Study Companion

Computer Science Assessment

For the most up-to-date information, visit the ETS GACE website at **gace.ets.org**.

Last Updated: June 2018

This publication has been produced for the GaPSC by ETS. ETS is under contract to the GaPSC to administer the Georgia Assessments for the Certification of Educators. The Georgia Assessments for the Certification of Educators are administered under the authority of the GaPSC; regulations and standards governing the program are subject to change without notice at the discretion of the GaPSC. The GaPSC and ETS are committed to preventing discrimination on the basis of race, color, national origin, sex, religion, age, or disability in the administration of the testing program or the provision of related services.

Table of Contents

Note: After clicking on a link, right click and select "Previous View" to go back to original text.

About the Assessment

Assessment Name	Computer Science
Grade Level	P–12
Test Code	555
Testing Time	3 hours
Test Duration	3.5 hours
Test Format	Computer delivered
Number of Selected-response Questions	100
Question Format	The test consists of a variety of short-answer questions such as selected-response questions, where you select one answer choice or multiple answer choices (depending on what the question asks for), questions where you enter your answer in a text box, and other types of questions. You can review the possible question types in the ***Guide to Taking a GACE Computer-delivered Test***.
Number of Constructed-response Questions	0

The GACE Computer Science assessment is designed to measure the professional knowledge of prospective teachers of computer science in the state of Georgia.

The testing time is the amount of time you will have to answer the questions on the test. Test duration includes time for tutorials and directional screens that may be included in the test.

The total number of questions that are scored is typically smaller than the total number of questions on the test. Most tests that contain selected-response questions also include embedded pretest questions, which are not used in calculating your score. By including pretest questions in the assessment, ETS is able to analyze actual test-taker performance on proposed new questions and determine whether they should be included in future versions of the test.

Note: After clicking on a link, right click and select "Previous View" to go back to original text.

Content Specifications

This assessment is organized into content **subareas**. Each subarea is further defined by a set of **objectives** and their **knowledge statements**.

- The objectives broadly define what an entry-level educator in this field in Georgia public schools should know and be able to do.
- The knowledge statements describe in greater detail the knowledge and skills eligible for testing.
- Some tests also include content material at the evidence level. This content serves as descriptors of what each knowledge statement encompasses.

See a breakdown of the subareas and objectives for this assessment on the following pages.

Note: After clicking on a link, right click and select "Previous View" to go back to original text.

Test Subareas

Subarea	Approx. Percentage of Test
I. Impacts of Computing	15%
II. Algorithms and Computational Thinking	25%
III. Programming	30%
IV. Data	15%
V. Computing Systems and Networks	15%

Test Objectives

Subarea I: Impacts of Computing

Objective 1: Understands and applies knowledge of impact of, obstacles to, and effects of computing

The beginning Computer Science teacher:

A. Understands computing as a way of expressing creativity, solving problems, enabling communication, and fostering innovation in a variety of fields and careers
   - Recognizes that computers can be used to showcase creativity
   - Recognizes the benefits of using computers to solve problems
   - Provides examples of how computers enable communication and collaboration
   - Provides examples of how computers foster innovation

B. Knows the obstacles to equal access to computing among different groups and the impact of those obstacles
   - Identifies obstacles to equal access to computing among different groups (e.g., groups defined by gender, socioeconomic status, disability/accessibility needs) and the impact of those obstacles
   - Identifies factors that contribute to the digital divide
   - Matches obstacles to equal access with effective solutions

C. Understands beneficial and harmful effects of computing innovations and the trade-offs between them
   - Analyzes computing innovations in terms of their social, economic, and cultural impacts, both beneficial and harmful
   - Identifies trade-offs between beneficial and harmful effects of computer innovations

Note: After clicking on a link, right click and select "Previous View" to go back to original text.

Objective 2: Understands and applies knowledge of issues regarding intellectual property, ethics, privacy, and security in computing

The beginning Computer Science teacher:

A. Knows different methods of protecting intellectual property rights and the trade-offs between them in a variety of contexts (e.g., Creative Commons, open source, copyright)
   - Using correct vocabulary, describes how different methods of protecting intellectual property rights work
   - Given a context, identifies appropriate methods of protecting intellectual property rights
   - Identifies and compares trade-offs between different methods of protecting intellectual property rights

B. Understands ethical and unethical computing practices and their social, economic, and cultural implications
   - Identifies ethical and unethical computing practices in context
   - Describes the social, economic, and cultural implications of ethical and unethical computing practices
   - Identifies the conditions under which a given computing practice is ethical or legal

C. Knows privacy and security issues regarding the acquisition, use, and disclosure of information in a digital world
   - Using correct vocabulary, describes privacy and security issues
   - In context, identifies appropriate strategies to safeguard privacy and ensure security
   - Describes trade-offs between local and cloud-based data storage
   - Identifies methods that digital services use to collect information about users

Subarea II: Algorithms and Computational Thinking

Objective 1: Understands and applies knowledge of abstraction, pattern recognition, problem decomposition, number base conversion, and algorithm formats

The beginning Computer Science teacher:

A. Understands abstraction as a foundation of computer science
   - Identifies, creates, or completes the correct ordering, from low to high, of an abstraction hierarchy
   - Identifies abstractions in context
   - Identifies details that can be removed from a solution in order to generalize it

B. Knows how to use pattern recognition, problem decomposition, and abstraction to develop an algorithm
   - Given a table of values or other data source, identifies the patterns in the data and identifies algorithms that could produce the patterns

- Identifies components that could be part of an algorithm to solve a problem
- Identifies actions and actors when decomposing a problem
- Identifies appropriate decomposition strategies

C. Understands number base conversion and binary, decimal, and hexadecimal number systems
   - Converts between number bases
   - Analyzes and compares representations of numbers in different bases

D. Understands how to develop and analyze algorithms expressed in multiple formats (e.g., natural language, flowcharts, pseudocode)
   - Interprets diagrams that describe algorithms, given an explanation of the symbols used
   - Compares algorithms written in multiple formats
   - Traces and analyzes algorithms written in different formats
   - Identifies correct sequencing of steps in an algorithm and errors in sequencing

Objective 2: Understands and applies knowledge of algorithm analysis, searching and sorting algorithms, recursive algorithms, and randomization

The beginning Computer Science teacher:

A. Is familiar with the limitations of computing in terms of time, space, and solvability as well as with the use of heuristic solutions that can address these limitations
   - Identifies and compares algorithms that are linear, quadratic, exponential, or logarithmic
   - Recognizes the existence of problems that cannot be solved by a computer
   - In context, identifies factors that prevent a problem from being solvable
   - Identifies situations where heuristic solutions are useful
   - In context, identifies space and time limitations of computational solutions to problems

B. Understands searching and sorting algorithms; can analyze sorting algorithms for correctness and can analyze searching algorithms for correctness and efficiency
   - Traces algorithms and predicts output and intermediate results
   - Calculates the number of comparisons required for linear and binary search algorithms

C. Understands simple recursive algorithms (e.g., *n* factorial, sum of first *n* integers)
   - Traces simple recursive algorithms
   - Provides missing steps in incomplete simple recursive algorithms
   - Identifies parts of a recursive algorithm (e.g., base or stopping condition, recursive call)

- Identifies errors in simple recursive algorithms
- Identifies an iterative algorithm that is equivalent to a recursive algorithm

D. Is familiar with the use of randomization in computing
   - Identifies appropriate uses of randomization in a variety of applications
   - Identifies the difference between random and pseudorandom numbers

Subarea III: Programming

Objective 1: Understands and applies knowledge of programming control structures, standard operators, variables, correctness, extensibility, modifiability, and reusability

The beginning Computer Science teacher:

A. Understands how to write and modify computer programs in a text-based programming language
   - Describes what a program does or is able to choose the code segment that correctly implements a given intended purpose
   - Identifies missing code in a code segment with a stated intended purpose
   - Places statements in appropriate order to create a correct program
   - Identifies how changing one part of a code segment will affect the output

B. Understands how to analyze computer programs in terms of correctness
   - Traces code and indicates the output printed or the value of variables after code segment execution
   - Indicates the inputs that produce given outputs for a code segment
   - Describes what a program does or chooses the code segment that correctly implements a given intended purpose
   - Identifies valid preconditions and postconditions
   - Compares two code segments or algorithms
   - Identifies the type of error produced by a code segment (i.e., syntax, runtime, compile-time, overflow, round-off, logic)
   - Identifies errors in incorrect code and changes that can be made to correct them

C. Knows the concepts of extensibility, modifiability, and reusability
   - Identifies the meaning of the terms
   - Identifies functionally equivalent statements or code segments that differ in one of these three ways
   - Identifies situations where the use of constants or variables would be preferred over hard-coded values
   - Identifies opportunities for parameterization

- Chooses code that improves on given code by making it more extensible, modifiable, or reusable
- Identifies changes that would improve a given code segment

D. Understands the three basic constructs used in programming: sequence, selection, and iteration

- Traces code and indicates the output printed or the value of variables after code segment execution
- Indicates inputs that produce given outputs for a code segment
- Describes what a program does or chooses the code segment that correctly implements a given intended purpose
- Identifies missing code in a code segment with a stated intended purpose
- Identifies equivalent statements or code segments
- Identifies the three constructs when used in code
- Identifies which of the constructs are needed to implement given functionality
- Converts code that does not use iteration to equivalent code that uses iteration

E. Understands how to use standard operators (i.e., assignment, arithmetic, relational, logical) and operator precedence to write programs

- Traces code and indicates the output displayed or the value of variables after code segment execution
- Indicates inputs that produce given outputs for a code segment
- Describes what a program does or chooses the code segment that correctly implements a stated intended purpose
- Identifies missing code in a code segment with a stated intended purpose
- Identifies equivalent statements or code segments
- Places statements in appropriate order to create a correct program
- Uses Boolean algebra to identify equivalent Boolean expressions
- Writes a Boolean expression equivalent to a given code, or identifies code equivalent to a given Boolean expression or English description
- Identifies the correct implementation of a given formula, including formulas with fractions
- Evaluates expressions that include arithmetic operations

F. Understands how to use variables and a variety of data types

- Identifies variables and data types (e.g., integers, floating point, string, Booleans, arrays/lists)
- Identifies the need for type conversion
- Traces code and indicates the output printed or the value of variables after code segment execution

Note: After clicking on a link, right click and select "Previous View" to go back to original text.

- Indicates the inputs that produce given outputs for a code segment
- Describes what a program does or chooses the code segment that correctly implements a stated intended purpose
- Identifies missing code in a code segment with a stated intended purpose
- Identifies equivalent statements or code segments
- Places statements in appropriate order to creates a correct program
- Describes the difference between integer and floating point numeric data types
- Describes the difference between integer and floating point division
- Describes the benefits of the use of each data type
- Distinguishes between global and local scope
- Identifies the most appropriate data type in a given context
- Identifies the correct sequence of string operations to produce a given output

Objective 2: Understands and applies knowledge of procedures, event-driven programs, usability, data structures, debugging, documenting and reviewing code, libraries and APIs, IDEs, and programming language paradigms, including object-oriented concepts

The beginning Computer Science teacher:

A. Understands how to write and call procedures with parameters and return values
   - Traces code and indicates the output printed or the value of variables after code segment execution
   - Indicates inputs that produce given outputs for a code segment
   - Describes what a program does or chooses the code segment that correctly implements a stated intended purpose
   - Identifies missing code in a code segment with a stated intended purpose
   - Identifies equivalent statements or code segments
   - Places statements in appropriate order to create a correct program
   - Traces code when references to objects and arrays are passed to procedures
   - Traces code that includes nested procedure calls

B. Knows the concepts of event-driven programs that respond to external events (e.g., sensors, messages, clicks)
   - Traces code and indicates the output printed or the value of variables after code segment execution
   - Indicates inputs that produce given outputs for a code segment
   - Describes what a program does or chooses the code segment that correctly implements a stated intended purpose
   - Identifies missing code in a code segment with a stated intended purpose

- Identifies possible errors due to asynchronous events
- Identifies aspects of concurrency in event-driven programming

C. Is familiar with usability and user experience (e.g., ease of use and accessibility)
   - Identifies code that improves on given code in terms of usability or user experience
   - Identifies meaningful error messages
   - Identifies features that improve accessibility

D. Is familiar with dictionaries/maps, stacks, and queues
   - Identifies a data structure based on a description of behavior or appropriate use
   - Given goals, constraints, or context, identifies the most appropriate data structure
   - Traces code that uses a particular data structure

E. Understands how to use debugging techniques and appropriate test cases
   - Identifies which test cases are most useful for given code
   - Differentiates between different types of errors (e.g., overflow, round-off, syntax, runtime, compile-time, logic)
   - Describes useful debugging techniques (e.g., where to put print statements)
   - Differentiates between empirical testing and proof
   - Identifies errors in code and solutions to those errors

F. Is familiar with characteristics of well-documented computer programs that are usable, readable, and modular
   - Identifies characteristics of good documentation
   - Identifies good and poor documentation practices in context

G. Is familiar with techniques to obtain and use feedback to produce high-quality code (e.g., code reviews, peer feedback, end user feedback)
   - Identifies situations in which each of the three listed techniques are useful

H. Knows how to use libraries and APIs
   - Identifies correct call(s) and use of return values given an API definition
   - Identifies reasons to use or not use libraries in place of writing original code
   - Identifies applications (e.g., math libraries, random number generation) that use APIs

I. Understands programming techniques to validate correct input and detect incorrect input
   - Identifies effective input data validation strategies
   - Compares data validation (proper range and format) and data verification (e.g., password verification)
   - Identifies improvements to code for which data validation is required

Note: After clicking on a link, right click and select "Previous View" to go back to original text.

J. Is familiar with the features and capabilities of integrated development environments (IDEs)
   - Identifies components of IDEs
   - Identifies benefits and drawbacks of using IDEs
   - Identifies the costs and benefits of context editors

K. Is familiar with the differences between low- and high-level programming languages
   - Identifies characteristics of low- and high-level languages

L. Is familiar with different programming paradigms
   - Identifies the terminology of procedural programming
   - Identifies the terminology of object-oriented programming
   - Compares programming paradigms

M. Knows object-oriented programming concepts
   - Identifies classes, instance variables, and methods given a diagram
   - Identifies the benefits of inheritance and encapsulation
   - Identifies distinctions between overloading and overriding

N. Is familiar with program compilation and program interpretation
   - Identifies differences between compilation and interpretation
   - Identifies differences between source code and object code

Subarea IV: Data

Objective 1: Understands and applies knowledge of digitalization, data encryption and decryption, and computational tools

The beginning Computer Science teacher:

A. Understands bits as the universal medium for expressing digital information
   - Performs calculations, using bits and bytes
   - Determines the number of bits and bytes required to store a given amount of data
   - Given the description of an encoding scheme, encodes or decodes data
   - Describes lossy and lossless data compression
   - Explains why binary numbers are fundamental to the operation of computer systems

B. Is familiar with concepts of data encryption and decryption
   - Distinguishes between encoding and encryption
   - Identifies trade-offs in the use of data encryption

C. Knows how to use computational tools, including spreadsheets, to analyze data in order to discover, explain, and visualize patterns, connections, and trends

- Transforms data to make it more useful
- Identifies specific data or characteristics of specific data that need to be removed or modified before an entire data set can be used
- Describes the use of spreadsheet operations (e.g., formulas, filters, sorts, charts, graphs) to analyze and visualize data

Objective 2: Understands and applies knowledge of simulation, modeling, and manipulation of data

The beginning Computer Science teacher:

A. Is familiar with the use of computing in simulation and modeling

- Describes questions that can be answered with a given simulation, or explains what data and process are required in a simulation in order to answer a given question
- Traces code in a simulation context
- Identifies missing code in a simulation context
- Identifies the impact of changes to simulations (e.g., more or fewer variables, more or less data)
- Identifies applications of simulation and modeling

B. Is familiar with methods to store, manage, and manipulate data

- Uses terminology and concepts of files and databases
- Identifies measures of file size (e.g., byte, kilo, mega, giga, tera, peta)
- Identifies issues connected with the storage requirements of computing applications, including scale, redundancy, and backup

C. Is familiar with a variety of computational methods for data collection, aggregation, and generation

- Identifies the benefits of working with publicly available data sets
- Identifies the types of data generated by surveys and sensors
- Identifies examples of crowdsourcing and citizen science
- Identifies appropriate data-collection methods for a given context and purpose

Subarea V: Computing Systems and Networks

Objective 1: Understands and applies knowledge of operating systems, computing systems, communication between devices, and cloud computing

The beginning Computer Science teacher:

A. Knows that operating systems are programs that control and coordinate interactions between hardware and software components
   - Identifies hardware components and their functions
   - Identifies software components and their functions
   - Identifies common operating systems tasks
   - Identifies resource issues that have an impact on functionality

B. Is familiar with computing systems embedded in everyday objects (e.g., Internet of Things [IoT], ATMs, medical devices)
   - Describes what an embedded system is
   - Defines what the IoT is and how it is used
   - Describes how sensors are used in embedded systems

C. Knows the capabilities, features, and uses of different types of computing systems (e.g., desktop, mobile, cluster)
   - Identifies capabilities, features, and uses for each type of computer system
   - Identifies criteria to evaluate and compare computing systems

D. Is familiar with computers as layers of abstraction from hardware (e.g., logic gates, chips) to software (e.g., system software, applications)
   - Identifes appropriate abstraction layers for hardware and software components

E. Is familiar with the steps required to execute a computer program (fetch-decode-execute cycles)
   - Describes what happens during fetch, decode, and execute, including the order of the steps in the cycle

F. Is familiar with trade-offs between local, network, and cloud computing and storage
   - Identifies advantages and disadvantages in terms of performance, cost, security, reliability, and collaboration
   - Identifies means of storing binary data

G. Is familiar with communication between devices
   - Identifies and compares wireless communication systems
   - Identifies and compares wired communication systems
   - Identifies and compares network types

Objective 2: Understands and applies knowledge of networks, including security issues and the Web

The beginning Computer Science teacher:

A. Knows components of networks
   - Identifies network hardware devices and their functions
   - Describes possible abstraction models of networks

B. Is familiar with factors that have an impact on network functionality
   - Defines basic terminology (e.g., bandwidth, load, latency)
   - Estimates necessary bandwidth and data size for a given situation
   - Identifies critical resources for a given situation

C. Is familiar with how Internet and Web protocols work
   - Describes the purpose of protocols and identifies common Internet and Web protocols
   - Compares IPv4 and IPv6
   - Identifies and describes the basic parts of a URL (e.g., protocol, subdomain, domain name, port, path)
   - Describes the hierarchical structure of names in the domain name system (DNS)
   - Describes the purpose and function of IP addressing
   - Identifies how Internet protocols address reliability, redundancy, and error handling

D. Is familiar with digital and physical strategies for maintaining security
   - Identifies characteristics of strong passwords (e.g., length, bits per character)
   - Identifies digital and physical security strategies
   - Identifies trade-offs in the use of security measures (e.g., encryption, decryption, digital signatures and certificates)

E. Is familiar with concepts of cybersecurity
   - Identifies and defines the five pillars of cybersecurity: confidentiality, integrity, availability, nonrepudiation, and authentication

F. Is familiar with the components that make up the Web (e.g., HTTP, HTML, browsers, servers, clients)
   - Identifies the uses of markup languages
   - Identifies the purposes of browsers, servers, and clients

Note: After clicking on a link, right click and select "Previous View" to go back to original text.

Code Segments

Some stimulus material contains code segments written in pseudocode. The notation used in the pseudocode is described below

Pseudocode Notation

Explanation	Notation
Assignment operator	←
Arithmetic operators	`+ - / * ^ %` Note that `/` indicates floating point division unless stated otherwise.
Relational operators	`== < > ≤ ≥ ≠`
Logical operators	**`and`** **`or`** **`not`**
String concatenation operator	`+`
Boolean values	**`true`** **`false`**
Null	**`null`**
Comments	`// this is a single-line comment`

Placeholder for missing code	For example,
	`/* missing code */`
	`/* missing condition */`

Print A comment is used where necessary to indicate if a line feed or blank is appended to the argument.	**`print`** `arg`
Data types	**`boolean`** **`char`** **`double`** **`float`** **`int`** **`int[]`** **`int[][]`** **`short`** **`String`**
Array initialization and reference	**`int[]`** `a ← {1, 2, 3}` **`int`** `b[0..2] ← {1, 2, 3}` **`int[][]`** `c` `a[0]`

Note: After clicking on a link, right click and select "Previous View" to go back to original text.

Explanation	Notation

Conditional statements: Indentation and **end if** statements are significant.
 	
Example:
`if ( x > 10 )`
`    print "big number"`
`else`
`    print "small number"`
`end if`
	if (condition)
	    block of statements
	end if
	
	if (condition)
	    block of statements
	else
	    another block of statements
	end if

Iterative statements: Indentation and **end** statements are significant.	**for** (initialization; condition; increment)
	    block of statements
	end for
	
	while (condition)
	    block of statements
	end while
	
	do
	    block of statements
	while (condition)
	
	repeat
	    block of statements
	until (condition)

Procedures: Indentation and **end** statements are significant.
 	
The return type is indicated in the procedure header and is based on the value returned by the procedure or is **void** if the procedure does not return a value.
	int procedureName (arg1,
	arg2,
	...)
	    block of statements
	    return value
	end procedureName
	
	void procedureName (arg1,
	arg2,
	...)
	    block of statements
	end procedureName

Classes	**class** className
	    variable declarations
	    procedures
	end class className

Object-oriented keywords	**extends** **new** **public** **private**

Note: After clicking on a link, right click and select "Previous View" to go back to original text.

Practice Questions

The practice questions in this study companion are designed to familiarize you with the types of questions you may see on the assessment. While they illustrate some of the formats and types of questions you will see on the test, your performance on these sample questions should not be viewed as a predictor of your performance on the actual test. Fundamentally, the most important component in ensuring your success is familiarity with the content that is covered on the assessment.

To respond to a practice question, choose one of the answer options listed. Be sure to read the directions carefully to ensure that you know what is required for each question. You may find it helpful to time yourself to simulate actual testing conditions. A correct answer and a rationale for each sample test question are in the section following the practice questions.

Keep in mind that the test you take at an actual administration will have different questions, although the proportion of questions in each subarea will be approximately the same. You should not expect the percentage of questions you answer correctly in these practice questions to be exactly the same as when you take the test at an actual administration, since numerous factors affect a person's performance in any given testing situation.

Note: After clicking on a link, right click and select "Previous View" to go back to original text.

Directions: Each of the questions or incomplete statements below is followed by suggested answers or completions. Select the best option or options in each case.

For the following question, select all the answer choices that apply.

1. A technology company has developed an application that allows users to combine a text file and a recording of a person speaking to create an accurate audio representation of the person speaking the text. Which of the following are possible uses of the application?

   Select all that apply.

   A. Audio versions of unrecorded speeches delivered by famous individuals could be produced from transcripts and recorded speeches delivered by the same individual.
   B. An audio file could be fabricated in which an individual seems to be uttering slanderous statements about another individual.
   C. Security systems based on voice activation could be circumvented by unauthorized users.

Answer and Rationale

2. A programmer uses code published online under a Creative Commons Attribution (CC BY) license in a commercial product. Which of the following best describes an acceptable use of the code?

   A. Copying code from the online source into the programmer's product without any other actions
   B. Copying code from the online source into the programmer's product and limiting the copied code to ten code lines
   C. Copying code from the online source into the programmer's product and changing all variable names
   D. Copying code from the online source into the programmer's product and crediting the original author in the manner indicated by the license

Answer and Rationale

Note: After clicking on a link, right click and select "Previous View" to go back to original text.

3. Which of the following best describes the primary way in which a distributed denial-of-service (DDoS) attack differs from a denial-of-service (DoS) attack?

   A. The goal of the attack
   B The number of computers being attacked
   C. The number of computers launching the attack
   D. The time period in which the attack occurs

Answer and Rationale

4. Consider the following list.

   - Assembly language
   - Block-based programming language
   - Logic gate
   - Machine language

   Which of the following arranges the list in order from highest level of abstraction to lowest level of abstraction?

   A. Block-based programming language, assembly language, machine language, logic gate
   B. Block-based programming language, machine language, assembly language, logic gate
   C. Block-based programming language, machine language, logic gate, assembly language
   D. Machine language, block-based programming language, assembly language, logic gate

Answer and Rationale

Note: After clicking on a link, right click and select "Previous View" to go back to original text.

5. Chloe is playing a game in which she rolls a pair of dice 6 times. Her initial score is 0, and after each roll her score is recalculated. The table shows the outcomes of each roll and Chloe's recalculated score.

Roll	1st	2nd	3rd	4th	5th	6th
Die One	3	1	6	1	6	2
Die Two	4	5	2	6	6	2
Score	7	13	13	13	0	4

Consider the following pseudocode segment.

```
int score ← 0
int dieOne ← 0
int dieTwo ← 0
for ( int i ← 0; i < 6; i ← i + 1 )
    dieOne ← getDieOneValue ()    // returns the value
                                  // of the first die
    dieTwo ← getDieTwoValue ()    // returns the value
                                  // of the second die
    score ← newScore ( dieOne, dieTwo, score )
end for
```

Based on the data in the table, which of the following correctly implements the `newScore` procedure?

A.
```
int newScore ( int diceOne, int diceTwo, int oldScore )
    return oldScore + diceOne + diceTwo
end newScore
```

B.
```
int newScore ( int diceOne, int diceTwo, int oldScore )
    oldScore ← oldScore + diceOne + diceTwo
    if ( ( diceOne == 6 ) or ( diceTwo == 6 ) )
        oldScore ← oldScore - diceOne - diceTwo
    else
        if ( ( diceOne == 6 ) and ( diceTwo == 6 ) )
            oldScore ← 0
        end if
    end if
    return oldScore
end newScore
```

Note: After clicking on a link, right click and select "Previous View" to go back to original text.

C.
```
int newScore ( int diceOne, int diceTwo, int oldScore )
    if ( ( diceOne ≠ 6 ) and ( diceTwo ≠ 6 ) )
        return oldScore + diceOne + diceTwo
    else
        if ( ( diceOne == 6 ) and ( diceTwo == 6 ) )
            return 0
        else
            return oldScore
        end if
    end if
end newScore
```

D.
```
int newScore ( int diceOne, int diceTwo, int oldScore )
    if ( diceOne == diceTwo )
        return 0
    else
        if ( ( diceOne == 6 ) or ( diceTwo == 6 ) )
            return oldScore
        else
            return oldScore + diceOne + diceTwo
        end if
    end if
end newScore
```

Answer and Rationale

Note: After clicking on a link, right click and select "Previous View" to go back to original text.

6. Which of the following is the hexadecimal representation of the decimal number 231_{10}?

   A. 17_{16}
   B. $E4_{16}$
   C. $E7_{16}$
   D. $F4_{16}$

Answer and Rationale

7. A sales representative has a list of clients to visit during an upcoming sales trip. No two clients live in the same city, and each client will be visited one time. The sales representative will return to the starting city at the end of the trip. To minimize travel expenses, a programmer at the sales representative's company has implemented an algorithm that generates all possible orderings of the clients' cities and then evaluates each ordering with respect to travel expenses. Which of the following best describes the running time of the algorithm in terms of the number of cities?

   A. Factorial
   B. Linear
   C. Logarithmic
   D. Quadratic

Answer and Rationale

Note: After clicking on a link, right click and select "Previous View" to go back to original text.

8. Consider the following pseudocode procedure, which sorts an integer array `arr` of length `len`. The first element of `arr` is at index `0`. A call `swap ( arr, i , j )` swaps the values of `arr[i]` and `arr[j]`.

```
void sort ( int[] arr, int len )
    int pos ← 0
    while ( pos < len )
        if ( pos == 0 )
            pos ← pos + 1
        else
            if ( arr[pos] > arr[pos - 1] )
                pos ← pos + 1
            else
                swap ( arr, pos, pos - 1 )
                pos ← pos - 1
            end if
        end if
    end while
end sort
```

If `arr` originally contains the values `{2, 1, 5, 3, 4}`, what will the values in `arr` be after 6 iterations of the **`while`** loop?

A. `{1, 2, 3, 4, 5}`
B. `{1, 2, 3, 5, 4}`
C. `{1, 2, 5, 3, 4}`
D. `{2, 1, 5, 3, 4}`

Answer and Rationale

Note: After clicking on a link, right click and select "Previous View" to go back to original text.

9. Consider the recursive pseudocode procedure `f,` which is intended to return the product of consecutive integers from `5` to `n,` inclusive, for `n ≥ 5` and to return zero otherwise. For example, `f ( 7 )` returns `210,` which is `5*6*7.`

```
int f ( int n )
    if ( n < 5 )
        return 0
    else
        if ( n == 5 )
            return 5
        else
            /* missing statement */
        end if
    end if
end f
```

Which of the following could replace `/* missing statement */` so that the procedure `f` works as intended?

A. `return n * f ( n - 1 )`
B. `return n * f ( n - 5 )`
C. `return ( n - 5 ) * f ( n )`
D. `return ( n - 5 ) + f ( n - 1 )`

Answer and Rationale

Note: After clicking on a link, right click and select "Previous View" to go back to original text.

10. Consider the pseudocode procedure `findMax,` which is intended to return the largest value in an integer array `numList` of length `n.` The first element of `numList` is at index `0`.

```
int findMax ( int[] numList, int n )
    int max ← numList[0]
    int i ← 1
    while ( i < n )
        if ( /* missing condition */ )
            max ← numList[i]
        end if
        i ← i + 1
    end while
    return max
end findMax
```

Which of the following could replace `/* missing condition */` so that `findMax` works as intended?

A. `numList[i] < n`

B. `numList[i] < max`

C. `numList[i] > max`

D. `numList[i] > numList[n - 1]`

Answer and Rationale

Note: After clicking on a link, right click and select "Previous View" to go back to original text.

11. Consider the following pseudocode procedure, which is intended to print the odd integers from 1 up to `n`.

```
// precondition: n is a positive integer
void f ( int n )
    for ( int c ← 1; c ≤ n;  c ← c + 1 )
        if ( ( c / 2 ) ≠ 0 )
            print c
        end if
    end for
end f
```

A. The procedure works correctly and prints all the odd integers from 1 up to `n`.

B. The procedure does not work correctly; a new variable needs to be declared inside the loop to test for odd numbers.

C. The procedure does not work correctly; the variable `c` needs to be declared before the **`for`** loop.

D. The procedure does not work correctly; there is a logic error in the **`if`** condition, which should say `c % 2` instead of `c / 2`.

Answer and Rationale

Note: After clicking on a link, right click and select "Previous View" to go back to original text.

12. Consider the following pseudocode procedure.

```
void mystery ( int n )
    while ( n ≠ 1 )
        if ( ( n % 2 ) == 1 )
            n ← 3 * n + 1
        else
            n ← n / 2
        end if
        print ( n )    // print a space after the number
    end while
end mystery
```

What is printed by the call `mystery ( 6 )` ?

A. `3 10 5 16 8 4 2`
B. `3 10 5 16 8 4 2 1`
C. `10 5 16 8 4 2 1`
D. Infinitely many numbers are printed because the while **`loop`** does not terminate.

Answer and Rationale

13. Consider the following pseudocode Boolean expression, where `age` and `height` are two properly declared and initialized integer variables.

```
( age > 10 ) and ( height > 36 )
```

Which of the following is an equivalent Boolean expression?

A. `not ( ( age < 10 ) and ( height < 36 ) )`
B. `not ( ( age ≤ 10 ) and ( height ≤ 36 ) )`
C. `not ( ( age < 10 ) or ( height < 36 ) )`
D. `not ( ( age ≤ 10 ) or ( height ≤ 36 ) )`

Answer and Rationale

Note: After clicking on a link, right click and select "Previous View" to go back to original text.

14. Consider a class **`String`** with the methods shown.

Procedure	Explanation
`String` `toLowerCase ()`	Converts the characters in the string to lowercase and returns the string.
`String` `toUpperCase ()`	Converts the characters in the string to uppercase and returns the string.
`String` `substring ( int begin, int end )`	Returns the substring of the string from position `begin` to position `end - 1` or returns `""` if `begin` is less than zero or `begin ≥ end` or `end` is greater than the length of the string. The first character in the string is at position `0`.
`int` `length ()`	Returns the length of the string.

Which of the following pseudocode segments would produce the output `"aPPLESAUCE"`?

A.
```
String value ← "Applesauce"
value ← value.substring ( 0, 1 ).toLowerCase () +
        value.substring ( 1, value.length () ).toUpperCase ()
print value
```

B.
```
String value ← "Applesauce"
value ← value.substring ( 0, 2 ).toLowerCase () +
        value.substring ( 1, value.length () ).toUpperCase ()
print value
```

C.
```
String value ← "Applesauce"
value ← value.substring ( 0, 1 ).toUpperCase () +
        value.substring ( 1, value.length () ).toLowerCase()
print value
```

D.
```
String value ← "Applesauce"
value ← value.toUpperCase ()
value ← value.toLowerCase ()
print value
```

Answer and Rationale

Note: After clicking on a link, right click and select "Previous View" to go back to original text.

15. Consider the following pseudocode procedure

```
int mystery ( int n )
    int temp ← 0
    for ( int c ← 1; c ≤ n; c ← c + 1 )
        if ( ( c % 3 ) == 0 )
            temp ← temp + c
        end if
    end for
    return temp
end mystery
```

Which of the following best describes procedure `mystery` ?

A. It returns a list of numbers from `1` to `n`.
B. It prints every third number from `1` to `n`.
C. It returns the sum of the numbers from `1` to `n`.
D. It returns the sum of the multiples of `3` from `1` to `n`.

Answer and Rationale

Note: After clicking on a link, right click and select "Previous View" to go back to original text.

16. Consider the following three scenarios, which could be modeled using three data structures—dictionary/map, queue, and stack.

    Scenario 1: Cars line up in a single lane at a car wash. As each driver reaches the entrance of the car wash, the driver gets out of the car. An automatic pulley moves the car through the car wash as the driver walks to the exit of the car wash to pay and pick up the car.

    Scenario 2: Contestants auditioning for a reality television talent show are assigned a unique numeric ID upon completing a registration form.

    Scenario 3: Tennis balls are sold in a cylindrical can that can hold a maximum of 3 balls, where each ball except the bottom one rests on top of the ball below it. The 3 balls are placed in the can one at a time through one opening at the top of the can. The 3 balls are removed from the can one at a time through the same opening.

    Which of the following shows the data structures that best model the scenarios?

	Scenario 1	**Scenario 2**	**Scenario 3**
A.	Dictionary/map	Queue	Stack
B.	Dictionary/map	Stack	Queue
C.	Queue	Dictionary/map	Stack
D.	Stack	Queue	Dictionary/map

Answer and Rationale

Note: After clicking on a link, right click and select "Previous View" to go back to original text.

17. Consider the following pseudocode segment with integer variables, which is intended to determine the largest value among variables `a, b,` and `c,` and to store that value in variable `max`. The code segment does not work as intended.

```
max ← -1
if ( a > b )
    if ( a > c )
        max ← a
    end if
else
    if ( b > c )
        max ← b
    else
        max ← c
    end if
end if
```

Which of the following test cases can be used to demonstrate that the code segment does not work as intended?

A.
```
a ← 20
b ← 15
c ← 15
```

B.
```
a ← 20
b ← 15
c ← 25
```

C.
```
a ← 20
b ← 25
c ← 25
```

D.
```
a ← 20
b ← 25
c ← 30
```

Answer and Rationale

Note: After clicking on a link, right click and select "Previous View" to go back to original text.

18. Huffman coding assigns unique variable-length codes to input values based on the frequency of occurrence of each value. Frequently occurring values are assigned codes that contain fewer bits than values that occur less frequently, which are assigned codes that contain more bits. Which of the following best describes an appropriate use of Huffman coding?

    A. Decryption
    B. Efficient sorting
    C. Lossless compression
    D. Lossy compression

Answer and Rationale

19. Which of the following spreadsheet functions would be most useful for detecting improbably high or low values that have become part of a data set as a result of data entry errors?

    A. A function that averages numeric values in a column or row
    B. A function that counts the values in a column or row
    C. A function that rounds a numeric value
    D. A function that sorts values in a column or row

Answer and Rationale

20. A computer simulation is created to simulate the growth of a certain plant species in different conditions. Which of the following actions could be used to validate the model used in the simulation?

    A. Express the simulation software using both recursive and iterative algorithms. Compare the results of the recursive algorithm to those of the iterative algorithm.
    B. Perform real-world experiments on the plant species' growth in different environments. Compare the experimental results to the results provided by the simulation.
    C. Remove any unnecessary details from the model. Compare the running times of the original simulation and the simplified simulation.
    D. Run the simulation software on multiple devices. Compare the results obtained from each of the devices.

Answer and Rationale

Note: After clicking on a link, right click and select "Previous View" to go back to original text.

21. Consider the following assumptions about electronic storage for the text in all the books of a university's libraries.

- The libraries at the university collectively contain 3 million books.
- A book contains an average of 400 pages.
- A page contains an average of 50 lines.
- A line on a page contains an average of 10 words.
- A word contains an average of 5 letters/characters.
- A letter/character is represented by 1 byte.

Based on the given assumptions, which of the following is the unit in which the electronic storage required for the text in all the books of the university's libraries would best be measured?

A. Megabyte (2^{20} or approximately 10^6 bytes)
B. Gigabyte (2^{30} or approximately 10^9 bytes)
C. Terabyte (2^{40} or approximately 10^{12} bytes)
D. Petabyte (2^{50} or approximately 10^{15} bytes)

Answer and Rationale

22. Which of the following is an example of the use of a device on the Internet of Things (IoT)?

A. A car alerts a driver that it is about to hit an object.
B. A hiker uses a GPS watch to keep track of her position.
C. A refrigerator orders milk from an online delivery service when the milk in the refrigerator is almost gone.
D. A runner uses a watch with optical sensors to monitor his heart rate.

Answer and Rationale

Note: After clicking on a link, right click and select "Previous View" to go back to original text.

23. What is the role of the compiler in the process of developing executable software?

    A. Managing specification files created as part of the development process
    B. Running and testing the executable created by the programmer
    C. Tracking older versions of the software in case an error is found and the software needs to be reverted to an earlier form
    D. Translating a program written in an abstract, high-level language into a program with the same behavior expressed in machine code

Answer and Rationale

24. Which of the following best describes a Web server?

    A. A computer system that delivers Web pages to clients
    B. A computer system that determines the shortest path between two computers over the Internet
    C. A computer system running software that provides a user-friendly interface for creating Web pages
    D. A computer system that translates domain names to IP addresses

Answer and Rationale

25. Which pillar of cybersecurity is compromised when someone logs into a system using a stolen login and password?

    A. Authentication
    B. Confidentiality
    C. Integrity
    D. Nonrepudation

Answer and Rationale

Note: After clicking on a link, right click and select "Previous View" to go back to original text.

Answer Key and Rationales

Question Number	Correct Answer	Rationale
1	A, B, C	**Options A, B, and C are correct.** All three choices are possible uses of the application. **Option A:** Using the application to combine a transcript of an unrecorded speech with existing recordings of an individual would create an audio reconstruction of the unrecorded speech. **Option B:** Using the application to combine false statements with existing recordings of an individual would create an audio file in which the individual seems to be uttering the false statements. **Option C:** The application could be used to create voice commands that could fool voice activation systems. Back to Question
2	D	**Option D is correct.** The Creative Commons Attribution (CC BY) license allows anyone to use, revise, and distribute the code, even commercially, as long as the original author is credited properly. Back to Question
3	C	**Option C is correct.** A DDoS attack is a DoS attack that originates from many different sources. Back to Question
4	A	**Option A is correct.** Of the entries in the list, block-based programming language has the highest level of abstraction, logic gate has the lowest level of abstraction, and the other two entries are in between. Machine language is closer to hardware than is assembly language, so assembly language has a higher level of abstraction than does machine language. Back to Question

Note: After clicking on a link, right click and select "Previous View" to go back to original text.

Question Number	Correct Answer	Rationale
5	C	**Option C is correct.** An analysis of the information in the table shows that: (i) when neither of the two dice is a 6, the score increases by the sum of the values of the two dice; (ii) when both of the dice are a 6, the score becomes 0; and (iii) when one but not both of the two dice is a 6, the score remains the same. Only the code segment in option C is compatible with these observations — the first `return` statement in option C corresponds to case (i) above; the second `return` statement corresponds to case (ii); and the third `return` statement corresponds to case (iii). Back to Question
6	C	**Option C is correct.** Since 231 = 14 x 16 + 7, the decimal number 231_{10} is equivalent to the hexadecimal number $E7_{16}$. Back to Question
7	A	**Option A is correct.** The algorithm solves the traveling salesman problem using a brute-force approach, which enumerates all possible trips and identifies the least expensive one. The number of possible orderings of the clients' cities is factorial in the number of cities, so the running time of the algorithm is factorial. Back to Question
8	B	**Option B is correct.** At the end of the first iteration, the value of `pos` is `1` and the array is unchanged. At the end of the second iteration, the value of `pos` is `0` and the array is `{1, 2, 5, 3, 4}`. At the end of the third iteration, the value of `pos` is `1` and the array is `{1, 2, 5, 3, 4}`. At the end of the fourth iteration, the value of `pos` is `2` and the array is `{1, 2, 5, 3, 4}`. At the end of the fifth iteration, the value of `pos` is `3` and the array is `{1, 2, 5, 3, 4}`. At the end of the sixth iteration, the value of `pos` is `2` and the array is `{1, 2, 3, 5, 4}`. Back to Question

Note: After clicking on a link, right click and select "Previous View" to go back to original text.

Question Number	Correct Answer	Rationale
9	A	**Option A is correct.** When the execution of the code reaches the missing statement, the value of `n` is greater than or equal to `6`. When `n ≥ 6`, the value of `f ( n )` is the product of consecutive integers from `5` to `n`, which is the product of the consecutive integers from `n` down to `5`, which is `n` times the product of the consecutive integers from `n – 1` down to `5`, which is `n` times `f ( n – 1 )`. Therefore, when `n ≥ 6`, we have the recurrence relation `f ( n ) = n * f ( n – 1 )`. Back to Question
10	C	**Option C is correct.** At the beginning of each iteration of the **`while`** loop, the value of `max` is the largest value in the subarray `numList[0..i-1]`. The missing statement is `numList[i] > max`, which is equivalent to saying that `numList[i]` is greater than all the elements in the subarray `numList[0..i-1]`. If the comparison is true, the value of `numList[i]` is assigned to `max`; If not, the value of `max` is unchanged. Back to Question
11	D	**Option D is correct.** The procedure does not work correctly — as is, the procedure does not print anything when `n` is `1` and prints all consecutive integers from `2` to `n` when `n ≥ 2`. To fix the procedure, the integer division operator `/` in the **`if`** condition needs to be replaced with the modulus (remainder) operator `%`. Back to Question

Note: After clicking on a link, right click and select "Previous View" to go back to original text.

Question Number	Correct Answer	Rationale
12	B	**Option B is correct.** The `if-else` statement checks whether the value of `n` is an odd integer; if `n` is odd, it replaces it with one more than its triple; if `n` is even, it halves it. Since the initial value of `n` is `6`, an even number, the procedure prints `3`. Since `3` is an odd number, the procedure prints `10`. Since `10` is an even number, the procedure prints `5`. Since `5` is an odd number, the procedure prints `16`. Since `16` is an even number, the procedure prints `8`. Since `8` is an even number, the procedure prints `4`. Since `4` is an even number, the procedure prints `2`. Since `2` is an even number, the procedure prints `1`. Since the value of `n` is now `1`, the execution exits the `while` loop and the procedure finishes executing. Back to Question
13	D	**Option D is correct.** The Boolean expression `( age > 10 ) and ( height > 36 )` is true exactly when both `age` is greater than `10` and `height` is greater than `36`. The same Boolean expression is false exactly either when `age` is less than or equal to `10` or when `height` is less than or equal to `36`, which is the argument of `not ()` in option D. Another approach to solve this question is to use Boolean logic rules. According to one of De Morgan's laws, if `p` and `q` represent Boolean variables, the Boolean expression `p and q` is equivalent to the Boolean expression `not ( not ( p ) or not ( q ))`. If `p` represents the Boolean expression `age > 10` and if `q` represents the Boolean expression `height > 36`, then `not ( p )` represents `age ≤ 10` and `not ( q )` represents `height ≤ 36`. Back to Question

Note: After clicking on a link, right click and select "Previous View" to go back to original text.

Question Number	Correct Answer	Rationale
14	A	**Option A is correct.** To transform `"Applesauce"` into `"aPPLESAUCE"`, the character at position `0` needs to change to lowercase and the characters from position `1` through the end of the string need to change to uppercase. The string that represents the character at position `0` is `value.substring ( 0, 1 )`. The string that represents the characters at positions `1` and higher is `value.substring ( 1, value.length() ).` Back to Question
15	D	**Option D is correct.** The **`for`** loop iterates over the values of `c` from `1` to `n`, but the value of `temp` increases by `c` only when `c` is a multiple of `3`. Back to Question
16	C	**Option C is correct.** In scenario 1, the cars are washed in first-in first-out order, which characterizes the functionality of a queue. In scenario 2, the unique numeric IDs represent a mapping function, which is best supported by a dictionary/map data structure. In scenario 3, the balls are inserted in and removed from the can in first-in last-out order, which characterizes the functionality of a stack. Back to Question
17	B	**Option B is correct.** Since `20` is greater than `15`, the outer **`if`** condition evaluates to **`true`** and then the condition `a > c` is evaluated. Since `20` is not greater than `25`, the condition `a > c` evaluates to **`false`** and `max` is not updated. The result is that the value of `max` at the end of the code segment is `-1` instead of `25`. Back to Question

Note: After clicking on a link, right click and select "Previous View" to go back to original text.

Question Number	Correct Answer	Rationale
18	C	**Option C is correct.** Huffman coding is appropriately used for lossless compression because the original values can be recovered from the uniquely assigned codes with no loss of data. Back to Question
19	D	**Option D is correct.** A function that sorts values in a column or row will facilitate detecting extreme values in that column or row. Back to Question
20	B	**Option B is correct.** The model is validated by comparing the simulation results with the real-world data. Back to Question
21	C	**Option C is correct.** The size of the storage needed is approximately $3 \times 10^6 \times 4 \times 10^2 \times 5 \times 10^1 \times 10^1 \times 5 \times 1$ bytes, which is equivalent to 3×10^{12} bytes. This number is best expressed in terabytes. Back to Question
22	C	**Option C is correct.** An Internet-connected smart refrigerator that can track its milk inventory would be able to communicate with online delivery services and order milk shipments whenever the milk inventory is low. Back to Question

Note: After clicking on a link, right click and select "Previous View" to go back to original text.

Question Number	Correct Answer	Rationale
23	D	**Option D is correct.** A compiler translates a program written in a high-level programming language into low-level instructions that can be read and executed by the computer. Back to Question
24	A	**Option A is correct.** A Web server is a server that provides content to clients by using Hypertext Transfer Protocol (HTTP). Back to Question
25	A	**Option A is correct.** Authentication is the process of confirming a valid identification. Successful use of false credentials results in incorrect identification of the user. Back to Question

Note: After clicking on a link, right click and select "Previous View" to go back to original text.

Preparation Resources

The resources listed below may help you prepare for the GACE assessment in this field. These preparation resources have been identified by content experts in the field to provide up-to-date information that relates to the field in general. You may wish to use current issues or editions of these materials to obtain information on specific topics for study and review.

Guide to Taking a GACE Computer-delivered Assessment

This guide explains how to navigate through a GACE assessment and how to answer different types of test questions. This free download is available in the Test Preparation Resources section of the GACE website at ***www.gace.ets.org/prepare***.

Reducing Test Anxiety

This guide provides practical help for people who suffer from test anxiety. Designed specifically for GACE test takers, but useful to anyone who has to take tests, this guide reviews the major causes of test anxiety and offers practical advice for how to counter each one. Download this guide for free from the Test Preparation Resources section of the GACE website at ***www.gace.ets.org/prepare***.

Study Tips: Preparing for a GACE Assessment

This document contains useful information on preparing for selected-response and constructed-response tests. The instruction, tips, and suggestions can help you become a better-prepared test taker. See the Test Preparation Resources section of the GACE website at ***www.gace.ets.org/prepare*** for this free download.

Journals

Journal of Research on Technology in Education, International Society for Technology in Education — www.iste.org

The ACM Transactions on Computing Education, Association for Computing Machinery — toce.acm.org

Journal of Digital Learning in Teacher Education, Taylor & Francis Group — www.tandfonline.com/toc/ujdl20/current

Converge, Center for Digital Education, e.Republic Inc. — www.centerdigitaled.com

State-adopted Instructional Materials

Georgia Department of Education: Learning Resources/Textbook and Instructional Materials — www.gadoe.org/Curriculum-Instruction-and-Assessment/Curriculum-and-Instruction/Pages/Learning-Resources.aspx

Note: After clicking on a link, right click and select "Previous View" to go back to original text.

Other Resources

Bitter, G., and Legacy, J. (2008). *Using Technology in the Classroom*. Boston, Mass.: Allyn and Bacon.

Brookshear, J. G. (2009). *Computer Science: An Overview*. Boston, Mass.: Pearson Addison Wesley.

Campione, M., Walrath, K., and Huml, A. (2006). *The Java Tutorial: A Short Course on the Basics*. Upper Saddle River, N.J.: Pearson/Addison-Wesley.

Cormen, T. H., Leiserson, C. E., Rivest, R. L., and Stein, C. (2009). *Introduction to Algorithms*. New York, N.Y.: MIT Press, McGraw-Hill.

Goodrich, M. T., and Tamassia, R. (2010). *Data Structures and Algorithms in Java*. Indianapolis, Ind.: Wiley Text Books.

Horstmann, C. (2008). *Big Java*. Indianapolis, Ind.: Wiley Text Books.

Lever-Duffy, J., McDonald, J., and Mizell, A. P. (2008). *Teaching and Learning with Technology*. Boston, Mass.: Pearson Allyn & Bacon.

Lockard, J., and Abrams, P. (2004). *Computers for Twenty-First Century Educators*. Boston, Mass.: Pearson Allyn & Bacon.

Norton, P. (2006). *Computing Fundamentals*. Westerville, Ohio: Glencoe/McGraw-Hill.

Oualline, S. (1997). *Practical C Programming*. Sebastopol, Calif.: O'Reilly & Associates.

Roblyer, M. D. and Doering A. (2010). *Integrating Educational Technology into Teaching*. Boston, Mass.: Pearson Allyn & Bacon.

Sebesta, R. W. (2010). *Concepts of Programming Languages*. Boston, Mass.: Pearson Addison-Wesley.

Sommerville, I. (2006). *Software Engineering*. Boston, Mass.: Addison-Wesley Publishing Co.

Tannenbaum, A. S. (2006). *Structured Computer Organization*. Upper Saddle River, N.J.: Prentice Hall.

Weiss, M. A. (2007). *Data Structures and Algorithm Analysis in C++*. Boston, Mass.: Pearson Addison-Wesley.

Online Resources

AP Computer Science — ***apcentral.collegeboard.com/apc/public/courses/teachers_corner/4483.html***

Association for Computing Machinery — ***www.acm.org/education***

Association for Computing Machinery (ACM) Special Interest Group on Computer Science Education — ***www.sigcse.org***

Blue Pelican Java (free Java textbook and videos) — ***www.bluepelicanjava.com***

Code.org — ***code.org/educate/curriculum***

Code Highschool (codeHS) — ***codehs.com***

CS Unplugged — ***csunplugged.org***

Note: After clicking on a link, right click and select "Previous View" to go back to original text.

eSchoolNews — ***www.eschoolnews.com/2013/12/10/computer-science-resources-168***

Georgia Association of Educators (GAE) — ***pv.gae2.org***

Georgia Computer Science Teachers Association (CSTA) — ***sites.google.com/site/georgiacsta***

Georgia Department of Education — ***www.gapsc.com***

Georgia Professional Standards Commission — ***www.gadoe.org***

Hour of Code — ***hourofcode.com***

International Society for Technology in Education (ISTE) — ***www.iste.org***

ISTE Standards for Computer Science Educators — ***www.iste.org/standards/iste-standards/standards-for-computer-science-educators***

Javabat (free online Java interactive learning tool) — ***www.javabat.com***

National Center for Women & Information Technology (NCWIT) — ***www.ncwit.org***

Professional Association of Georgia Educators (PAGE) — ***www.pageinc.org***

Project Lead the Way (PLTW) — ***www.pltw.org/our-programs/computer-science***

Stanford Computer Science, Nifty Assignments from the Annual SIGCSE Meeting — ***http://nifty.stanford.edu***

STEM Georgia — ***www.stemgeorgia.org***

U.S. Department of Education — ***www.ed.gov***

Note: After clicking on a link, right click and select "Previous View" to go back to original text.