Florida Department of Education
Curriculum Framework

Program Title: **Computer Science Principles**
Program Type: **Career Preparatory**
Career Cluster: **Information Technology**

Program Number	9007600
CIP Number	0511020316
Grade Level	9-12
Program Length	4 credits
Teacher Certification	Refer to the **Program Structure** section.
CTSO	FBLA, BPA, FL-TSA
SOC Codes (all applicable)	15-1151 – Computer User Support Specialist 15-1131 – Computer Programmers
CTE Program Resources	http://www.fldoe.org/academics/career-adult-edu/career-tech-edu/program-resources.stml

Purpose

This program offers a sequence of courses that provides coherent and rigorous content aligned with challenging academic standards and relevant technical knowledge and skills needed to prepare for further education and careers such as a Computer Users Support Specialists, Computer Programmer Assistants, Computer Network Architects, and Computer Systems Analysts in the Information Technology career cluster; provides technical skill proficiency, and includes competency-based applied learning that contributes to the academic knowledge, higher-order reasoning and problem-solving skills, work attitudes, general employability skills, technical skills, and occupation-specific skills, and knowledge of all aspects of the Information Technology career cluster.

The content includes but is not limited to practical experiences in computer programming, algorithms, program design structure, logical thinking, development methodologies, essential programming techniques, and implementation issues. Specialized programming skills involving advanced mathematical calculations and physics are also integrated into the curriculum

Additional Information relevant to this Career and Technical Education (CTE) program is provided at the end of this document.

Program Structure

This program is a planned sequence of instruction consisting of four (4) credits.

To teach the courses listed below, instructors must hold at least one of the teacher certifications indicated for that course.

The following table illustrates the secondary program structure:

Course Number	Course Title		Length	SOC Code	Level	Graduation Requirement
9007610	Advanced Information Technology	BUS ED 1 @2 COMPU SCI 6 COMP PROG 7G	1 credit	15-1151	3	CT
9007210	Foundations of Programming		1 credit	15-1131	3	CT
9007220	Procedural Programming		1 credit	15-1131	3	CT
0200335	**OR** *AP Computer Science Principles*	COMPU SCI 6 ANY FIELD WHEN CERT REFLECTS BACHELORS OR HIGHER	1 credit	15-1131	3	MA
9007230	Object-Oriented Programming Fundamentals**	BUS ED 1 @2 COMPU SCI 6 COMP PROG 7G	1 credit	15-1131	3	CT
0200320	**OR** *AP Computer Science A*	COMPU SCI 6 ANY FIELD WHEN CERT REFLECTS BACHELORS OR HIGHER	1 credit	15-1131	3	MA

(Graduation Requirement Codes: CT=Career & Technical Education, EQ= Equally Rigorous Science, EC= Economics, MA=Mathematics, PL=Personal Financial Literacy)

**Students should have a strong procedural programming knowledge base prior to enrolling in this course. The Procedural Programming course (9007220) or *AP Computer Science Principles* course are recommended to provide this knowledge base.

Common Career Technical Core – Career Ready Practices

Career Ready Practices describe the career-ready skills that educators should seek to develop in their students. These practices are not exclusive to a Career Pathway, program of study, discipline or level of education. Career Ready Practices should be taught and reinforced in all career exploration and preparation programs with increasingly higher levels of complexity and expectation as a student advances through a program of study.

1. Act as a responsible and contributing citizen and employee.

2. Apply appropriate academic and technical skills.

3. Attend to personal health and financial well-being.

4. Communicate clearly, effectively and with reason.

5. Consider the environmental, social and economic impacts of decisions.

6. Demonstrate creativity and innovation.

7. Employ valid and reliable research strategies.

8. Utilize critical thinking to make sense of problems and persevere in solving them.

9. Model integrity, ethical leadership and effective management.

10. Plan education and career path aligned to personal goals.

11. Use technology to enhance productivity.

12. Work productively in teams while using cultural/global competence.

Standards

After successfully completing this program, the student will be able to perform the following:

1.0 Develop an awareness of microprocessors and digital computers.
2.0 Demonstrate an understanding of computer operating systems.
3.0 Demonstrate an understanding of global and local network systems.
4.0 Incorporate appropriate leadership and supervision techniques, customer service strategies, and standards of personal ethics to accomplish job objectives and enhance workplace performance.
5.0 Demonstrate competence using computer networks, internet and online databases to facilitate collaborative or individual learning and communication.
6.0 Develop an awareness of emerging technologies.
7.0 Develop awareness of web development.
8.0 Demonstrate proficiency in physical computing with hardware devices or emulators.
9.0 Use oral and written communication skills in creating, expressing and interpreting information and ideas.
10.0 Explore the characteristics, tasks, work attributes, options, and tools associated with a career in software development.
11.0 Demonstrate an understanding of the characteristics, use, and selection of numerical, non-numerical, and logical data types.
12.0 Distinguish between iterative and non-iterative program control structures.
13.0 Describe the processes, methods, and conventions for software development and maintenance.
14.0 Explain the types, uses, and limitations of testing for ensuring quality control.
15.0 Create a program design document using common design tool.
16.0 Solve problems using critical thinking skills, creativity and innovation.
17.0 Describe the importance of security and privacy information sharing, ownership, licensure and copyright.
18.0 Create programs that solve a problem using non-iterative and iterative algorithms.
19.0 Design a computer program to meet specific physical, operational, and interaction criteria.
20.0 Create and document a computer program that uses a variety of internal and control structures for manipulating varied data types.
21.0 Create and document an interactive computer program that employs functions, subroutines, or methods to receive, validate, and process user input.
22.0 Effectively communicate and collaborate.
23.0 Demonstrate responsible use of technology and information.
24.0 Differentiate among procedural, object-oriented, compiled, interpreted, and translated programming languages.
25.0 Explain key concepts that distinguish object-oriented programming from procedural programming.
26.0 Create a project plan for an object-oriented programming project that defines requirements, structural design, time estimates, and testing elements.
27.0 Design, document, and create object-oriented computer programs.
28.0 Design a unit test plan for an object-oriented computer program, test and debug the program, and report the results.
29.0 Understand human-AI interaction.
30.0

Florida Department of Education
Student Performance Standards

Course Title: **Advanced Information Technology**
Course Number: **9007610**
Course Credit: **1**

Course Description:

This course provides a basic overview of current business and information systems and their trends. Students gain fundamental knowledge and experience in computer technology that is required for today's business and academic environments. With the development of basic computer science knowledge and understanding, this course prepares students to be successful both personally and professionally in an information-based society. Advanced Information Technology includes industry-driven standards that allow student exploration of computers and their networks, as well as other emergent technology, hardware/software installation and functionality, web development practices, and the benefits and risks of using computers both locally and globally.

CTE Standards and Benchmarks	
1.0	Develop an awareness of microprocessors and digital computers. The student will be able to:
1.1	Explain the general architecture of a microcomputer system.
1.2	Explain the need for and use of peripherals (e.g., keyboards, sensory input, geospatial input).
1.3	Demonstrate proficiency using peripherals.
1.4	Differentiate between diagnosing and troubleshooting peripherals.
1.5	Describe the necessary components for data storage and memory, and how it affects programming (e.g., RAM, ROM).
1.6	Differentiate between multiple levels of hardware and software (e.g., CPU hardware, operating system, translation, interpretation) that support program execution.
1.7	Evaluate various forms of input and output (e.g., IO, storage devices, digital media).
2.0	Demonstrate an understanding of computer operating systems. The student will be able to:
2.1	Identify various types of computer operating systems.
2.2	Compare and contrast various types of computer operating systems.
2.3	Describe the evolution of computer operating systems.
2.4	Compare and contrast different computer system viruses and how they affect various computer operating systems.
2.5	Understand the advantages and disadvantages of open-source computer operating systems.
3.0	Demonstrate an understanding of global and local network systems. The student will be able to:
3.1	Identify types of networks and how they work.
3.2	Identify the role of servers and clients on a network.
3.3	Identify benefits and risks of networked computing.
3.4	Identify the relationship between computer networks and other communications networks (e.g., Wi-Fi, teleconference, telepresence).
3.5	Identify intranets, extranets and how they relate to the Internet.

	3.6	Describe how the Internet facilitates global communication.
4.0	Incorporate appropriate leadership and supervision techniques, customer service strategies, and standards of personal ethics to accomplish job objectives and enhance workplace performance. The student will be able to:	
	4.1	Demonstrate awareness of the following workplace essentials: quality customer service; business ethics; confidentiality of information; copyright violations; accepted workplace rules, regulations, policies, procedures, processes, and workplace safety, and appropriate attire and grooming.
	4.2	Demonstrate ways of providing and accepting constructive criticism on collaborative projects within the workplace.
	4.3	Apply appropriate collaborative skills to manage and resolve conflicts in work situations.
	4.4	Demonstrate human relations, personal and interpersonal skills appropriate for the workplace, including responsibility, dependability, punctuality, integrity, positive attitude, initiative, respect for self and others, and professional dress.
	4.5	Discuss and analyze the impact of values and points of view that are presented in media message (e.g., racial, gender, political, biases).
5.0	Demonstrate competence using computer networks, internet and online databases to facilitate collaborative or individual learning and communication. The student will be able to:	
	5.1	Demonstrate how to connect to the Internet and use appropriate Internet protocol.
	5.2	Identify and describe web terminology, addresses and how browsers work.
	5.3	Describe appropriate browser security configurations.
	5.4	Understand and apply level one Universal Resource Locator (URL) and associated protocols (e.g., com, org, edu, gov, net, mil).
	5.5	Evaluate quality of digital resources for reliability (e.g., currency, relevancy, authority, accuracy, purpose of digital information).
	5.6	Compare and contrast techniques for analyzing massive data collections.
6.0	Develop an awareness of emerging technologies. The student will be able to:	
	6.1	Compare and contrast emerging technologies and describe how they impact business in the global marketplace (e.g., wireless network, tablets, cell phones, satellite technology, nanotechnology, smart devices, home networks, peer-to-peer).
	6.2	Describe how digital tools and resources are used in today's society (i.e., efficiency and effectiveness, individual and collaborative use).
	6.3	Explain different file types used for various purposes based on file size and data input (e.g., word processing, images, music, three-dimensional drawings).
	6.4	Develop criteria for selecting appropriate hardware and software when solving a specific real-world problem.
	6.5	Define scale-ability as it relates to emerging technology.
7.0	Develop awareness of web development. The student will be able to:	
	7.1	Define basic terminology used in web page development.
	7.2	Create web pages using HTML tags (e.g., headings, character styles, paragraphs, text alignments, lists, images)
	7.3	Describe the purpose of storyboarding techniques.
	7.4	Describe the basic functions of WYSIWYG editors.
	7.5	Create a simple example of wire framing with, at least, three web pages.
	7.6	Explain the use of Cascading Style Sheets.
	7.7	Apply the use of Cascading Style Sheets in a web page.

	7.8	Test web pages for display, functionality, and accessibility before publishing a site to the Internet (i.e., validate web page code using W3C validation tool).
	7.9	Discuss issues related to using music, videos, or images from the Internet on your website (e.g., ethical use, illegal use, Creative Commons).
8.0	Demonstrate proficiency in physical computing with hardware devices or emulators. The student will be able to:	
	8.1	View hardware as an approachable and fun topic in computing. *Physical computing is meant to encourage interdisciplinary and entrepreneurial thinking and foster student creativity.*
	8.2	Demonstrate the use of Physical computing, which is about the interaction between the person and the machine.
	8.3	Use physical computing devices or emulators to solve problems.
	8.4	Determine how computers sense and respond to their environment.
	8.5	Determine the kind of information that can be communicated with hardware outputs.
	8.6	Analyze how simple hardware can be used to develop innovative new products.
	8.7	Define prototype in relation to digital design.
	8.8	Create a prototype of an original game that can be played using a physical computing device.
	8.9	Design a prototype of an original device that can be used to assist someone with a physical challenge.

Florida Department of Education
Student Performance Standards

Course Title: **Foundations of Programming**
Course Number: **9007210**
Course Credit: **1**

Course Description:

This course introduces concepts, techniques, and processes associated with computer programming and software development.

CTE Standards and Benchmarks		
9.0	Use oral and written communication skills in creating, expressing and interpreting information and ideas. The student will be able to:	
	9.1	Select and employ appropriate communication concepts and strategies to enhance oral and written communication in the workplace.
	9.2	Locate, organize and reference written information from various sources.
	9.3	Construct writings and/or communications using developmentally appropriate terminology.
	9.4	Analyze the positive and negative impacts of technology on popular culture and personal life.
	9.5	Discuss how technology has changed the way people build and manage organizations and how technology impacts personal life.
	9.6	Evaluate ways in which adaptive technologies may assist users with special needs.
	9.7	Explain how societal and economic factors are affected by access to critical information.
	9.8	Discuss the challenges (e.g., political, social, and economic) in providing equal access and distribution of technology in a global society.
10.0	Explore the characteristics, tasks, work attributes, options, and tools associated with a career in software development. The student will be able to:	
	10.1	Explore a variety of careers to which computing is central.
	10.2	Discuss the impact of computing on business and commerce (e.g., automated inventory processing, financial transactions, e-commerce, virtualization, and cloud computing).
	10.3	Evaluate the impacts of irresponsible use of information (e.g., plagiarism and falsification of data) on collaborative projects.
	10.4	Identify tasks performed by programmers.
	10.5	Describe how businesses use computer programming to solve business problems.
	10.6	Investigate job opportunities in the programming field.
	10.7	Explain different specializations and the related training in the computer programming field.
	10.8	Explain the need for continuing education and training of computer programmers.
	10.9	Understand and identify ways to use technology to support lifelong learning.
	10.10	Explain software as a service (SaaS) and how it impacts business.
	10.11	Describe ethical responsibilities of computer programmers.
	10.12	Identify credentials and certifications that may improve employability for a computer programmer.
	10.13	Identify devices, tools, and other environments for which programmers may develop software.

1.0	Demonstrate an understanding of the characteristics, use, and selection of numerical, non-numerical, and logical data types. The student will be able to:	
	1.1	Identify the characteristics (e.g., size, limits) and uses of different numerical and non-numerical data types.
	1.2	Explain the types and uses of variables in programs.
	1.3	Determine the best data type to use for given programming problems.
	1.4	Compare and contrast simple data structures and their uses.
	1.5	Identify the types of operations that can be performed on different data types (e.g., math operations on numerical data types, concatenation, and other string operations).
	1.6	Evaluate arithmetic and logical expressions using appropriate operator precedence.
	1.7	Explain how computers store different data types in memory.
	1.8	Demonstrate the difference between "data" and "information".
	1.9	Use different number systems to represent data.
	1.10	Explain how national and international standards (i.e., ASCII, UNICODE) are used to represent non-numerical data.
	1.11	Use Boolean logic to perform logical operations using Boolean algebra and truth tables.
2.0	Distinguish between iterative and non-iterative program control structures. The student will be able to:	
	2.1	Identify the uses of non-iterative and iterative programming structures using pseudocode and flowcharts.
	2.2	Create iterative programming structures and their uses.
	2.3	Explain how sequence, selection, and iteration are building blocks of algorithms.
3.0	Describe the processes, methods, and conventions for software development and maintenance. The student will be able to:	
	3.1	Describe a software development process that is used to solve problems at different software development stages.
	3.2	Define alternative methods of program development (e.g., rapid prototyping, waterfall, spiral model, peer coding).
	3.3	List and explain the steps in the program development cycle.
	3.4	Describe different types of documentation used in the program development cycle (*e.g.*, requirements document, program design documents, test plans).
	3.5	Describe different methods used to facilitate version control.
4.0	Explain the types, uses, and limitations of testing for ensuring quality control. The student will be able to:	
	4.1	Explain the uses and limits of testing in ensuring program quality.
	4.2	Explain testing performed at different stages of the program development cycle (*e.g.,* unit testing, system testing, user acceptance testing).
	4.3	Describe and identify types of programming errors.
5.0	Create a program design document using common design tool. The student will be able to:	
	5.1	Describe different design methodologies and their uses (*e.g.*, object-oriented design, structured design, rapid application development).
	5.2	Describe and use tools for developing a program design (*e.g.*, flowcharts, design documents, pseudocode).
	5.3	Explain the role of existing libraries and packages in facilitating programmer productivity.
	5.4	Participate and contribute to a design review of a program design developed using a common program design tool (*e.g.*, UML, flowcharts, design documents, pseudocode).

	5.5	Develop a software artifact (independently and collaboratively) in phases (or stages) according to a common software development methodology (e.g., Waterfall or Spiral model).
	5.6	Define input and output for a program module using standard design methodology.
6.0	Solve problems using critical thinking skills, creativity and innovation. The student will be able to:	
	6.1	Employ critical thinking skills independently and in teams to solve problems and make decisions.
	6.2	Employ critical thinking and collaborative skills to resolve conflicts.
	6.3	Identify and document workplace performance goals and monitor progress toward those goals.
	6.4	Conduct technical research to gather information necessary for decision-making.
	6.5	Discuss digital tools or resources to use for a real-world task based on their efficiency and effectiveness, individually and collaboratively.
7.0	Describe the importance of security and privacy information sharing, ownership, licensure and copyright. The student will be able to:	
	7.1	Describe security and privacy issues that relate to computer networks including the permanency of data on the Internet, online identity, and privacy.
	7.2	Discuss the impact of government regulation on privacy and security.
	7.3	Describe how different types of software licenses (e.g., open source and proprietary licenses) can be used to share and protect intellectual property.
	7.4	Explain how access to information may not include the right to distribute the information.
	7.5	Describe differences between open source, freeware, and proprietary software licenses, and how they apply to different types of software.
	7.6	Discuss security and privacy issues that relate to computer networks.
	7.7	Identify computer-related laws and analyze their impact on digital privacy, security, intellectual property, network access, contracts, and harassment.
1.0	Create programs that solve a problem using non-iterative and iterative algorithms. The student will be able to:	
	1.1	Apply the developmental cycle methodologies to create a program.
	1.2	Develop a program using string and/or numeric data types.
	1.3	Develop a program using sequential algorithms.
	1.4	Develop a program using selection structures.
	1.5	Develop a program using looping structures.

Florida Department of Education
Student Performance Standards

Course Title: **Procedural Programming**
Course Number: **9007220**
Course Credit: **1**

Course Description:

This course continues the study of computer programming concepts with a focus on the creation of software applications employing procedural programming techniques.

CTE Standards and Benchmarks	
2.0	Design a computer program to meet specific physical, operational, and interaction criteria. The student will be able to:
2.1	Choose appropriate data types depending on the needs of the program.
2.2	Define appropriate user prompts for clarity and usability (e.g., user guidance for data ranges, data types).
2.3	Design and develop program for efficiency (e.g., less memory usage, less inputs/outputs, faster processing).
2.4	Compare techniques for analyzing massive data collections.
2.5	Identify the software environment required for a program to run (e.g., operating system required, mobile, web-based, desktop, delivery method).
2.6	Create mobile computing applications and/or dynamic webpages through the use of a variety of design and development tools, programming languages and mobile devices/emulators.
2.7	Explain the role of an application programming interface (API) in the development of applications and the distinction between a programming language's syntax and the API.
2.8	Identify the tools required to develop a program (e.g., editors, compilers, linkers, integrated development environments, APIs, libraries).
3.0	Create and document a computer program that uses a variety of internal and control structures for manipulating varied data types. The student will be able to:
3.1	Use appropriate naming conventions to define program variables and methods.
3.2	Use a program editor to write the source code for a program.
3.3	Write programs that use selection structures.
3.4	Write programs that use repetition structures.
3.5	Write programs that use nested structures.
3.6	Use internal documentation (e.g., single-line and multi-line comments, program headers, module descriptions, meaningful variable and function/module names) to document a program according to accepted standards.
3.7	Compile, run, test and debug programs.
3.8	Write programs that use standard arithmetic operators with different numerical data types.
3.9	Write programs that use standard logic operators.

	3.10	Write programs that use a variety of common data types.
	3.11	Write programs that perform data conversion between numeric and string data types.
	3.12	Write programs that define, use, search, and sort arrays.
	3.13	Write programs that use user-defined data types.
	3.14	Demonstrate understanding and use of appropriate variable scope.
	3.15	Use global and local scope appropriately in program implementation.
	3.16	Distinguish between binary and sequential searches.
1.0	Create and document an interactive computer program that employs functions, subroutines, or methods to receive, validate, and process user input. The student will be able to:	
	1.1	Determine the results of code segments.
	1.2	Write programs that perform user input and output.
	1.3	Write programs that validate user input (*e.g.*, range checking, data formats, valid/invalid characters).
	1.4	Write program modules such as functions, subroutines, or methods.
	1.5	Write program modules that accept arguments.
	1.6	Write program modules that return values.
	1.7	Write program modules that validate arguments and return error codes.
	1.8	Design and implement a simple simulation algorithm to analyze, represent and understand natural phenomena.
	1.9	Use APIs and libraries to facilitate programming solutions.
	1.10	Participate in a peer code review to verify program functionality, programming styles, program usability, and adherence to common programming standards.
	1.11	Explain how abstraction manages complexity.
2.0	Effectively communicate and collaborate. The student will be able to:	
	2.1	Evaluate modes of communication and collaboration.
	2.2	Select appropriate tools within a project environment to communicate with project team members.
	2.3	Utilize project collaboration tools (such as version control systems and integrated development environments) while working on a collaborative software project.
	2.4	Generate, evaluate, and prioritize questions that can be researched through digital resources and online tool.
	2.5	Perform advanced searches to locate information and/or design a data-collection approach to gather original data.
	2.6	Communicate and publish key ideas and details to a variety of audiences using digital tools and media-rich resources.
3.0	Demonstrate responsible use of technology and information. The student will be able to:	
	3.1	Implement an encryption, digital signature, or authentication method.
	3.2	Describe computer security vulnerabilities and methods of attack, and evaluate their social and economic impact on computer systems and people (e.g., phishing, keylogging, virus, malware, intercepting data over public networks).
	3.3	Identify and explain the existence of biases in computer programming.
	3.4	Explain how computing can play a role in social and political issues.
4.0	Differentiate among procedural, object-oriented, compiled, interpreted, and translated programming languages. The student will be able to:	
	4.1	Differentiate between multiple levels of operating system, translation, and interpretation that support program execution.
	4.2	Explain the program execution process (by an interpreter and in CPU hardware).

4.3	Describe object-oriented concepts.
4.4	Explain the characteristics of procedural and object-oriented programming languages.
4.5	Compare and contrast programming languages that are compiled, interpreted, and translated.

Florida Department of Education
Student Performance Standards

Course Title: **Object-Oriented Programming Fundamentals**
Course Number: **9007230**
Course Credit: **1**

Course Description:

This course continues the study of computer programming concepts with a focus on the creation of software applications employing object-oriented programming techniques.

CTE Standards and Benchmarks	
5.0	Explain key concepts that distinguish object-oriented programming from procedural programming. The student will be able to:
5.1	Demonstrate the understanding and use of classes, objects, attributes, and behaviors.
5.2	Demonstrate the understanding and use of inheritance.
5.3	Demonstrate the understanding and use of data encapsulation.
5.4	Demonstrate the understanding and use of polymorphism.
5.5	Use predefined functions and parameters, classes, and methods to divide a complex problem into simpler parts by using the principle of abstraction to manage complexity (e.g., by using searching and sorting as abstractions).
6.0	Create a project plan for an object-oriented programming project that defines requirements, structural design, time estimates, and testing elements. The student will be able to:
6.1	Write a project plan for completion of a project that includes gathering program requirements, developing the program, and testing it.
6.2	Write a program requirements document that identifies business purpose, functional requirements, system requirements, and other common components of a requirements document.
6.3	Design an object-oriented program using standard design methodology.
6.4	Work with other team members to develop a project plan for a program.
6.5	Work with other team members to write a design document for a program with multiple functions and shared data.
6.6	Participate in design meetings that review program design documents for conformance to program requirements.
6.7	Estimate the time to develop a program or module.
6.8	Evaluate algorithms by their efficiency, correctness, and clarity (e.g., by analyzing and comparing execution times, testing with multiple inputs or data sets, and by debugging).
7.0	Design, document, and create object-oriented computer programs. The student will be able to:
7.1	Compare and contrast recursive functions to other iterative methods.
7.2	Understand the implementation of character strings in the programming language.
7.3	Write programs that perform string processing (e.g., manipulating, comparing strings, concatenation).
7.4	Write programs that implements user-defined data types.
7.5	Decompose a problem by defining new functions and classes.

	7.6	Write object-oriented programs that implement inheritance.
	7.7	Write object-oriented programs that implement polymorphism.
	7.8	Develop class constructors.
	7.9	Write programs that define and use program constants.
	7.10	Write programs that perform error handling.
	7.11	Participate in program code review meetings to evaluate program code for validity, quality, performance, data integrity, and conformance to program design documents.
	7.12	Describe the concept of parallel processing as a strategy to solve large problems.
	7.13	Demonstrate concurrency by separating processes into threads of execution and dividing data into parallel streams.
	7.14	Update a program module to implement enhancements or corrections and demonstrate appropriate documentation (internal and external) related to version control.
	7.15	Write programs that are event-driven.
	7.16	Write programs that perform file input and output (i.e., sequential and random-access file input/output).
	7.17	Explain the value of heuristic algorithms to approximate solutions for unmanageable problems (e.g., a heuristic solution to Towers of Hanoi).
8.0	Design a unit test plan for an object-oriented computer program, test and debug the program, and report the results. The student will be able to:	
	8.1	Develop a test plan for an object-oriented program.
	8.2	Write test plans for programs that perform file input and output.
	8.3	Perform test and debug activities on object-oriented programs, including those written by someone else.
	8.4	Perform test and debug activities on programs that perform file input and output and verify the correctness of output files.
	8.5	Document the findings of testing in a test report.
9.0	Understand human-AI interaction. The student will be able to:	
	9.1	Describe the unique features of computers embedded in mobile devices and vehicles.
	9.2	Describe the common physical and cognitive challenges faced by users when learning to use software and hardware.
	9.3	Describe the process of designing software to support specialized forms of human-computer interaction.
	9.4	Explain the notion of intelligent behavior through computer modeling and robotics.
	9.5	Describe common measurements of machine intelligence (e.g., Turing test).
	9.6	Describe a few of the major branches of artificial intelligence (e.g., expert systems, natural language processing, machine perception, machine learning).
	9.7	Describe major applications of artificial intelligence and robotics, including, but not limited to, the medical, space, and automotive fields.

Additional Information

Laboratory Activities

Laboratory investigations that include scientific inquiry, research, measurement, problem solving, emerging technologies, tools and equipment, as well as, experimental, quality, and safety procedures are an integral part of this career and technical program/course. Laboratory investigations benefit all students by developing an understanding of the complexity and ambiguity of empirical work, as well as the skills required to manage, operate, calibrate and troubleshoot equipment/tools used to make observations. Students understand measurement error; and have the skills to aggregate, interpret, and present the resulting data. Equipment and supplies should be provided to enhance hands-on experiences for students.

Florida Standards for English Language Development (ELD)

English language learners communicate for social and instructional purposes within the school setting. ELD.K12.ELL.SI.1

English Language Development (ELD) Standards Special Notes:
Teachers are required to provide listening, speaking, reading and writing instruction that allows English language learners (ELL) to communicate for social and instructional purposes within the school setting. For the given level of English language proficiency and with visual, graphic, or interactive support, students will interact with grade level words, expressions, sentences and discourse to process or produce language necessary for academic success. The ELD standard should specify a relevant content area concept or topic of study chosen by curriculum developers and teachers which maximizes an ELL's need for communication and social skills. For additional information on the development and implementation of the ELD standards, please contact the Bureau of Student Achievement through Language Acquisition at sala@fldoe.org .

Career and Technical Student Organization (CTSO)

Future Business Leaders of America (FBLA), Business Professionals of America (BPA) and Florida Technology Student Association (FL-TSA) are the co-curricular career and technical student organizations providing leadership training and reinforcing specific career and technical skills. Career and Technical Student Organizations provide activities for students as an integral part of the instruction offered.

Cooperative Training – OJT

On-the-job training is appropriate but not required for this program. Whenever offered, the rules, guidelines, and requirements specified in the OJT framework apply.

Accommodations

Federal and state legislation requires the provision of accommodations for students with disabilities as identified on the secondary student's Individual Educational Plan (IEP) or 504 plan or postsecondary student's accommodations' plan to meet individual needs and ensure equal access. Accommodations change the way the student is instructed. Students with disabilities may need accommodations in such areas as instructional methods and materials, assignments and assessments, time demands and schedules, learning environment, assistive technology and special communication systems. Documentation of the accommodations requested and provided should be maintained in a confidential file.

In addition to accommodations, some secondary students with disabilities (students with an IEP served in Exceptional Student Education (ESE)) will need modifications to meet their needs. Modifications change the outcomes or what the student is expected to learn, e.g., modifying the curriculum of a secondary career and technical education course. Note: postsecondary curriculum and regulated secondary programs cannot be modified.

Some secondary students with disabilities (ESE) may need additional time (i.e., longer than the regular school year), to master the student performance standards associated with a regular course or a modified course. If needed, a student may enroll in the same career and technical course more than once. Documentation should be included in the IEP that clearly indicates that it is anticipated that the student may need an additional year to complete a Career and Technical Education (CTE) course. The student should work on different competencies and new applications of competencies each year toward completion of the CTE course. After achieving the competencies identified for the year, the student earns credit for the course. It is important to ensure that credits earned by students are reported accurately. The district's information system must be designed to accept multiple credits for the same course number for eligible students with disabilities.