789
State Standards: Computer Science 1
Date of Standards: 2021

TEKS	Course Title (a or b), if applicable, e.g. Game Design 1a	Unit Name(s)	Lesson(s) Numbers
(1) Employability. The student identifies various employment opportunities in the computer science field. The student is expected to:			
(A) identify job opportunities and accompanying job duties and tasks;	Introduction to Programming 1a: Introduction	Unit 8: Skill Spotlight: A World of Programming	Lesson 3, Lab
(B) examine the role of certifications, resumes, and portfolios in the computer science profession;	Introduction to Programming 1a: Introduction	Unit 8: Skill Spotlight: A World of Programming	Lesson 3
(C) employ effective technical reading and writing skills;	Introduction to Programming 1b: Problem Solving Through Programming	Unit 8: Skill Spotlight: Involved and Informed	Lesson 2, Lab
(D) employ effective verbal and non-verbal communication skills;	Introduction to Programming 1a: Introduction	Unit 8: Skill Spotlight: A World of Programming	Lesson 2
(E) solve problems and think critically;	Introduction to Programming 1a: Introduction	Unit 3: Problems and Solutions	All Lessons Associated
(F) demonstrate leadership skills and function effectively as a team member;	Introduction to Programming 1a: Introduction	Unit 8: Skill Spotlight: A World of Programming	Lesson 2
(G) demonstrate an understanding of legal and ethical responsibilities in relation to the field of computer science;	Introduction to Programming 1b: Problem Solving Through Programming	Unit 6: Legal and Ethical Computing	Lessons 1-3
(H) demonstrate planning and time-management skills; and	Introduction to Programming 1a: Introduction	Unit 3: Problems and Solutions	Lab
(I) compare university computer science programs.			
(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) participate in learning communities as a learner, initiator, contributor, and teacher/mentor; and	Introduction to Programming 1b: Problem Solving Through Programming	Unit 8: Skill Spotlight: Involved and Informed	Activity
(B) seek and respond to advice from peers, educators, or professionals when evaluating quality and accuracy of the student's product.	Introduction to Programming 1b: Problem Solving Through Programming	Unit 8: Skill Spotlight: Involved and Informed	Activity
(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:			
(A) create and properly label and display output;	Introduction to Programming 1a: Introduction	Unit 4: A Deep Dive with Data	Activity

(B) create interactive input interfaces, with relevant user prompts, to acquire data from a user such as console displays or Graphical User Interfaces (GUIs);	Introduction to Programming 1a: Introduction	Unit 6: The Data Files	Lessons 2, 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;	Introduction to Programming 1b: Problem Solving Through Programming	Unit 3: Abstraction	Activity
(D) format data displays using standard formatting styles; and	Introduction to Programming 1a: Introduction	Unit 6: The Data Files	Lessons 2, 3
(E) display simple vector graphics using lines, circles, and rectangles.	Introduction to Programming 1a: Introduction	Unit 7: Running the Numbers	Lesson 3
(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;	Introduction to Programming 1b: Problem Solving Through Programming	Unit 1: Designing Programs	Activity
(B) create a high-level program plan using a visual tool such as a flow chart or graphic organizer;	Introduction to Programming 1b: Problem Solving Through Programming	Unit 2: Plan for Success	Lessons 1, 2
(C) identify the tasks and subtasks needed to solve a problem;	Introduction to Programming 1a: Introduction	Unit 3: Problems and Solutions	Lesson 2
	Introduction to Programming 1b: Problem Solving Through Programming	Unit 2: Plan for Success	Lessons 1, 2
(D) identify the data types and objects needed to solve a problem;	Introduction to Programming 1b: Problem Solving Through Programming	Unit 3: Abstraction	Activity
(E) identify reusable components from existing code;	Introduction to Programming 1b: Problem Solving Through Programming	Unit 1: Designing Programs	Lesson 3
(F) design a solution to a problem;	Introduction to Programming 1b: Problem Solving Through Programming	Unit 1: Designing Programs	Activity
(G) code a solution from a program design;	Introduction to Programming 1b: Problem Solving Through Programming	Unit 3: Abstraction	Activity
(H) identify error types, including syntax, lexical, run time, and logic;	Introduction to Programming 1b: Problem Solving Through Programming	Unit 5: Running the Tests	Lesson 2
(I) test program solutions with valid and invalid test data and analyze resulting behavior;	Introduction to Programming 1b: Problem Solving Through Programming	Unit 5: Running the Tests	Lesson 3
(J) debug and solve problems using error messages, reference materials, language documentation, and effective strategies;	Introduction to Programming 1b: Problem Solving Through Programming	Unit 5: Running the Tests	Lessons 2, 3
(K) explore common algorithms such as finding greatest common divisor, finding the biggest number out of three, finding primes, making change, and finding the average;	Introduction to Programming 1a: Introduction	Unit 7: Running the Numbers	Lesson 2
(L) create program solutions that address basic error handling such as preventing division by zero and type mismatch;	Introduction to Programming 1b: Problem Solving Through Programming	Unit 1: Designing Programs	Lab
(M) select the most appropriate construct for a defined problem;	Introduction to Programming 1a: Introduction	Unit 5: All About Algorithms	Activity

(N) create program solutions by using the arithmetic operators to create mathematical expressions, including addition, subtraction, multiplication, real division, integer division, and modulus division;	Introduction to Programming 1a: Introduction	Unit 3: Problems and Solutions	Activity
(O) create program solutions to problems using available mathematics library functions or operators, including absolute value, round, power, square, and square root;	Introduction to Programming 1a: Introduction	Unit 7: Running the Numbers	Lesson 2, Lab
(P) develop program solutions that use assignment;	Introduction to Programming 1a: Introduction	Unit 3: Problems and Solutions	Lesson 3
(Q) develop sequential algorithms to solve non-branching and non-iterative problems;	Introduction to Programming 1a: Introduction	Unit 5: All About Algorithms	Lesson 1
(R) develop algorithms to decision-making problems using branching control statements;	Introduction to Programming 1a: Introduction	Unit 5: All About Algorithms	Lesson 2
(S) develop iterative algorithms and code programs to solve practical problems;	Introduction to Programming 1a: Introduction	Unit 5: All About Algorithms	Lesson 3
(T) demonstrate proficiency in the use of the relational operators;	Introduction to Programming 1a: Introduction	Unit 5: All About Algorithms	Lesson 2
(U) demonstrate proficiency in the use of the logical operators; and	Introduction to Programming 1a: Introduction	Unit 5: All About Algorithms	Lesson 2
(V) generate and use random numbers.	Introduction to Programming 1a: Introduction	Unit 5: All About Algorithms	Lesson 3
(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;	Introduction to Programming 1b: Problem Solving Through Programming	Unit 6: Legal and Ethical Computing	Lessons 1-3
(B) model ethical acquisition and use of digital information;	Introduction to Programming 1b: Problem Solving Through Programming	Unit 6: Legal and Ethical Computing	Lesson 3
(C) demonstrate proper digital etiquette, responsible use of software, and knowledge of acceptable use policies;	Introduction to Programming 1b: Problem Solving Through Programming	Unit 6: Legal and Ethical Computing	Lesson 1
(D) investigate measures, including strong passwords, pass phrases, and other methods of authentication, as well as virus detection/prevention for privacy and security; and	Introduction to Programming 1b: Problem Solving Through Programming	Unit 7: Safe and Secure	All Lessons Associated
(E) investigate computing and computing-related advancements and the social and ethical ramifications of computer usage.	Introduction to Programming 1a: Introduction	Unit 8: Skill Spotlight: A World of Programming	Lesson 1
(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) demonstrate knowledge of major hardware components, including primary and secondary memory, a central processing unit (CPU), and peripherals;	Introduction to Programming 1a: Introduction	Unit 4: A Deep Dive with Data	Lesson 1
(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;	Introduction to Programming 1a: Introduction	Unit 2: Speaking the Language	Lessons 1, 2
(C) differentiate between a high-level compiled language and an interpreted language;	Introduction to Programming 1a: Introduction	Unit 2: Speaking the Language	Lesson 3

(D) identify and use concepts of object-oriented design;	Introduction to Programming 1a: Introduction	Unit 2: Speaking the Language	Lesson 2, Lab
	Introduction to Programming 1b: Problem Solving Through Programming	Unit 3: Abstraction	Lab
(E) differentiate between local and global scope access variable declarations;	Introduction to Programming 1b: Problem Solving Through Programming	Unit 3: Abstraction	Lesson 3
(F) encapsulate data and associated subroutines into an abstract data type;	Introduction to Programming 1a: Introduction	Unit 7: Running the Numbers	Lesson 4
(G) create subroutines that do not return values with and without the use of arguments and parameters;	Introduction to Programming 1a: Introduction	Unit 6: The Data Files	Activity
(H) create subroutines that return typed values with and without the use of arguments and parameters;	Introduction to Programming 1a: Introduction	Unit 6: The Data Files	Activity
(I) create calls to processes passing arguments that match parameters by number, type, and position;	Introduction to Programming 1a: Introduction	Unit 6: The Data Files	Lab
(J) compare data elements using logical and relational operators;	Introduction to Programming 1a: Introduction	Unit 4: A Deep Dive with Data	Lesson 1
(K) identify and convert binary representation of numeric and nonnumeric data in computer systems using American Standard Code for Information Interchange (ASCII) or Unicode;	Introduction to Programming 1a: Introduction	Unit 4: A Deep Dive with Data	Lesson 2
(L) identify finite limits of numeric data such as integer wrap around and floating point precision;	Introduction to Programming 1a: Introduction	Unit 7: Running the Numbers	Lesson 2
(M) perform numerical conversions between the decimal and binary number systems and count in the binary number system;	Introduction to Programming 1a: Introduction	Unit 7: Running the Numbers	Lesson 1
(N) choose, identify, and use the appropriate data types for integer, real, and Boolean data when writing program solutions;	Introduction to Programming 1a: Introduction	Unit 6: The Data Files	Activity
(O) analyze the concept of a variable, including primitives and objects;	Introduction to Programming 1a: Introduction	Unit 2: Speaking the Language	Lessons 1, 3
(P) represent and manipulate text data, including concatenation and other string functions;	Introduction to Programming 1a: Introduction	Unit 4: A Deep Dive with Data	Lesson 3
(Q) identify and use the structured data type of one-dimensional arrays to traverse, search, and modify data;	Introduction to Programming 1a: Introduction	Unit 7: Running the Numbers	Lesson 4
(R) choose, identify, and use the appropriate data type or structure to properly represent the data in a program problem solution; and	Introduction to Programming 1a: Introduction	Unit 7: Running the Numbers	Lesson 4
(S) compare strongly typed and un-typed programming languages.	Introduction to Programming 1a: Introduction	Unit 2: Speaking the Language	Lesson 3