Texas University Interscholastic League

Contest Event: Computer Science

The contest challenges high school students to gain an understanding of the significance of computation as well as the details of Java programming, to be alert to new technology and information, to gain an understanding of the basic principles of computer science, and to give students a start in one of the most important fields of the Information Age.

The Texas Essential Knowledge and Skills (TEKS) are categorized by course area and grade level. The following are course area abbreviations used for the TEKS in Computer Science:

Computer Science 1 = CS1;
Computer Science 2 = CS2.

Each TEKS begins with the outline number for the appropriate course area.

Texas Essential Knowledge and Skills	**Contest Knowledge and Skills**
Basic Programming and PC Skills: 1A. Demonstrate knowledge and appropriate use of operating systems, software applications, and communication and networking components. (CS1) 1C. Make decisions regarding the selection, acquisition, and use of software taking under consideration its quality, appropriateness, effectiveness, and efficiency. (CS1) 1D. Delineate and make necessary adjustments regarding compatibility issues including, but not limited to, digital file formats and cross platform connectivity. (CS1) 1E. Differentiate current programming languages, discuss the use of the languages in other fields of study, and demonstrate knowledge of specific programming terminology and concepts. (CS1) 1F. Differentiate among the levels of programming languages including machine, assembly, high-level compiled and interpreted languages. (CS1) 1G. Demonstrate coding proficiency in a contemporary programming language. (CS1) 4A. Use local area networks (LANs) and wide area networks (WANs), including the Internet and intranet, in research and resource sharing. (CS1) 3B. Demonstrate proper etiquette and knowledge of acceptable use policies when using networks, especially resources on the Internet and intranet. (CS1/CS2)	Basic Programming and PC Skills: -Use an operating system. -Use a programming language. -Use an editor. -Use a compiler. -Work as a team.
Number Systems Knowledge and Skills: 1F. Differentiate among the levels of programming languages including machine, assembly, high-level compiled and interpreted languages. (CS1)	Number Systems Knowledge and Skills: -Perform Base Conversions and Arithmetic. -Use Bitwise operators (<<, >>, >>>, &, ~, |, ^).

Basic Input Knowledge and Skills: 1B. Compare, contrast, and appropriately use the various input, processing, output, and primary/secondary storage devices. (CS1) 2A. Demonstrate proficiency in the use of a variety of input devices such as keyboard. (CS1/CS2) 2B. Use digital keyboarding standards for the input of data. (CS1/CS2)	Basic Input Knowledge and Skills: - Use Java Standard Library. - Use Scanner to perform basic input. - Use System.out.print(). - Use System.out.println(). - Use System.out.printf(). - Use Escape sequences (\" \\ \n \' \t). - Use String.split(). - Use Integer.parseInt(). - Use Double.parseDouble()). - Use Exceptions, throwing standard unchecked exceptions, checked exceptions (try/catch/finally, throw, throws).
Data Types Knowledge and Skills: 7A. Apply problem-solving strategies such as design specifications, modular top-down design, step-wise refinement, or algorithm development. (CS1) 7D. Code using various data types. (CS1) 7E. Demonstrate effective use of predefined input and output procedures for lists of computer instructions including procedures to protect from invalid input. (CS1) 7F. Develop coding with correct and efficient use of expressions and assignment statements including the use of standard/user-defined functions, data structures, operators/proper operator precedence, and sequential/conditional/repetitive control structures. (CS1) 7G. Create and use libraries of generic modular code to be used for efficient programming; 7H. Identify actual and formal parameters and use value and reference parameters. (CS1) 9B. Use correct programming style to enhance the readability and functionality of the code such as spacing, descriptive identifiers, comments, or documentation. (CS1)	Data Types Knowledge and Skills: - Use System.out.print(). - Use System.out.println(). - Use System.out.printf(). - Use Primitive types (int, double, boolean, short, long, byte, char, float), casting of primitives, autoboxing/unboxing. - Use Arithmetic operators (+, -, *, /, %, ++, --) and string concatenation. - Use ++, --. - Use assignment operators (=, +=, -=, *=, /=, %=). - Convert to supertypes and (Subtype) casts. - Use instanceof. - Compare references with == and !=. - Compare reference contents using equals() and compareTo(). - Use the Java Standard Library Classes (String,

	StringBuffer, Integer, Double, Character, Math, Random, Object, Comparable, Exception, Scanner).
Conditional Knowledge and Skills: 7A. Apply problem-solving strategies such as design specifications, modular top-down design, step-wise refinement, or algorithm development. (CS1) 7F. Develop coding with correct and efficient use of expressions and assignment statements including the use of standard/user-defined functions, data structures, operators/proper operator precedence, and sequential/conditional/repetitive control structures. (CS1) 7I. Use control structures such as conditional statements and iterated, pretest, and posttest loops. (CS1) 7J. Use sequential, conditional, selection, and repetition execution control structures such as menu-driven programs that branch and allow user input. (CS1)	Conditional Knowledge and Skills: - Use if statements. - Use if/else statements. - Use ternary statements(?:). - Use switch case statements. - Use break and continue. - Use System.out.print(). - Use System.out.println(). - Use System.out.printf(). - Use boolean expressions and operators. - Use (==, !=, <, <=, >, >=, &&, ||, !, &, |). - Use and recognize short circuit evaluation.
Looping Knowledge and Skills: 7A. Apply problem-solving strategies such as design specifications, modular top-down design, step-wise refinement, or algorithm development. (CS1) 7E. Demonstrate effective use of predefined input and output procedures for lists of computer instructions including procedures to protect from invalid input. (CS1) 7F. Develop coding with correct and efficient use of expressions and assignment statements including the use of standard/user-defined functions, data structures, operators/proper operator precedence, and sequential/conditional/repetitive control structures. (CS1) 7G. Create and use libraries of generic modular code to be used for efficient programming; 7I. Use control structures such as conditional statements and iterated, pretest, and posttest loops. (CS1) 7J. Use sequential, conditional, selection, and repetition execution control structures such as menu-driven programs that branch and allow user input. (CS1)	Looping Knowledge and Skills: - Use for loops. - Use while loops. - Use do while loops. - Use nested loops. - Use the enhanced for loop. - Use break and continue. - Use System.out.print(). - Use System.out.println(). - Use System.out.printf(). - Use boolean expressions and operators. - Use (==, !=, <, <=, >, >=, &&, ||, !, &, |). - Use and recognize short circuit evaluation.

<u>Parameters Knowledge and Skills:</u> 7H. Identify actual and formal parameters and use value and reference parameters. (CS1)	<u>Parameters Knowledge and Skills:</u> - Pass primitive types as parameters. - Pass references as parameters.
<u>Arrays Knowledge and Skills:</u> 7A. Apply problem-solving strategies such as design specifications, modular top-down design, step-wise refinement, or algorithm development. (CS1) 7C. Develop sequential and iterative algorithms and codes programs in prevailing computer languages to solve practical problems modeled from school and community. (CS1) 7D. Code using various data types. (CS1) 7F. Develop coding with correct and efficient use of expressions and assignment statements including the use of standard/user-defined functions, data structures, operators/proper operator precedence, and sequential/conditional/repetitive control structures. (CS1) 7G. Create and use libraries of generic modular code to be used for efficient programming. (CS1) 7I. Use control structures such as conditional statements and iterated, pretest, and posttest loops. (CS1) 7J. Use sequential, conditional, selection, and repetition execution control structures such as menu-driven programs that branch and allow user input. (CS1) 7K. Identify and use structured data types of one-dimensional arrays, records, and text files. (CS1)	<u>Arrays Knowledge and Skills:</u> - Use Primitive types (int, double, boolean, short, long, byte, char, float), casting of primitives, autoboxing/unboxing. - Use one dimensional arrays. - Use arrays of references. - Use matrices(arrays of arrays). - Use initialized arrays. - Convert to supertypes and (Subtype) casts. - Use instanceof. - Compare references with == and !=. - Compare reference contents using equals() and compareTo(). - Use Java Collections(Collection, List, Set, Map, Map.Entry, ArrayList, LinkedList, HashSet, TreeSet, HashMap, TreeMap, Iterator, ListIterator). - Use Generic Java Collections(Collection, List, Set, Map, Map.Entry, ArrayList, LinkedList, HashSet, TreeSet, HashMap, TreeMap, Iterator, ListIterator). - Use Arrays class - Arrays.sort(). -Use Collections class - Collections.sort().
<u>Object Oriented Knowledge and Skills:</u> 1A. Identify object-oriented data types and delineate the advantages/disadvantages of object data. (CS2) 1B. Demonstrate coding proficiency in contemporary programming languages including	<u>Object Oriented Knowledge and Skills:</u> - Recognize and create classes (constructors, methods, instance variables, private vs. public, overloading, overriding, final local variables, static final class variables, static methods, static non-final variables).

an object-oriented language. (CS2) 1C. Survey the issues accompanying the development of large software systems such as design/implementation teams, software validation/testing, and risk assessment. (CS2)	- Use constructors. - Recognize the initialization of static variables, default initialization of instance variables, static initialization blocks. - Use and recognize inheritance, abstract classes, interfaces and polymorphism. - Use and recognize null, this, super, super.method(args), super(args), this.var, this.method(args), this(args).
Sorting and Searching Knowledge and Skills: 4A. Construct search algorithms including linear and binary searches. (CS2) 4B. Compare and contrast search and sort algorithms including linear and binary searches for different purposes and search time. (CS2) 7D. Identify, describe, and use sequential/non-sequential files; multidimensional arrays and arrays of records; and quadratic sort algorithms such as selection, bubble, or insertion, and more efficient algorithms including merge, shell, and quick sorts. (CS2) 7E. Create robust programs with increased emphasis on design, style, clarity of expression and documentation for ease of maintenance, program expansion, reliability, and validity. (CS2) 7F. Apply methods for computing iterative approximations and statistical algorithms. (CS2)	Sorting and Searching Knowledge and Skills: -Use and recognize all sorts (Selection, Insertion, Mergesort, Quicksort). - Analyze algorithms: informal comparison of running times, exact calculation of statement execution counts.
Recursion Knowledge and Skills: 7A. Use appropriately and trace recursion in program design comparing invariant, iterative, and recursive algorithms. (CS2) 7J. Use depth-first/breadth-first and heuristic search strategies. (CS2)	Recursion Knowledge and Skills: -Determine the output of recursive methods. -Write recursive methods.
String Manipulation Knowledge and Skills: 7B. Manipulate data structures using string processing. (CS2)	String Manipulation Knowledge and Skills: - Use String.split(). - Use Integer.parseInt(). - Use Double.parseDouble()). - Use and recognize regex (. + * \d \D \s \S \w \W [abc] [^abc] [a-zA-Z]). - Use Exceptions, throwing standard unchecked exceptions, checked exceptions (try/catch/finally,

	throw, throws). - Use the Java Standard Library Classes (String, StringBuffer, Integer, Double, Character, Math, Random, Object, Comparable, Exception, Scanner).
<u>Abstract Data Types Knowledge and Skills:</u> 7G. Define and develop code using the concepts of abstract data types including stacks, queues, linked lists, trees, graphs, and information hiding. (CS2) 7H. Identify and describe the correctness and complexity of algorithms such as divide and conquer, backtracking, or greedy algorithms. (CS2)	<u>Abstract Data Types Knowledge and Skills:</u> - Use and recognize Stacks, Queues, Binary Trees, Linked Lists, Heaps, Hash Tables, Priority Queues. - Use Java Collections(Collection, List, Set, Map, Map.Entry, ArrayList, LinkedList, HashSet, TreeSet, HashMap, TreeMap, Iterator, ListIterator). - Use Generic Java Collections(Collection, List, Set, Map, Map.Entry, ArrayList, LinkedList, HashSet, TreeSet, HashMap, TreeMap, Iterator, ListIterator). - Use Arrays class - Arrays.sort(). - Use Collections class - Collections.sort()
<u>Analysis of Algorithms Knowledge and Skills:</u> 9A. Demonstrate the ability to read and modify large programs including the design description and process development. (CS2) 9B. Analyze algorithms using "big-O" notation, best, average, and worst case space techniques. (CS2) 9C. Compare and contrast design methodologies including top-down and bottom-up. (CS2) 9D. Analyze models used in development of software including software life cycle models, design objectives, documentation, and support. (CS2)	<u>Analysis of Algorithms Knowledge and Skills:</u> -Use and recognize all sorts (Selection, Insertion, Mergesort, Quicksort). - Analyze algorithms: informal comparison of running times, exact calculation of statement execution counts. - Recognize and determine Big-O notation, worst case/average case time and space analysis.
<u>Hands On Section Knowledge and Skills:</u> 5B. Use a variety of resources, including foundation and enrichment curricula, together with various productivity tools to gather authentic data as a basis for individual and group programming projects. (CS1/CS2) 6A. Determine and employ methods to evaluate the design and functionality of the process using effective coding, design, and test data. (CS1/CS2) 6B. Implement methods for the evaluation of the	<u>Hands On Section Knowledge and Skills:</u> -Work as a team to solve problems. -Share one PC in a timed environment. -Work together to create solutions. -Work together to repair/improve code.

information using defined rubrics. (CS1/CS2) 8C. Extend the learning environment beyond the school walls with digital products created to increase teaching and learning in the foundation and enrichment curricula. (CS1/CS2) 12A. Write technology specifications for planning/evaluation rubrics documenting variables, prompts, and programming code internally and externally. (CS1/CS2) 12B. Seek and respond to advice from peers and professionals in evaluating the product (CS1/CS2) 12C. Debug and solve problems using reference materials and effective strategies (CS1/CS2)	

Appendix – Examples of Computer Science Reading List

Big Java, by Cay Horstmann.

How to Prepare for the AP Computer Science Exam (Barron's Review), by Roselyn Teukolsky.

Java: How to Program, by Deitel & Deitel.

Java Language Specification, by Gosling et al.

C**lassroom textbook.**