GACE® Computer Science Assessment
Test at a Glance

Updated May 2017

See the GACE® Computer Science Assessment Study Companion for practice questions and preparation resources.

Assessment Name	Computer Science
Grade Level	P–12
Test Code	554
Testing Time	3 hours
Test Duration	3.5 hours
Number of Selected-response Questions	100
Number of Constructed-response Questions	0
Test Format	Computer delivered

About this Assessment

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.

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.

Test Subareas

Subarea	Approx. Percentage of Test
I. Technology Applications Core	33.3%
II. Program Design and Development	33.3%
III. Programming Language Topics	33.3%

Test Objectives

Subarea I: Technology Applications Core

Objective 1: Knows technology terminology and concepts; the appropriate use of hardware, software, and digital files; and how to acquire, analyze, and evaluate digital information

The beginning Computer Science teacher:

A. Knows technology terminology and concepts

B. Demonstrates knowledge of various types of networks (e.g., LAN, WAN) and models for defining network standards and protocols (e.g., OSI, TCP/IP)

C. Knows the appropriate use of hardware components (e.g., input, processing, output, primary/secondary storage devices), operating systems, software applications, and networking components

D. Knows how to select, connect, and use a variety of input, output, and storage devices and peripherals (e.g., scanner, voice/sound recorders, touch screen, digital camera, printer)

E. Knows how to evaluate software (e.g., graphics, animation, multimedia, video, Web authoring) for quality, appropriateness, effectiveness, and efficiency and how to make decisions regarding its proper acquisition and use

F. Knows how to perform basic application functions (e.g., opening an application program; creating, modifying, saving, and printing documents) and how to access, manage, and manipulate information from secondary storage devices

G. Knows strategies for acquiring information from electronic resources (e.g., encyclopedias, databases, libraries of images, reference software, Internet)

H. Knows search strategies (e.g., keyword, Boolean, natural language) for locating and retrieving information in electronic formats (e.g., text, audio, video, graphics)

I. Knows how to assess the accuracy and validity of acquired information

J. Knows how to resolve information conflicts through research and comparison of data from multiple sources

K. Demonstrates knowledge of the ethical acquisition (e.g., citing sources using established methods) and acceptable versus unacceptable use of information (e.g., privacy, hacking, piracy, vandalism, viruses, current laws and regulations)

L. Demonstrates knowledge of intellectual property rights and related issues (e.g., copyright laws, fair use, patents, trademarks) when using, manipulating, and editing electronic data

M. Knows how to use online help and other support documentation

N. Knows how to use technical-writing strategies to develop documentation for a variety of communication products

O. Demonstrates knowledge of the impact of technology on society and the importance of technology to future careers, lifelong learning, and daily living for individuals of all ages

P. Investigates measures (e.g., passwords, virus detection/prevention) to protect computer systems and databases from unauthorized use and tampering

Objective 2: Knows how to use technology tools to solve problems, evaluate results, and communicate information in a variety of formats for diverse audiences

The beginning Computer Science teacher:

A. Knows how to plan, create, and edit documents using word processing features (e.g., readable fonts, alignment, page setup, tabs, ruler settings) to solve problems and communicate results

B. Knows how to plan, create, and edit spreadsheets using spreadsheet features (e.g., data types, formulas, functions, charts) to solve problems and communicate results

C. Knows how to plan, create, and edit databases using database features (e.g., defining fields, entering data, creating horizontal and vertical layouts) to solve problems and communicate results

D. Knows how to integrate one or more objects (e.g., tables, charts, graphs, graphics) into a product

E. Knows how to use productivity tools to create products (e.g., slide shows, posters, multimedia presentations, spreadsheets) for defined audiences

F. Knows how to publish information in a variety of ways (e.g., printed copy, monitor displays, Internet documents, video)

G. Knows how to use telecommunications tools (e.g., Internet browsers, video conferencing, distance learning) for a variety of purposes

H. Knows how to use interactive virtual environments (e.g., virtual field trips, instructional simulations)

I. Knows how to use collaborative software

J. Knows how to share information through online communication

K. Demonstrates knowledge of issues concerning proper etiquette when communicating using electronic tools

L. Demonstrates knowledge of how to design and implement procedures to track trends, set timelines, and review and evaluate products using technology tools (e.g., database managers, daily/monthly planners, project management tools)

M. Knows how to evaluate projects for design, purpose, audience, and content delivery using various criteria (e.g., technology specifications, established criteria, rubrics)

N. Knows how to select representative products to be collected and stored in an electronic evaluation tool and how to evaluate products for relevance to the assignment or task

O. Knows how to plan and design communication products that are accessible to learners with diverse needs and abilities

Objective 3: Knows how to plan, organize, deliver, and evaluate instruction that effectively utilizes current technology for teaching technology applications for all students

The beginning Computer Science teacher:

A. Knows how to plan computer science lessons using a range of instructional strategies for individuals and groups

B. Demonstrates knowledge of issues related to the equitable use of technology (e.g., gender, ethnicity, language, disabilities, access to technology)

C. Knows how to plan and implement instruction that allows students to use computer science in problem-solving and decision-making situations

D. Knows how to develop and facilitate collaborative tasks and teamwork among group members

E. Knows how to use technology tools to perform administrative tasks (e.g., attendance, grades, communication)

F. Knows how to use a variety of instructional strategies to ensure students’ reading comprehension

G. Knows strategies to help students learn how to locate, retrieve, analyze, evaluate, communicate, and retain content-related information

H. Knows how to evaluate student projects and portfolios using formal and informal assessment methods

I. Knows the relationship between instruction and assessment and uses assessment results for gauging student progress and adjusting instruction

J. Identifies resources to keep current with the use of technology in education and issues related to legal and ethical use of technology resources

K. Knows how to use technology to participate in self-directed activities in society and how to participate within electronic communities in a variety of roles (e.g., collaborator, learner, contributor, teacher/mentor)

Subarea II: Program Design and Development

Objective 1: Knows problem-solving strategies and different procedures for program design

The beginning Computer Science teacher:

A. Exhibits knowledge of the analysis and design phases of the software system life cycle

B. Knows the characteristics of programming design strategies

C. Knows how to apply problem-solving strategies (e.g., design specification, top-down design, step-wise refinement, object-oriented design)

D. Demonstrates the ability to compare and contrast design strategies (e.g., top-down, bottom-up, object-oriented)

E. Demonstrates the use of visual organizers (e.g., flowcharts, schematic drawings) to design solutions to problems

F. Knows how to create robust programs with emphasis on design to facilitate maintenance, program expansion, reliability, validity, and efficiency

Objective 2: Knows procedures for software development and implementation

The beginning Computer Science teacher:

A. Knows the characteristics of models (e.g., waterfall, incremental, spiral) used in the development of software systems

B. Knows how to survey the issues accompanying the development of large software systems (e.g., design/implementation teams, software validation/testing, risk assessment)

C. Demonstrates the use of programming style conventions (e.g., spacing, indentation, descriptive identifiers, comments, documentation) to enhance the readability and functionality of code

D. Knows how to create robust programs with emphasis on style, clarity of expression, and documentation to facilitate maintenance, program expansion, reliability, validity, and efficiency

E. Knows how to create and use libraries of generic modular code to be used for efficient programming

F. Demonstrates the ability to read and modify large programs, including design description and process development

G. Demonstrates effective use of predefined input and output, including logic to protect from invalid input

H. Demonstrates the ability to debug and solve problems using reference materials and effective strategies

I. Knows how to determine and employ methods to evaluate the design and functionality of information acquisition processes and algorithms, using effective coding, design, and test data

Objective 3: Knows computer science terminology and concepts and the characteristics of different programming languages and paradigms

The beginning Computer Science teacher:

A. Knows necessary vocabulary related to computer science (e.g., cache, bits, encryption)

B. Knows specific programming terminology (e.g., data type, data structure, encapsulation) and programming concepts (e.g., procedural, object-oriented)

C. Demonstrates knowledge of advanced computer science concepts (e.g., computer architecture, operating systems, artificial intelligence)

D. Demonstrates the ability to use notation for language definition (e.g., syntax diagrams, Backus-Naur forms)

E. Knows the differences in the levels of languages (e.g., machine, assembly, high-level compiled, interpreted)

F. Knows the characteristics of and differences in current programming languages and paradigms

G. Demonstrates knowledge of the uses of current programming languages and paradigms in other fields of study

Subarea III: Programming Language Topics

Objective 1: Correctly and efficiently uses data types, data structures, and functions in the development of code

The beginning Computer Science teacher:

A. Knows the characteristics and uses of constants, variables, and simple data types in current programming languages (e.g., int, short, char, double, boolean)

B. Demonstrates effective use of standard and user-defined methods or functions in the development of code

C. Knows how to identify and use parameters, both actual and formal, and how to pass parameters by value and by reference

D. Knows how to identify object-oriented data types and delineate the advantages and disadvantages of object data

E. Demonstrates the ability to identify and use one-dimensional arrays, records, and sequential and nonsequential files

F. Knows how to identify and use multidimensional arrays and arrays of records

G. Demonstrates the ability to develop coding with the use of data structures, and to manipulate data structures using string processing routines (e.g., concatenation of strings, substring search)

H. Knows the characteristics of and develops code using abstract data types (e.g., stacks, queues, linked lists, trees, graphs)

Objective 2: Correctly and efficiently uses statements and control structures in the development of code

The beginning Computer Science teacher:

A. Applies standard operators (e.g., arithmetic, relational, logical, assignment, increment/decrement, input/output) and correct operator precedence

B. Identifies the characteristics of control structures

C. Uses conditional control structures (e.g., if, if . . . else statements)

D. Constructs iterative control structures (e.g., for and while statements, do loops)

E. Uses pretest (e.g., for, while) and posttest (e.g., do . . . while) loops

F. Uses sequential, conditional, selection, and repetition execution control structures such as menu-driven programs that branch and allow user input

G. Demonstrates coding proficiency in contemporary programming languages, including an object-oriented language

Objective 3: Knows how to construct, compare and analyze various algorithms

The beginning Computer Science teacher:

A. Constructs searching algorithms (e.g., linear and binary searches)

B. Constructs sorting algorithms (e.g., selection, bubble, insertion, merge, shell, and quick sorts)

C. Compares and contrasts searching and sorting algorithms for space and time requirements

D. Constructs and appropriately uses iterative and recursive algorithms

E. Compares and contrasts iterative and recursive algorithms

F. Develops sequential, iterative, and recursive algorithms and codes programs in prevailing computer languages to solve practical problems

G. Analyzes various algorithms using “big-O” notation and best-, average-, and worst-case space and time techniques

H. Identifies and describes the correctness and complexity of specific types of algorithms (e.g., divide and conquer, greedy, backtracking)

Code Segments

Example 1

The following are some examples of pseudocode stimulus material.

Class declaration and object instantiation

```
class StudentInfo
   int studentID
   string name
end class StudentInfo
StudentInfo x ← new StudentInfo()
x.studentID ← 1234
   // the value 1234 is assigned to x.studentID
x.name ← "John"
print ( x.studentID )
print ( x.name )
```

Example 2

The following procedure uses different parameter-passing mechanisms for the two parameters.

```
void f ( pass-by-reference int x, pass-by-value int y )
   x ← y + 1
   y ← x + 2
end f
```

Example 3

InsertionSort

```
// precondition 1: A is an array of integers.
// precondition 2: The length of array A is n.
// precondition 3: The index of array A starts at 0.
int[] insertionSort ( pass-by-reference int[] A, int n )
   for ( int j ← 1; j ≤ n - 1; j ← j + 1 )
      int temp ← A[j]
      int k ← j - 1
      while ( ( k ≥ 0 ) and ( A[k] > temp ) )
         A[k + 1] ← A[k]
         k ← k - 1
      end while
      A[k + 1] ← temp
   end for
   return A // returns the sorted array
end insertionSort
```