Texas CTE: Science, Technology, Engineering, and Mathematics

Computer Science II (2022)

Implementation. The provisions of this section shall be implemented by school districts beginning with the 2024-2025 school year. A

1 **No later than August 1, 2024, the commissioner of education shall determine whether instructional materials funding has been made available to Texas public schools for materials that cover the essential knowledge and skills identified in this section.** A.1

2 **If the commissioner makes the determination that instructional materials funding has been made available this section shall be implemented beginning with the 2024-2025 school year and apply to the 2024-2025 and subsequent school years.** A.2

3 **If the commissioner does not make the determination that instructional materials funding has been made available under subsection (a) of this section, the commissioner shall determine no later than August 1 of each subsequent school year whether instructional materials funding has been made available. If the commissioner determines that instructional materials funding has been made available, the commissioner shall notify the State Board of Education and school districts that this section shall be implemented for the following school year.** A.3

General requirements. This course is recommended for students in Grades 10-12. Prerequisites: Algebra I and Computer Science I or AP Computer Science Principles. Students shall be awarded one credit for successful completion of this course. B

b **General requirements. This course is recommended for students in Grades 10-12. Prerequisites: Algebra I and Computer Science I or AP Computer Science Principles. Students shall be awarded one credit for successful completion of this course.** B

Introduction. C

1 **Career and technical education instruction provides content aligned with challenging academic standards, industry-relevant technical knowledge, and college and career readiness skills for students to further their education and succeed in current and emerging professions.** C.1

2 **The Science, Technology, Engineering, and Mathematics (STEM) Career Cluster focuses on planning, managing, and providing scientific research and professional and technical services such as laboratory and testing services and research and development services.** C.2

3 **Computer Science II 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 computational thinking and 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 gain an understanding of computer science through the study of technology operations, systems, and concepts.** C.3

4 **Students are encouraged to participate in extended learning experiences such as career and technical student organizations and other leadership or extracurricular organizations.** C.4

5 **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.5

Knowledge and skills. D

1 **Employability. The student identifies various employment opportunities in the computer science field. The student is expected to:** D.1

- A identify job and internship opportunities and accompanying job duties and tasks and contact one or more companies or organizations to explore career opportunities; D.1.A
- B examine the role of certifications, resumes, and portfolios in the computer science profession; D.1.B
- C employ effective technical reading and writing skills; D.1.C
- D employ effective verbal and non-verbal communication skills; D.1.D
- E solve problems and think critically; D.1.E
- F demonstrate leadership skills and function effectively as a team member; D.1.F
- G identify legal and ethical responsibilities in relation to the field of computer science; D.1.G
- H demonstrate planning and time-management skills; and D.1.H
- I compare university computer science programs. D.1.II

2 **Creativity and innovation. The student develops products and generates new understandings by extending existing knowledge. The student is expected to:** D.2

- A use program design problem-solving strategies to create program solutions; D.2.A
- B read, analyze, and modify programs and their accompanying documentation such as an application programming interface (API), internal code comments, external documentation, or readme files; D.2.B
- C follow a systematic problem-solving process that identifies the purpose and goals, the data types and objects needed, and the subtasks to be performed; D.2.C
- D compare design methodologies and implementation techniques such as top-down, bottom-up, and black box; D.2.D
- E trace a program, including inheritance and black box programming; D.2.E
- F choose, identify, and use the appropriate abstract data type, advanced data structure, and supporting algorithms to properly represent the data in a program problem solution; and D.2.F
- G use object-oriented programming development methodology, including data abstraction, encapsulation with information hiding, inheritance, and procedural abstraction in program development. D.2.G

3 **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:** D.3

- A use the principles of software development to work in software design teams; D.3.A
- B break a problem statement into specific solution requirements; D.3.B
- C create a program development plan; D.3.C
- D code part of a solution from a program development plan while a partner codes the remaining part; D.3.D
- E collaborate with a team to test a solution, including boundary and standard cases; and D.3.E
- F develop presentations to report the solution findings. D.3.F

4 **Data literacy and management. The student locates, analyzes, processes, and organizes data. The student is expected to:** D.4

- A use programming file structure and file access for required resources; D.4.A
- B acquire and process information from text files, including files of known and unknown sizes; D.4.B
- C manipulate data using string processing; D.4.C
- D manipulate data values by casting between data types; D.4.D
- E use the structured data type of one-dimensional arrays to traverse, search, modify, insert, and delete data; D.4.E
- F identify and use the structured data type of two-dimensional arrays to traverse, search, modify, insert, and delete data; D.4.F
- G identify and use a list object data structure to traverse, search, insert, and delete data; and D.4.G
- H differentiate between categories of programming languages, including machine, assembly, high-level compiled, high-level interpreted, and scripted. D.4.H

5 Critical thinking, problem solving, and decision making. The student uses appropriate strategies to analyze problems and design algorithms. The student is expected to: D.5

- A develop sequential algorithms using branching control statements, including nested structures, to create solutions to decision-making problems; D.5.A
- B develop choice algorithms using selection control statements based on ordinal values; D.5.B
- C demonstrate the appropriate use of short-circuit evaluation in certain situations; D.5.C
- D use Boolean algebra, including De Morgan's Law, to evaluate and simplify logical expressions; D.5.D
- E develop iterative algorithms using nested loops; D.5.E
- F identify, trace, and appropriately use recursion in programming solutions, including algebraic computations; D.5.F
- G trace, construct, evaluate, and compare search algorithms, including linear searching and binary searching; D.5.G
- H identify, describe, trace, evaluate, and compare standard sorting algorithms, including selection sort, bubble sort, insertion sort, and merge sort; D.5.H
- I measure time and space efficiency of various sorting algorithms, including analyzing algorithms using "big-O" notation for best, average, and worst-case data patterns; D.5.I
- J develop algorithms to solve various problems such as factoring, summing a series, finding the roots of a quadratic equation, and generating Fibonacci numbers; D.5.J
- K test program solutions by investigating boundary conditions; testing classes, methods, and libraries in isolation; and performing stepwise refinement; D.5.K
- L identify and debug compile, syntax, runtime, and logic errors; D.5.L
- M compare efficiency of search and sort algorithms by using informal runtime comparisons, exact calculation of statement execution counts, and theoretical efficiency values using "big-O" notation, including worst-case, best-case, and average-case time/space analysis; D.5.M
- N count, convert, and perform mathematical operations in the decimal, binary, octal, and hexadecimal number systems; D.5.N
- O identify maximum integer boundary, minimum integer boundary, imprecision of real number representations, and round-off errors; D.5.O
- P create program solutions to problems using a mathematics library; D.5.P
- Q use random number generator algorithms to create simulations; D.5.Q
- R use composition and inheritance relationships to identify and create class definitions and relationships; D.5.R
- S explain and use object relationships between defined classes, abstract classes, and interfaces; D.5.S

T create object-oriented class definitions and declarations using variables, constants, methods, parameters, and interface implementations; D.5.T

U create adaptive behaviors using polymorphism; D.5.U

V use reference variables for object and string data types; D.5.V

W use value and reference parameters appropriately in method definitions and method calls; D.5.W

X implement access scope modifiers; D.5.X

Y use object comparison for content quality; D.5.Y

Z duplicate objects using the appropriate deep or shallow copy; D.5.Z

AA apply functional decomposition to a program solution; D.5.AA

BB create objects from class definitions through instantiation; and D.5.BB

CC examine and mutate the properties of an object using accessors and modifiers. D.5.CC