STATE BOARD FOR EDUCATOR CERTIFICATION/TEXAS EDUCATION AGENCY

TEXAS EDUCATOR CERTIFICATION

TExES/ExCET TOPT TExMaT TASC/TASC-ASL

**TExES** | Texas Examinations of Educator Standards

# Preparation Manual

141 Computer Science 8–12

# Chapter 1

## Introduction to the Computer Science 8–12 Test and Suggestions for Using This Test Preparation Manual

1

## OVERVIEW

The State Board for Educator Certification (SBEC) has approved Texas educator standards that delineate what the beginning educator should know and be able to do. These standards, which are based on the state-required curriculum for students — the Texas Essential Knowledge and Skills (TEKS) — form the basis for the Texas Examinations of Educator Standards® (TExES®) program. This initiative, administered by Texas Education Agency (TEA), will affect all areas of Texas education — from the more than 170 approved Texas Educator Preparation Programs (EPPs) to the more than 7,000 Texas school campuses. This standards-based system reflects SBEC's commitment to help align Texas education from kindergarten through college. SBEC and TEA's roles in this K–16 initiative will ensure that newly certified Texas educators have the essential knowledge and skills to teach the TEKS to the state's public school students.

This manual is designed to help examinees prepare for the TExES test in this field. Its purpose is to familiarize examinees with the competencies to be tested, test question formats and pertinent study resources. EPP staff may also find this information useful as they help examinees prepare for careers as Texas educators.

**KEY FEATURES OF THE MANUAL**

- *List of competencies that will be tested*
- *Strategies for answering multiple-choice questions*
- *Sample test questions and answer key*

If you have any questions after reading this preparation manual or you would like additional information about the TExES tests or the educator standards, please visit the TEA website at **www.tea.state.tx.us**.

## USING THE TEST FRAMEWORK

The Texas Examinations of Educator Standards (TExES) tests measure the content knowledge required of an entry-level educator in a particular field in Texas public schools. This manual is designed to guide your preparation by helping you become familiar with the material to be covered on the test you are planning to take, identify areas where you feel you may be weak and increase your knowledge in those areas by helping you design a study plan.

**When preparing for this test, you should focus on the competencies and descriptive statements, which delineate the content that is eligible for testing. A portion of the content is represented in the sample questions that are included in this manual. These test questions represent only a sampling of questions. Thus, your test preparation should focus on the competencies and descriptive statements and not simply on the sample questions.**

## ORGANIZATION OF THE TExES TEST FRAMEWORK

The test framework is based on the educator standards for this field.

The content covered by this test is organized into broad areas of content called domains. Each domain covers one or more of the educator standards for this field. Within each domain, the content is further defined by a set of competencies. Each competency is composed of two major parts:

1. the **competency statement**, which broadly defines what an entry-level educator in this field in Texas public schools should know and be able to do, and
2. the **descriptive statements**, which describe in greater detail the knowledge and skills eligible for testing.

The educator standards being assessed within each domain are listed for reference at the beginning of the test framework, which begins on page 12. These are followed by a complete set of the framework's competencies and descriptive statements.

An example of a competency and its accompanying descriptive statements is provided below.

**SAMPLE COMPETENCY**

**Computer Science 8–12**

**COMPETENCY 001**

THE COMPUTER SCIENCE TEACHER KNOWS TECHNOLOGY TERMINOLOGY AND CONCEPTS; THE APPROPRIATE USE OF HARDWARE, SOFTWARE AND DIGITAL FILES; AND HOW TO ACQUIRE, ANALYZE AND EVALUATE DIGITAL INFORMATION.

1

**SAMPLE DESCRIPTIVE STATEMENTS**

The beginning 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.

## STUDYING FOR THE TExES TEST

The following steps may be helpful in preparing for the TExES test.

1. Identify the information the test will cover by reading through the test competencies (see Chapter 3). Within each domain of this TExES test, each competency will receive approximately equal coverage.
2. Read each competency with its descriptive statements in order to get a more specific idea of the knowledge you will be required to demonstrate on the test. You may wish to use this review of the competencies to set priorities for your study time.
3. Review the "Preparation Resources" section of this manual (Appendix B) for possible resources to consult. Also, compile key materials from your preparation course work that are aligned with the competencies.
4. Study this manual for approaches to taking the TExES test.
5. When using resources, concentrate on the key skills and important abilities that are discussed in the competencies and descriptive statements.
6. Use the study plan sheet (Appendix A) to help you plan your study.

**NOTE: This preparation manual is the only TExES test study material endorsed by Texas Education Agency (TEA) for this field. Other preparation materials may not accurately reflect the content of the test or the policies and procedures of the TExES program.**

# Chapter 2

## Background Information on the TExES Testing Program

## THE TExES TESTS FOR TEXAS TEACHERS

As required by the Texas Education Code §21.048, successful performance on educator certification examinations is required for the issuance of a Texas educator certificate. Each TExES test is a criterion-referenced examination designed to measure the knowledge and skills delineated in the corresponding TExES test framework. Each test framework is based on standards that were developed by Texas educators and other education stakeholders.

Each TExES test is designed to measure the requisite knowledge and skills that an entry-level educator in this field in Texas public schools must possess. The tests include both individual (stand-alone) test questions and questions that are arranged in clustered sets based on real-world situations faced by educators.

### DEVELOPMENT OF THE NEW TExES TESTS

Committees of Texas educators and members of the community guide the development of the new TExES tests by participating in each stage of the test development process. These working committees are composed of Texas educators from public and charter schools, university and EPP faculty, education service center staff, representatives from professional educator organizations, content experts and members of the business community. The committees are balanced in terms of position, affiliation, years of experience, ethnicity, gender and geographical location. The committee membership is rotated during the development process so that numerous Texas stakeholders may be actively involved. The steps in the process to develop the TExES tests are described below.

1. **Develop Standards.** Committees are established to recommend what the beginning educator should know and be able to do. Using the Texas Essential Knowledge and Skills (TEKS) as the focal point, draft standards are prepared to define the knowledge and skills required of the beginning educator.
2. **Review Standards.** Committees review and revise the draft standards. The revised draft standards are then placed on the TEA website for public review and comment. These comments are used to prepare a final draft of the standards that will be presented to the SBEC Board for discussion, the State Board of Education (SBOE) for review and comment and the SBEC Board for approval. Standards not based specifically on the TEKS, such as those for librarians and counselors, are proposed as rule by the SBEC Board; sent to the SBOE for its 90-day review; and, if not rejected by the SBOE, adopted by the SBEC Board.
3. **Develop Test Frameworks.** Committees review and revise draft test frameworks that are based on the standards. These frameworks outline the specific competencies to be measured on the new TExES tests. Draft frameworks are not finalized until after the standards are approved and the job analysis/content validation survey (see #4) is complete.

4. **Conduct Job Analysis/Content Validation Surveys.** A representative sample of Texas educators who practice in or prepare individuals for each of the fields for which an educator certificate has been proposed are surveyed to determine the relative job importance of each competency outlined in the test framework for that content area. Frameworks are revised as needed following an analysis of the survey responses.
5. **Develop and Review New Test Questions.** The test contractor develops draft questions that are designed to measure the competencies described in the test framework. Committees review the newly developed test questions that have been written to reflect the competencies in the new test framework. Committee members scrutinize the draft questions for appropriateness of content and difficulty; clarity; match to the competencies; and potential ethnic, gender and regional bias.
6. **Conduct Pilot Test of New Test Questions.** All of the newly developed test questions that have been deemed acceptable by the question review committees are then administered to an appropriate sample of candidates for certification.
7. **Review Pilot Test Data.** Pilot test results are reviewed to ensure that the test questions are valid, reliable and free from bias.
8. **Administer TExES Tests.** New TExES tests are constructed to reflect the competencies, and the tests are administered to candidates for certification.
9. **Set Passing Standard.** A Standard Setting Committee convenes to review performance data from the initial administration of each new TExES test and to recommend a final passing standard for that test. The SBEC Board considers this recommendation as it establishes a passing score on the test.

2

## TAKING THE TExES TEST AND RECEIVING SCORES

Please refer to the current TExES *Registration Bulletin* or the ETS TExES website at **www.texes.ets.org** for information on test dates, test centers, fees, registration procedures and program policies.

Your score report will be available to you in your testing account on the ETS TExES online registration system by 5 p.m. Central time on the score reporting date indicated in the *Registration Bulletin*. The report will indicate whether you have passed the test and will include:

- A total test scaled score. Scaled scores are reported to allow for the comparison of scores on the same content-area test taken on different test administration dates. The total scaled score is not the percentage of questions answered correctly and is not determined by averaging the number of questions answered correctly in each domain.
  - For all TExES tests, the score scale is 100–300 with a scaled score of 240 as the minimum passing score. This score represents the minimum level of competency required to be an entry-level educator in this field in Texas public schools.
- Your performance in the major content domains of the test and in the specific content competencies of the test.
  - This information may be useful in identifying strengths and weaknesses in your content preparation and can be used for further study or for preparing to retake the test. However, it is important to use caution when interpreting scores reported by domain and competency as these scores are typically based on a smaller number of items than the total score and therefore may not be as reliable as the total score.
- A link to information to help you understand the score scale and interpret your results.

A score report will not be available to you if you are absent or choose to cancel your score.

For more information about scores or to access scores online, go to **www.texes.ets.org**.

### EDUCATOR STANDARDS

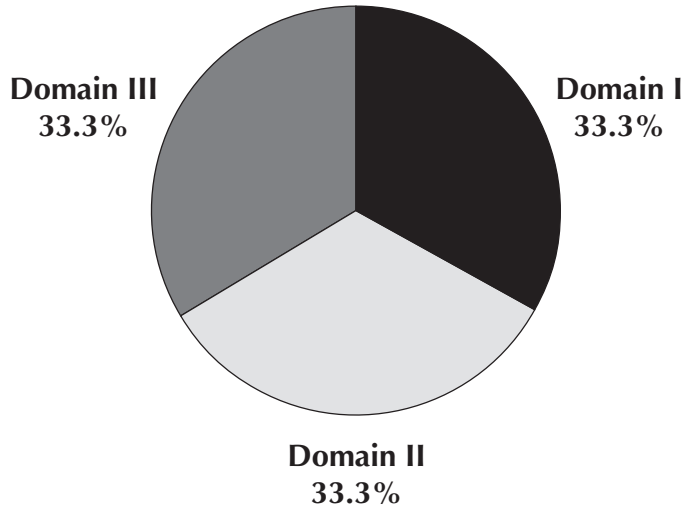
Complete, approved educator standards are posted on the TEA website at **www.tea.state.tx.us**.

# Chapter 3

## Study Topics

## TEST FRAMEWORK FOR FIELD 141: COMPUTER SCIENCE 8–12

**THE DOMAINS**




- **Domain I: Technology Applications Core**
  Standards Assessed: I–VI
- **Domain II: Program Design and Development**
  Standard Assessed: VI
- **Domain III: Programming Language Topics**
  Standard Assessed: VI

**TOTAL TEST BREAKDOWN**

- Exam is offered as a paper-based test
- 100 Multiple-Choice Questions (80 Scored Questions*)

* The number of scored questions will not vary; however, the number of questions that are not scored may vary in the actual test. Your final scaled score will be based only on scored questions.

## THE STANDARDS

**DOMAIN I — TECHNOLOGY APPLICATIONS CORE (approximately 33.3% of the test)**

**COMPUTER SCIENCE 8–12 STANDARD I:**
All teachers use technology-related terms, concepts, data input strategies and ethical practices to make informed decisions about current technologies and their applications.

**COMPUTER SCIENCE 8–12 STANDARD II:**
All teachers identify task requirements, apply search strategies and use current technology to efficiently acquire, analyze and evaluate a variety of electronic information.

**COMPUTER SCIENCE 8–12 STANDARD III:**
All teachers use task-appropriate tools to synthesize knowledge, create and modify solutions and evaluate results in a way that supports the work of individuals and groups in problem-solving situations.

**COMPUTER SCIENCE 8–12 STANDARD IV:**
All teachers communicate information in different formats and for diverse audiences.

**COMPUTER SCIENCE 8–12 STANDARD V:**
All teachers know how to plan, organize, deliver and evaluate instruction for all students that incorporates the effective use of current technology for teaching and integrating the Technology Applications Texas Essential Knowledge and Skills (TEKS) into the curriculum.

**COMPUTER SCIENCE 8–12 STANDARD VI:**
The computer science teacher has the knowledge and skills needed to teach the Foundations, Information Acquisition, Work in Solving Problems and Communication strands of the Technology Applications Texas Essential Knowledge and Skills (TEKS) in computer science, in addition to the content described in Technology Applications Standards I–V.

**DOMAIN II — PROGRAM DESIGN AND DEVELOPMENT (approximately 33.3% of the test)**

**COMPUTER SCIENCE 8–12 STANDARD VI:**
The computer science teacher has the knowledge and skills needed to teach the Foundations, Information Acquisition, Work in Solving Problems and Communication strands of the Technology Applications Texas Essential Knowledge and Skills (TEKS) in computer science, in addition to the content described in Technology Applications Standards I–V.

**DOMAIN III — PROGRAMMING LANGUAGE TOPICS (approximately 33.3% of the test)**

**COMPUTER SCIENCE 8–12 STANDARD VI:**
The computer science teacher has the knowledge and skills needed to teach the Foundations, Information Acquisition, Work in Solving Problems and Communication strands of the Technology Applications Texas Essential Knowledge and Skills (TEKS) in computer science, in addition to the content described in Technology Applications Standards I–V.

## COMPETENCIES

### DOMAIN I — TECHNOLOGY APPLICATIONS CORE

#### COMPETENCY 001

THE COMPUTER SCIENCE TEACHER 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 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.



**COMPETENCY 002**

THE COMPUTER SCIENCE TEACHER KNOWS HOW TO USE TECHNOLOGY TOOLS TO SOLVE PROBLEMS, EVALUATE RESULTS AND COMMUNICATE INFORMATION IN A VARIETY OF FORMATS FOR DIVERSE AUDIENCES.

The beginning 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.

### COMPETENCY 003

THE COMPUTER SCIENCE TEACHER KNOWS HOW TO PLAN, ORGANIZE, DELIVER AND EVALUATE INSTRUCTION THAT EFFECTIVELY UTILIZES CURRENT TECHNOLOGY FOR TEACHING THE TECHNOLOGY APPLICATIONS TEXAS ESSENTIAL KNOWLEDGE AND SKILLS (TEKS) FOR ALL STUDENTS.

The beginning 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).

## DOMAIN II — PROGRAM DESIGN AND DEVELOPMENT



### COMPETENCY 004

THE COMPUTER SCIENCE TEACHER KNOWS PROBLEM-SOLVING STRATEGIES AND DIFFERENT PROCEDURES FOR PROGRAM DESIGN.

The beginning 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.

### COMPETENCY 005

THE COMPUTER SCIENCE TEACHER KNOWS PROCEDURES FOR SOFTWARE DEVELOPMENT AND IMPLEMENTATION.

The beginning 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.

## COMPETENCY 006

THE COMPUTER SCIENCE TEACHER KNOWS COMPUTER SCIENCE TERMINOLOGY AND CONCEPTS AND THE CHARACTERISTICS OF DIFFERENT PROGRAMMING LANGUAGES AND PARADIGMS.

The beginning 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.

## DOMAIN III — PROGRAMMING LANGUAGE TOPICS

### COMPETENCY 007

THE COMPUTER SCIENCE TEACHER CORRECTLY AND EFFICIENTLY USES DATA TYPES, DATA STRUCTURES AND FUNCTIONS IN THE DEVELOPMENT OF CODE.

The beginning 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).

### COMPETENCY 008

THE COMPUTER SCIENCE TEACHER CORRECTLY AND EFFICIENTLY USES STATEMENTS AND CONTROL STRUCTURES IN THE DEVELOPMENT OF CODE.

The beginning 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.

**COMPETENCY 009**

THE COMPUTER SCIENCE TEACHER KNOWS HOW TO CONSTRUCT, COMPARE AND ANALYZE VARIOUS ALGORITHMS.

The beginning 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 code 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).

# Chapter 4

## Succeeding on Multiple-Choice Questions

## APPROACHES TO ANSWERING MULTIPLE-CHOICE QUESTIONS

The purpose of this section is to describe multiple-choice question formats that you will see on the TExES Computer Science 8–12 test and to suggest possible ways to approach thinking about and answering the multiple-choice questions. However, these approaches are not intended to replace familiar test-taking strategies with which you are already comfortable and that work for you.

The Computer Science 8–12 test is designed to include a total of 100 multiple-choice questions, out of which 80 are scored. The number of scored questions will not vary; however, the number of questions that are not scored may vary in the actual test. Your final scaled score will be based only on scored questions. The questions that are not scored are being pilot tested in order to collect information about how these questions will perform under actual testing conditions. These questions are not identified on the test.



All multiple-choice questions on this test are designed to assess your knowledge of the content described in the test framework. The multiple-choice questions assess your ability to recall factual information and to think critically about the information, analyze it, consider it carefully, compare it with other knowledge you have, or make a judgment about it.

When you are ready to respond to a multiple-choice question, you must choose one of four answer options labeled A, B, C and D. Leave no questions unanswered. Nothing is subtracted from your score if you answer a question incorrectly. Questions for which you mark no answer or more than one answer will be counted as incorrect. Your score will be determined by the number of questions for which you select the best answer.

## QUESTION FORMATS

You may see the following types of multiple-choice questions on the test.

— Single Questions

— Questions with Stimulus Materials (Code Segments)

On the following pages, you will find descriptions of these commonly used question formats, along with suggested approaches for responding to each type of question. In the actual testing situation, you may mark the test questions and/or write in the margins of your test booklet. **Your final response must be indicated on the answer sheet provided.**

### SINGLE QUESTIONS

In the single-question format, a problem is presented as a direct question or an incomplete statement, and four answer options appear below the question. The following two questions are examples of this type. The first question tests knowledge of Computer Science 8–12, Competency 003: *The Computer Science teacher knows how to plan, organize, deliver and evaluate instruction that effectively utilizes current technology for teaching the Technology Applications Texas Essential Knowledge and Skills (TEKS) for all students.*

*EXAMPLE 1*

The most effective method of helping students improve their comprehension of the concepts and constructs of a specific programming language is to

A. have the students memorize the syntax of the language.

B. require the students to read several books about the language.

C. give the students quizzes on a variety of topics related to the language.

D. ask the students to write and test several programs using the language.



*SUGGESTED APPROACH*

Read the question carefully and critically. Think about what it is asking and the situation it is describing. Eliminate any obviously wrong answers, select the correct answer choice and mark your answer.

In this situation, think about the most effective method of helping students improve their comprehension of the concepts and constructs of a specific programming language. Now look at the answer options and consider which describes the most effective method.

Option A indicates that having students memorize the syntax of a programming language is the most effective way of helping them improve their comprehension of its concepts and constructs. Knowledge of the syntax of a programming language is important. However, memorizing syntax is not effective in improving comprehension of the language. Therefore, option A is not an appropriate response.

Option B indicates that having students read several books on a programming language is the most effective way of helping them improve their comprehension of its concepts and constructs. Reading books on the language is worthwhile. However, doing so is not an extremely effective way of improving comprehension of the concepts and constructs of the language. Therefore, option B is not an appropriate response.

Option C indicates that giving students quizzes on several topics related to a programming language is the most effective way of helping them improve their comprehension of its concepts and constructs. Testing students on their knowledge of the language is worthwhile. However, doing so in the form of quizzes is not effective in helping them improve their comprehension of the concepts and constructs of the language. Therefore, option C is not an appropriate response.

Option D indicates that asking students to write and test several programs using the language is the most effective way of helping them improve their comprehension of its concepts and constructs. By writing programs and then testing them, students are able to implement various concepts of a programming language and, in doing so, they can improve their understanding of these concepts. Therefore, option D is an appropriate response.

Of the alternatives offered, having students write and test several programs using a particular programming language is the most effective way of helping them improve their comprehension of the language's concepts and constructs. Therefore, **the correct response is option D**.

***EXAMPLE 2***

The second question tests knowledge of Computer Science 8–12, Competency 004: *The computer science teacher knows problem-solving strategies and different procedures for program design*.

> Which of the following distinguishes an object-oriented programming design strategy from other design strategies?
>
> A. Modules are tested individually and then together
> B. A requirements specification is created as a first step
> C. Functions and data are treated as integrated components
> D. An incremental approach is used in the system development

***SUGGESTED APPROACH***

Read the question carefully and critically. Think about what it is asking and the situation it is describing. Eliminate any obviously wrong answers, select the correct answer choice and mark your answer.

In this situation, think about the design characteristics of an object-oriented programming design strategy that are distinct from those of a procedural programming design strategy. Now look at the answer options and consider which best describes one such distinction.

Option A suggests that a feature that distinguishes an object-oriented programming design strategy from other design strategies is that modules are tested individually and then together. This is a feature of object-oriented programming design strategies. However, it is also a feature of other programming design strategies, including procedural programming designs. Therefore, option A is not a feature that distinguishes an object-oriented programming design strategy from other design strategies.

Option B suggests that a feature that distinguishes an object-oriented programming design strategy from other design strategies is that a requirements specification is created as a first step in the design. This is a feature of object-oriented programming design strategies. However, it is also a feature of other programming design strategies, including procedural programming designs. Therefore, option B is not a feature that distinguishes an object-oriented programming design strategy from other design strategies.

Option C suggests that a feature that distinguishes an object-oriented programming design strategy from other design strategies is that functions and data are treated as integrated components. This is a feature of object-oriented programming design strategies that is not present in other programming design strategies, including procedural programming designs. Therefore, option C is a feature that distinguishes an object-oriented programming design strategy from other design strategies.

Option D suggests that a feature that distinguishes an object-oriented programming design strategy from other design strategies is that an incremental approach is used in the system development. This is a feature of object-oriented programming design strategies. However, it is also a feature of other programming design strategies, including procedural programming designs. Therefore, option D is not a feature that distinguishes an object-oriented programming design strategy from other design strategies.

Of the alternatives offered, only treating functions and data as integrated components could be considered a distinguishing feature of an object-oriented programming design strategy. Therefore, **the correct response is option C**.



### QUESTIONS WITH STIMULUS MATERIAL

Some questions on this test are preceded by stimulus material that relates to the question. Some types of stimulus material included on the test are reading passages, code segments, graphics, tables or a combination of these. In such cases, you will generally be given information followed by an event to analyze, a problem to solve or a decision to make.

You can use several different approaches to respond to these types of questions. Some commonly used strategies are listed below.

**Strategy 1** Skim the stimulus material to understand its purpose, its arrangement and/or its content. Then read the question and refer again to the stimulus material to obtain the specific information you need to answer the question.

**Strategy 2** Read the question *before* considering the stimulus material. The theory behind this strategy is that the content of the question will help you identify the purpose of the stimulus material and locate the information you need to respond to the question.

**Strategy 3** Use a combination of both strategies; apply the "read the stimulus first" strategy with shorter, more familiar stimuli and the "read the question first" strategy with longer, more complex or less familiar stimuli. You can experiment with the sample questions in this manual and then use the strategy with which you are most comfortable when you take the actual test.

Whether you read the stimulus before or after you read the question, you should read it carefully and critically. You may want to underline its important points to help you answer the question.

As you consider questions set in educational contexts, try to enter into the identified teacher's frame of mind and use that teacher's point of view to answer the questions that accompany the stimulus. Be sure to consider the questions in terms of only the information provided in the stimulus — not in terms of your own class experiences or individual students you may have known.

***EXAMPLE 1***

The following question contains a code segment as stimulus material. The question measures knowledge of Computer Science 8–12, Competency 008: *The computer science teacher correctly and efficiently uses statements and control structures in the development of code*.



Consider the following pseudocode segment.

```
int k ← 18
for ( int j ← 1; j ≤ 5; j ← j + 1 )
   if ( k > 10 )
      /* missing if body */
   end if
end for
```

If the value of `k` at the end of the code segment is `10`, which of the following expressions should replace `/* missing if body */`?

A. `k ← k - 1`

B. `k ← k - 3`

C. `k ← k - 4`

D. `k ← k - 6`

***SUGGESTED APPROACH***

Examine carefully the pseudocode segment presented in the stimulus; then read the question. Now substitute each response choice and hand trace each resulting code segment to determine which one yields a value of `10` for the variable `k`. The initial value of `k` is `18`. The index variable `j` in the **`for`** statement increases from `1` to `5` by `1`.

Option A, which indicates that `k ← k - 1` should be the missing code, would result in a value of `13` for `k`. Thus, option A is an incorrect response.

Option B, which indicates that `k ← k - 3` should be the missing code, would result in a value of `9` for `k`. Thus, option B is an incorrect response.

Option C, which indicates that `k ← k - 4` should be the missing code, would result in a value of `10` for `k`. Thus, option C is a correct response.

Option D, which indicates that `k ← k - 6` should be the missing code, would result in a value of `6` for `k`. Thus, option D is an incorrect response.

Of the alternatives offered, only `k ← k - 4` results in a value of `10` for `k`. Therefore, **the correct response is option C**.

## 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
```

# Chapter 5



## Multiple-Choice Practice Questions

## SAMPLE MULTIPLE-CHOICE QUESTIONS

This section presents some sample test questions for you to review as part of your preparation for the test. To demonstrate how each competency may be assessed, each sample question is accompanied by the competency that it measures. While studying, you may wish to read the competency before and after you consider each sample question. Please note that the competency statements will not appear on the actual test.

An answer key follows the sample questions. The answer key lists the question number and correct answer for each sample test question. Please note that the answer key also lists the competency assessed by each question and that the sample questions are not necessarily presented in competency order.

The sample questions are included to illustrate the formats and types of questions you will see on the test; however, your performance on the sample questions should not be viewed as a predictor of your performance on the actual test.

**COMPETENCY 001**

1. Which of the following is the principal advantage of saving a word processing document in rich-text format?

   A. The document can be viewed in any Web browser

   B. A formatted document can be transferred between different applications

   C. The document can take up less space in memory

   D. A formatted document can be scanned for viruses when sent as an e-mail attachment

**COMPETENCY 001**

2. Which of the following would most likely be considered unacceptable use of information by a teacher?

   A. Using the school district's database to determine gender distribution in local schools

   B. Using the Internet history on a classroom computer to audit student Internet use

   C. Using students' personal data to create a mailing list for a local charity

   D. Using classroom records to determine recipients of academic awards

**COMPETENCY 002**

3. Students in a Texas classroom have been communicating with a class in New York by videoconference. The two classes find that the images they receive from each other occasionally freeze for up to 30 seconds before the video continues. This type of problem can most often be solved by

   A. increasing bandwidth.

   B. upgrading cameras.

   C. increasing video resolution.

   D. upgrading monitors.

**COMPETENCY 002**

4. Which of the following is the most appropriate format for graphics that are to be embedded within an Internet document?

   A. BMP

   B. TIFF

   C. GIF

   D. HTML

### COMPETENCY 002

5. Suppose that the class grade for a six-week period is based on 3 tests (T1, T2, T3), each of which counts for 15%, 4 quizzes (Q1, Q2, Q3, Q4), each of which counts for 10%, and a homework notebook (HW), which counts for 15%. The grades are recorded in a spreadsheet similar to the one below.

| | A | B | C | D | E | F | G | H | I | J |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | Name | T1 | T2 | T3 | Q1 | Q2 | Q3 | Q4 | HW | AVG |
| 2 | Jane | 87 | 92 | 80 | 76 | 79 | 87 | 74 | 90 | |
| 3 | Joe | 91 | 85 | 77 | 78 | 88 | 96 | 90 | 92 | |
| 4 | Bill | 65 | 72 | 70 | 80 | 81 | 74 | 77 | 80 | |
| 5 | Brenda | 96 | 88 | 91 | 76 | 91 | 100 | 74 | 98 | |

Which of the following formulas would NOT be a correct calculation of the six-week weighted average for Jane?

A. =B2*0.15+C2*0.15+D2*0.15+E2*0.1+F2*0.1+G2*0.1+H2*0.1+I2*0.15

B. =(B2+C2+D2+I2)*0.15+(E2+F2+G2+H2)*0.1

C. =((B2+C2+D2+I2)*1.5+(E2+F2+G2+H2))/10

D. =(B2+C2+D2+I2)/15+(E2+F2+G2+H2)/10

### COMPETENCY 003

6. A teacher has assigned students several topics to discuss outside of class using an electronic form of communication. The teacher wants the students' messages to be organized by topic and wants to have all historical messages available to students. To facilitate this type of communication most effectively, the teacher should have students

A. participate in a threaded discussion group.

B. send e-mail messages with attached document files.

C. update pages on the class's website.

D. engage in dialogue in a real-time chat room.

### COMPETENCY 004

7. Which of the following best describes the purpose of generating a flowchart as part of the design of a computer program?

A. To test and maintain the efficiency of the overall program

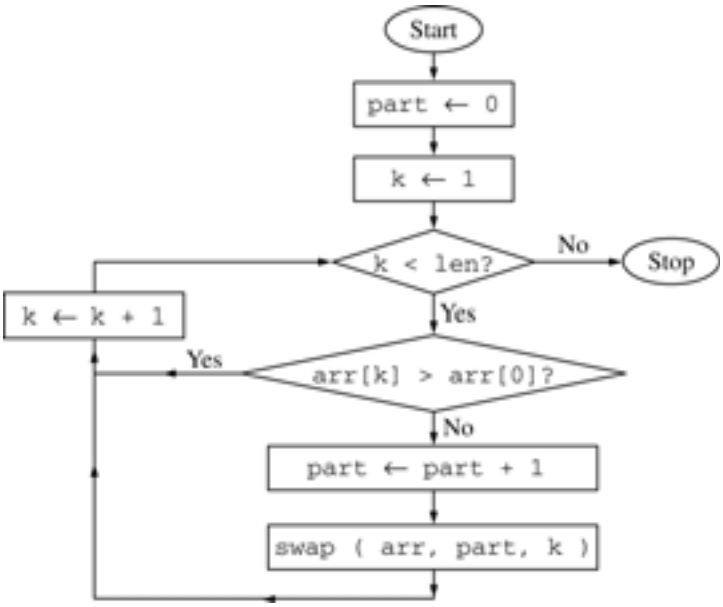
B. To present the steps needed to solve the programming problem

C. To ensure that all methods are appropriately linked

D. To determine the necessary number of global and local variables

**COMPETENCY 004**

8. Consider the following flowchart diagram, where `arr[0..len-1]` is an integer array of length `len`. Assume that the elements `arr[0]`, `arr[1]`, ..., `arr[len-1]` have already been initialized.




Which of the following pseudocode segments implements the algorithm in the flowchart?

A.
```
int part ← 0
int k ← 1
while ( k < len )
  if ( arr[k] ≤ arr[0] )
    part ← part + 1
    swap ( arr, part, k )
  end if
  k ← k + 1
end while
```

B.
```
int part ← 0
int k ← 1
while ( k < len )
  k ← k + 1
  if ( arr[k] ≤ arr[0] )
    part ← part + 1
    swap ( arr, part, k )
  end if
end while
```

C.
```
int part ← 0
int k ← 1
while ( k < len )
  if ( arr[k] > arr[0] )
    part ← part + 1
    swap ( arr, part, k )
  end if
  k ← k + 1
end while
```

D.
```
int part ← 0
int k ← 1
while ( k < len )
  k ← k + 1
  if ( arr[k] > arr[0] )
    part ← part + 1
    swap ( arr, part, k )
  end if
end while
```

COMPETENCY 004

9. Which of the following would best facilitate the expansion of a computer program?

   A. Incorporation of diagrams into the design documents

   B. Minimal use of global variables in the source code

   C. Construction of methods that are highly dependent on one another

   D. Extensive use of functions that support the debugging process

COMPETENCY 005

10. A software system is to be developed for which the requirements are well understood and the risk of failure is minimal. To meet these requirements, which of the following software development models would be most appropriate to use?

    A. Chaos

    B. Incremental

    C. Spiral

    D. Waterfall

COMPETENCY 005

11. The most appropriate way to use a library of program code is to access the

    A. methods or functions by way of the interface.

    B. implementation details of the methods or functions.

    C. methods or functions by way of the source code.

    D. documentation of the methods or functions.

COMPETENCY 005

12. Consider the following pseudocode segment with integer variables, where the precondition at the beginning of the segment is missing.

```
// missing precondition
x ← x + 1
y ← y + x
// postcondition:
// y == 2 * x
```

    Which of the following would be a valid precondition for the code segment above?

    A. `y == x - 1`

    B. `y == x`

    C. `y == x + 1`

    D. `y == x + 2`

COMPETENCY 006

13. Which of the following techniques is used by most programming languages to intercept events that disrupt the normal flow of a program's execution?

    A. Code security

    B. Flow control

    C. Exception handling

    D. Error detection

COMPETENCY 006

14. A multibyte data representation is stored in memory with its most significant byte in the lowest memory address. Which of the following describes this method of addressing?

    A. ASCII

    B. Big-endian

    C. NUXI ordering

    D. Huffman encoding

COMPETENCY 007

15. Which of the following is most efficient for manipulating a list that contains integers and is of predefined size?

    A. A stack

    B. A linked list

    C. An array

    D. A sequential file

COMPETENCY 007

16. A binary heap data structure is best represented conceptually using which of the following?

    A. A binary tree

    B. A graph

    C. A linked list

    D. A stack

COMPETENCY 007

17. Consider the following pseudocode procedure `calc`, where the first and second parameters are passed by value and the third and fourth parameters are passed by reference. That is, actual parameters passed to formal parameters `w` and `x` are passed by value, while those passed to formal parameters `y` and `z` are passed by reference.

```
procedure calc ( pass-by-value int w,
                 pass-by-value int x,
                 pass-by-reference int y,
                 pass-by-reference int z )
    w ← w + 1
    x ← x * 2
    y ← y + 3
    z ← z * 4
end procedure
```

What are the values of `a` and `b` at the end of the code fragment below?

```
int a ← 5
int b ← 6
calc ( a, a, b, b )
```

A. `a = 5` and `b = 24`

B. `a = 5` and `b = 36`

C. `a = 10` and `b = 6`

D. `a = 12` and `b = 6`

**COMPETENCY 007**

18. Consider a class `Stack` defined with methods `push ( x )`, `pop()`, and `peek()` that implement a stack data structure. (Note that `void push ( int x )` pushes the integer `x` onto the top of the stack; `int pop()` removes the integer at the top of the stack and returns that integer; `int peek()` returns the integer at the top of the stack without removing it from the stack.)

    Consider the following pseudocode fragment, where `S` is a `Stack` instance that will hold integers.

```
Stack S ← new Stack()
S.push ( 4 )
S.push ( 3 )
S.push ( S.peek() + S.peek() )
S.push ( S.pop() * S.pop() )
print ( S.peek() )
```

    What is printed by the last line of code?

    A. `18`

    B. `21`

    C. `28`

    D. `32`

COMPETENCY 008

19. Consider the following pseudocode functions, where each `print` statement prints on a separate line of output and then executes a line feed.

```
void f1 ( int n )
    int k ← 0
    do {
        k ← k + 1
        print k
    } while ( k < n )
end f1
```

```
void f2 ( int n )
    int k ← 0
    while ( k < n )
        k ← k + 1
        print k
    end while
end f2
```

Which of the following describes all the values of the input `n` for which functions `f1` and `f2` print the same sequence of numbers?

A. $n > 0$

B. $n \geq 0$

C. $n < 0$

D. $n \leq 0$

### COMPETENCY 008

20. Consider the following pseudocode fragment, where `x` is an integer variable initialized to a nonnegative integer value.

```
// x is a nonnegative integer
int sum
x ← x / 2     // integer division; truncates fractions
for ( sum ← 1; x > 0; x ← x / 2 )
    sum ← sum + 1
end for
```

Which of the following will calculate the same value of `sum` as the fragment above?

A.
```
int sum ← 0
x ← x / 2
while ( x ≥ 0 )
    sum ← sum + 1
    x ← x / 2
end while
```

B.
```
int sum ← 1
x ← x / 2
while ( x ≥ 0 )
    sum ← sum + 1
    x ← x / 2
end while
```

C.
```
int sum ← 0
do {
    sum ← sum + 1
    x ← x / 2
} while ( x > 0 )
```

D.
```
int sum ← 1
do {
    sum ← sum + 1
    x ← x / 2
} while ( x > 0 )
```

### COMPETENCY 009

21. Which of the following represents the average-case performance of a quicksort algorithm?

A. $O(n)$

B. $O(\log_2 n)$

C. $O(n^2)$

D. $O(n \log_2 n)$

COMPETENCY 009

22. Consider the following pseudocode function, where each **print** statement prints on a separate line of output and then executes a line feed.

```
void h ( int n )
    if ( n ≥ 4 )
       h ( n / 2 )
    end if
    print n
end h
```

What is printed when the call `h ( 16 )` is executed?

A. `2`

B. `16`

C.
```
16
8
4
2
```

D.
```
2
4
8
16
```

COMPETENCY 009

23. A specific sorting algorithm begins by finding the largest element and swapping that element with the last element. Which of the following sorting algorithms fits this description?

A. Quicksort

B. Insertion sort

C. Heapsort

D. Selection sort

**COMPETENCY 009**

24. Consider the following pseudocode binary search function, which returns the largest array index `k` such that `a[k]` ≤ `x`.

```
// precondition 1:  integer array a is sorted in
//                  ascending order
// precondition 2:  0 ≤ first < last < length of array a
// precondition 3:  a[first] ≤ x < a[last]
int f(int array a, int x, int first, int last)
    while ( first + 1 ≠ last )
        int mid ← ( first + last ) / 2    // integer division
        if ( x < a[mid] )
            last ← mid
        else
            first ← mid
        end if
    end while
    return first
end f
```

Consider the following (incomplete) equivalent recursive implementation of function `f`.

```
int f(int array a, int x, int first, int last)
    if ( first + 1 == last )
       return first
    end if
    int mid ← ( first + last ) / 2
    // missing code block
end f
```

Which of the following could replace the missing code block so that the recursive function will work as intended?

A.
```
if ( x ≥ a[mid] )
   return f(a, x, first, mid)
end if
return f(a, x, mid, last)
```

B.
```
if ( x ≥ a[mid] )
   return f(a, x, mid, first)
end if
return f(a, x, last, mid)
```

C.
```
if ( x ≥ a[mid] )
   return f(a, x, mid, last)
end if
return f(a, x, first, mid)
```

D.
```
if ( x ≥ a[mid] )
   return f(a, x, last, mid)
end if
return f(a, x, mid, first)
```

COMPETENCY 009

25. Consider the following pseudocode function.

```
// precondition:  n and k are nonnegative integers
int f ( int n, int k )
    if ( k * n == 0 )
        return 1
    else
        return f ( n - 1, k - 1 ) + f ( n - 1, k )
    end if
end f
```

What value is returned by the call `f ( 4, 2 )` ?

A. `4`

B. `5`

C. `7`

D. `11`

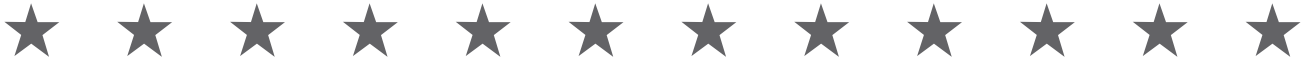
## ANSWER KEY

| Question Number | Correct Answer | Competency |
|---|---|---|
| 1 | B | 001 |
| 2 | C | 001 |
| 3 | A | 002 |
| 4 | C | 002 |
| 5 | D | 002 |
| 6 | A | 003 |
| 7 | B | 004 |
| 8 | A | 004 |
| 9 | B | 004 |
| 10 | D | 005 |
| 11 | A | 005 |
| 12 | C | 005 |
| 13 | C | 006 |
| 14 | B | 006 |
| 15 | C | 007 |
| 16 | A | 007 |
| 17 | B | 007 |
| 18 | A | 007 |
| 19 | A | 008 |
| 20 | C | 008 |
| 21 | D | 009 |
| 22 | D | 009 |
| 23 | D | 009 |
| 24 | C | 009 |
| 25 | D | 009 |

# Chapter 6

## Are You Ready? – Last-Minute Tips





## PREPARING TO TAKE THE TEST

### CHECKLIST

Complete this checklist to determine if you are ready to take your test.

- ✓ Do you know the testing requirements for your teaching field?
- ✓ Have you followed the test registration procedures?
- ✓ Have you reviewed the test center identification document requirements in the *Registration Bulletin* or on the ETS TExES website at **www.texes.ets.org**?
- ✓ Do you know the test frameworks that will be covered in each of the tests you plan to take?
- ✓ Have you used the study plan sheet at the end of this booklet to identify what content you already know well and what content you will need to focus on in your studying?
- ✓ Have you reviewed any textbooks, class notes and course readings that relate to the frameworks covered?
- ✓ Do you know how long the test will take and the number of questions it contains? Have you considered how you will pace your work?
- ✓ Are you familiar with the test directions and the types of questions for your test?
- ✓ Are you familiar with the recommended test-taking strategies and tips?
- ✓ Have you practiced by working through the sample test questions at a pace similar to that of an actual test?
- ✓ If constructed-response questions are part of your test, do you understand the scoring criteria for these questions?
- ✓ If you are repeating a test, have you analyzed your previous score report to determine areas where additional study and test preparation could be useful?

## THE DAY OF THE TEST

You should have ended your review a day or two before the actual test date. Many clichés you may have heard about the day of the test are true. You should:

- Be well rested.
- Take the appropriate identification document(s) with you to the test center (identification requirements are listed in the *Registration Bulletin* and on the ETS TExES website at **www.texes.ets.org**).
- Take 3 or 4 well-sharpened soft-lead (No. 2 or HD) pencils with good erasers.
- Eat before you take the test.
- Be prepared to stand in line to check in or to wait while other test takers are being checked in.
- Stay calm. You can't control the testing situation, but you can control yourself. Test administrators are well trained and make every effort to provide uniform testing conditions, but don't let it bother you if a test doesn't start exactly on time. You will have the necessary amount of time once it does start. Using the *Reducing Test Anxiety* booklet in the days before you test may be helpful in mentally and emotionally preparing yourself to test. It is available free at **www.texes.ets.org**.

You can think of preparing for this test as training for an athletic event. Once you have trained, prepared and rested, give it everything you've got. Good luck.

# Appendix A

## Study Plan Sheet

| STUDY PLAN | | | | | | |
|---|---|---|---|---|---|---|
| **Content covered on test** | **How well do I know the content?** | **What material do I have for studying this content?** | **What material do I need for studying this content?** | **Where can I find the materials I need?** | **Dates planned for study of content** | **Date completed** |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |

# Appendix B

## Preparation Resources

## PREPARATION RESOURCES

The resources listed below may help you prepare for the TExES test 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 to obtain information on specific topics for study and review.

### JOURNALS

*Connection Science,* Taylor & Francis Group — www.tandf.co.uk/journals/tf/09540091

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

*Journal of Computing in Teacher Education,* International Society for Technology in Education — www.iste.org

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

*Journal on Educational Resources in Computing,* Association for Computing Machinery — http://jeric.acm.org

*Texas Computer Education Association TechEdge and TechNotes* — www.tcea.org/publications

### STATE-ADOPTED INSTRUCTIONAL MATERIALS

www.tea.state.tx.us/textbooks

### 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**

Association for Computing Machinery — www.acm.org

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

Computer Science Teaching Center — www.citidel.org

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

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

State Board for Educator Certification, Standards and Testing — www.sbec.state.tx.us

State Board for Education Certification, Standards and Testing, Educator Standards — www.sbec.state.tx.us/SBECOnline/standtest/educstan.asp

Technology Applications Teacher Network — www.techappsnetwork.org

Texas Computer Education Association (TCEA) — www.tcea.org

Texas Computer Education Associate (TCEA) Tech-Apps/Computer Science Special Interest Group — www.tcea.org/SiGs

Texas Education Agency, Educational Technology — www.tea.state.tx.us/curriculum

Texas Education Agency, Technology Applications Curriculum — www.tea.state.tx.us/curriculum

Texas Education Agency, Texas Essential Knowledge and Skills (TEKS) — www.tea.state.tx.us/teks

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

83651-83651 • UNLPDF810
756107