Department of Education | College, Career and Technical Education

May 2023

Coding I

Primary Career Cluster:	Information Technology
Course Contact:	CTE.Standards@tn.gov
Course Code(s):	C10H14
Prerequisite(s):	*Algebra I* (G02X02, G02H00), *Computer Science Foundations* (C10H11)
Credit:	1
Grade Level:	10
Focus Elective Graduation Requirements:	This course satisfies one of three credits required for an elective focus when taken in conjunction with other *Information Technology* courses.
Program of Study (POS) Concentrator:	This course satisfies one out of two required courses that meet the Perkins V concentrator definition, when taken in sequence in the approved program of study.
Programs of Study and Sequence:	This is the second course in the *Coding* program of study.
Aligned Student Organization(s)	SkillsUSA: http://www.skillsusatn.org/ Technology Student Association (TSA): http://www.tntsa.org
Coordinating Work-Based Learning:	Teachers are encouraged to use embedded WBL activities such as informational interviewing, job shadowing, and career mentoring. For information, visit https://www.tn.gov/education/educators/career-and-technical-education/work-based-learning.html.
Promoted Tennessee Student Industry Credentials	Credentials are aligned with post-secondary and employment opportunities and with the competencies and skills that students acquire through their selected program of study. For a listing of promoted student industry credentials, visit https://www.tn.gov/content/tn/education/educators/career-and-technical-education/student-industry-certification.html.
Teacher Endorsement(s):	037, 041, 055, 056, 057, 152, 153,173, 203, 204, 311, 413, 434, 435, 436, 470, 474, 475, 476, 477, 582, 595, 596, 740, 742, 952, 953
Required Teacher Certifications/Training:	All endorsements except for 173 and 742 will require either the NOCTI test code 5906: Computer Programming certification **or** the equivalent of twelve semester hours of computer course work including at least six hours of programming language.
Teacher Resources:	https://www.tn.gov/education/educators/career-and-technical-education/career-clusters/cte-cluster-information-technology.html Best For All Central: https://bestforall.tnedu.gov/

Course at a Glance

CTE courses provide students with an opportunity to develop specific academic, technical, and 21st century skills necessary to be successful in career and in life. In pursuit of ensuring every student in Tennessee achieves this level of success, we begin with rigorous course standards which feed into intentionally designed programs of study.

Students engage in industry relevant content through general education integration and experiences such as career and technical student organizations (CTSO) and work-based learning (WBL). Through these experiences, students are immersed with industry standard content and technology, solve industry-based problems, meaningfully interact with industry professionals, and use/produce industry specific, informational texts.

Using a Career and Technical Student Organization (CTSO) in Your Classroom

CTSOs are a great resource to put classroom learning into real-life experiences for your students through classroom, regional, state, and national competitions, and leadership opportunities. Below are CTSO connections for this course, note this is not an exhaustive list.

- Participate in CTSO Fall Leadership Conference to engage with peers by demonstrating logical thought processes and developing industry specific skills that involve teamwork and project management.
- Participate in contests that highlight job skill demonstration, interviewing skills, community service activities, extemporaneous speaking, and job interview.
- Participate in leadership activities such as Student2Student Mentoring, National Week of Service, Officer Training, and Community Action Project.

For more ideas and information, visit Tennessee SkillsUSA at http://www.skillsusatn.org/.

Using Work-Based Learning (WBL) in Your Classroom

Sustained and coordinated activities that relate to the course content are the key to successful work-based learning. Possible activities for this course include the following. This is not an exhaustive list.

- **Standards 1.1-2.1** | Invite a computer programmer to give a job overview and discuss ethical issues faced in the industry.
- **Standards 3.1-3.7** | Have students partner with a computer programmer on real project.
- **Standards 4.1-4.5** | Have students visit a local industry and discuss project planning and quality assurance.

For more ideas and information, visit https://www.tn.gov/education/educators/career-and-technical-education/work-based-learning.html.

Course Description

Coding I is a course intended to teach students the basics of computer programming. The course places emphasis on practicing standard programming techniques and learning the logic tools and methods typically used by programmers to create simple computer applications. Upon completion of this course, proficient students will be able to solve problems by planning multistep procedures; write, analyze, review, and revise programs, converting detailed information from workflow charts and diagrams into coded instructions in a computer language; and will be able to troubleshoot/debug programs and software applications to correct malfunctions and ensure their proper execution.

Course Standards

1. Computer Programming Overview

1.1 Development of Computers and Logical Devices: Using news articles and instructional materials**, investigate key milestones in the development of computers and logical devises**. Create and present a document and/or illustration depicting the timeline of development that led to modern-day operating systems, programmable controllers, and widespread digital communications via the Internet and wireless networks, citing specific textual evidence.

1.2 Programming Language: Compare and contrast the **benefits, features, and typical applications of common modern programming languages and environments**. Craft an argument to defend the choice of a certain language to solve a particular problem, developing claim(s) and counterclaim(s) with specific textual evidence and reasoning.

2. Ethics

2.1 Ethical Programming Practices: Using news articles and text of legislation, analyze **ethical programming practices,** including but not limited to the **issues of confidentiality, privacy, piracy, fraud and misuse, liability, copyright, open source software, trade secrets, and sabotage.** For example, research and report on the effects of unethical programming practices on a business.

3. Programming Skills

3.1 System Level and Application Practices: Differentiate between **system-level and application solutions, and identify an appropriate code-based strategy to solve a given problem**. For example, given a file management problem, determine when a command-line script will be more efficient than a high-level program solution.

3.2 System Management Tools: Apply the **system management tools present in a programming development environment** to:

a. Select the most appropriate programming language for the task at hand.
b. Develop syntactically correct program code using current best practices and emerging classes of development techniques.
c. Use a compiler to interpret the source code and produce executable program code.

3.3 Developing and Implementing Strategies: In the process of developing and implementing programming solutions, **develop strategies that work within the constraints of major operating system fundamentals**, such as:

a. security protocols and procedures for accessing files and folders;
b. file management syntax requirements, including but not limited to creating, naming, organizing, copying, moving, and deleting files; and
c. file naming conventions, as they apply across multiple software applications and file types.

3.4 Flowchart: Write pseudocode and **construct a flowchart for a process before starting to develop the program code**. For example, code and flowchart a simple process that takes an integer and report whether it is odd or even.

3.5 Data Values: Organize and develop a plan to **acquire and manage the data values for a process**, including the following:

a. data types, such as string, numeric, character, integer, and date;
b. program variable names;
c. variables and constants;
d. arrays (at least one- and two-dimensional) and subscripts;
e. input from files and user responses; and
f. output to files and reports.

3.6 Code Design: Using a programming language specified by the instructor, **convert the pseudocode for a selected process to program code**, incorporating at least three of the following structures, the need for which will be dictated by the assigned problem(s) and process(es). The resulting code design can be event-driven, object-oriented, or procedural.

a. Operations and functions (user-defined and/or library)
b. Repetition (loops)
c. Decision (if…else, case)
d. Recursion

3.7 Operation of Program Code: **Verify the correct operation of the resulting program code with several test cases**:

a. all valid values,
b. error trapping of invalid values,
c. error trapping of invalid program operation, and
d. troubleshooting/remedying program problems.

4. Project Planning and Quality Assurance

4.1 Computer Programming Problem and Client Specifications: Compile the necessary documentation to **understand the nature of a computer programming problem and the customer/client specifications** for the request and summarize in an informational text. This will include evidence of the scope of the problem, its attendant input and output information, the required system processing, and the software specifications involved.

4.2 Project Plan: Analyze a given problem and **develop a coherent strategy in the form of a project plan to meet the customer/client's need**. The plan will include, but will not be limited to, defining the project scope as addressed by the problem documentation, identifying software development and implementation issues, timeline and benchmarks for design, and addressing issues associated with software maintenance and life cycle.

4.3 Nature of the Program: In the software development process, **articulate the nature of the program designs by creating documentation that addresses topics** including but not limited to:

a. the procedural, object-oriented, event-driven, or other nature of the various portions of the resulting application;
b. the data structures used for inputs, outputs, and internal manipulations;
c. the algorithms and guiding formulas used;
d. constraints on accurate operation and results;
e. modular designs that enable portability; and
f. interface details that permit ready maintenance and upkeep.

4.4 Quality Assurance: Apply **principles of quality assurance during application development to certify bug tracking, audit trails, testing results, and other quality considerations**. Annotate each quality assurance task with evidence from best practices endorsed by industry or research.

4.5 Security Risks: Document the **security risks associated with new applications and evaluate the severity of the risk involved in each**, including but not limited to:

a. identifying threats to information systems facilities, data communications systems, and other applications;
b. adhering to federal and state legislation pertaining to computer crime, fraud, and abuse;
c. providing means for preserving confidentiality and encryption of sensitive data; and
d. detailing steps to recover from routine errors or catastrophic failures, such as might be caused by a malicious computer virus.

Standards Alignment Notes

*References to other standards include:

- P21: Partnership for 21st Century Skills Framework for 21st Century Learning
    - Note: While not all standards are specifically aligned, teachers will find the framework helpful for setting expectations for student behavior in their classroom and practicing specific career readiness skills.