Coding I - NSS

Level: 2
Career Cluster: Advanced Manufacturing, Digital Technology
Program: Engineering, Information Technology
Pathway(s): Engineering Applications UAS, Engineering, Coding, Networking Systems & Security

Course Catalog Description

Introduces students to the foundations of computer programming through networking systems and security applications. Students use computational thinking, program design, data handling, and debugging strategies to create simple programs that support network operations, system monitoring, secure access, and problem solving in technical environments.

Summary

Teaches students the foundations of computer programming through networking systems and security applications. The course emphasizes computational thinking, program design, and standard programming techniques as students create simple tools that relate to network communication, device configuration, system monitoring, secure data handling, and technical troubleshooting. Upon completion of this course, proficient students will be able to plan algorithms using flowcharts and pseudocode, write and revise programs in a computer language, manage data and files, and test and debug code to improve accuracy, efficiency, reliability, and security in real-world technical contexts. Students also explore how coding supports network administration, automation, cybersecurity awareness, log analysis, and data-driven decision-making across technical fields. Emphasis is placed on logical thinking, precision, responsible computing, and clear technical communication as students build foundational skills for more advanced study in networking, cybersecurity, systems administration, and related information technology pathways.

Notes

This revised document more intentionally aligns the course to the Networking Systems & Security pathway. The updated version retains the foundational programming content from the original course—such as pseudocode, flowcharting, variables, data types, file handling, conditionals, loops, and debugging—but reframes these skills through networking and security-related applications. New language throughout the course catalog description, summary, objectives, and student learning outcomes emphasizes network operations, system monitoring, secure communication, data protection, automation, log analysis, credential validation, and responsible computing in technical environments. In addition, the revised document presents a more structured and measurable set of student learning outcomes, adds stronger connections to cybersecurity awareness and systems support, and creates a clearer bridge from introductory coding into higher-level coursework in networking, cybersecurity, and information technology systems.

Student Learning Outcomes

1. Career Exploration and Postsecondary Opportunities

Objective: *Students will analyze career pathways in information technology, exploring industry roles, workplace expectations, required education, certifications, professional organizations, and postsecondary opportunities available in Colorado in order to make informed academic and career decisions.*

1) Identify employment opportunities in various fields with a focus in the area of information technology
    a. Recognize the work typically performed, tools and technology used, and nature of work environments
    b. Identify potential certifications within the careers
    c. Find membership organizations associated with the careers
    d. Understand the necessary education associated within the careers
2) Define professionalism within the context of information technology
    a. Criticism and evaluation
    b. Presentation
    c. Assessment
3) Recognize laws, regulations, and ethics significant to the fields of information technology
4) Identify postsecondary opportunities offered in the state of Colorado
    a. Technical college, community college, 4-year institutions
    b. Certificates, associates, bachelor, and advanced degree opportunities

2. Computing, Coding, and Information Technology Applications

Objective: *Students will explain how computer programming supports networking systems, secure communication, automation, monitoring, and modern digital infrastructure.*

5) Identify major milestones in the development of computers, programming, and information technology.
6) Describe how software is used in networking systems for automation, monitoring, communication, device management, and security.
7) Explain the relationship between hardware, software, networks, protocols, and user input in digital systems.
8) Compare the roles of programming in personal, social, economic, and information technology contexts.
9) Analyze how computational innovations have changed information technology, business, health care, communication, and other industries.
10) Predict how coding, automation, network management, and security tools may continue to transform digital infrastructure in the future.

3. Ethical, Safe, and Responsible Programming

Objective: *Students will evaluate ethical, legal, privacy, and safety considerations related to programming, data use, networks, and secure technology systems.*

11) Describe ethical issues related to privacy, security, copyright, piracy, fraud, sabotage, and misuse of software.
12) Explain why responsible programming practices are essential in the design, operation, and protection of networked systems.
13) Analyze the risks of unreliable code in systems that collect data, make automated decisions, or control devices.
14) Evaluate programming decisions using ethical, legal, and safety-based criteria.
15) Defend responsible choices related to open-source software, intellectual property, and data protection.
16) Apply safe computing and digital citizenship practices while developing, testing, and sharing code used in networked environments.

4. Computational Thinking and Program Design

Objective: *Students will design solutions to programming problems by applying computational thinking and pre-coding planning strategies to networking and security scenarios.*

17) Define problems by identifying inputs, outputs, constraints, and desired results.
18) Break down multistep problems into smaller, manageable components.
19) Develop algorithms that solve programming and information technology-related challenges.
20) Construct pseudocode and flowcharts before writing code for networking, monitoring, or security-related tasks.
21) Organize program logic using sequence, selection, and iteration.
23) Design logic for a networking or security scenario such as validating user credentials, monitoring device status, processing logs, or analyzing traffic data.
23) Design logic for an information technology scenario such as processing user requests, monitoring system activity, or analyzing input data.

5. Programming Fundamentals and Code Construction

Objective: *Students will write syntactically correct programs that use variables, data types, operators, conditionals, and loops to solve introductory networking and security problems.*

24) Identify and use appropriate data types such as integers, floats, strings, booleans, and lists/arrays.
25) Create meaningful variable names and constants following accepted coding conventions.
26) Write programs that gather, store, process, and display information.
27) Apply arithmetic, relational, and logical operators to solve networking and security-related computational problems.
28) Develop code using conditional statements to support decision-making.
29) Develop code using loops to automate repetitive tasks.
30) Use functions or modular code to improve readability and reusability.
31) Convert planned algorithms into working program code in a teacher-selected programming language.
32) Create simple networking or security-themed applications such as password checkers, login attempt trackers, IP calculators, port check simulations, or system status tools.

6. Data, Files, and Information Management

Objective: *Students will manage data effectively within programs and use file structures appropriate for networking, security, and real-world technical tasks.*

33) Explain how programs receive, store, manipulate, and output data.
34) Organize program data using variables, lists, arrays, and basic file structures.
35) Read data from user input and files and write data to files or reports.
36) Apply file naming, organization, and management practices across software, networking, and security environments.
37) Distinguish between temporary program data and stored data.
38) Analyze how networking and security systems use data from user input, device activity, network traffic, transactions, and stored values.
39) Develop programs that process network logs, user records, device data, or system alerts.
40) Interpret data outputs to support decisions about system performance, accuracy, or design success.

7. Testing, Debugging, and Quality Assurance

Objective: *Students will test, debug, and improve code using structured quality assurance practices relevant to networking and security tools.*

41) Test programs using valid, invalid, and edge-case inputs.
42) Locate syntax, logic, and runtime errors in code.
43) Debug code systematically using error messages, tracing, and test cases.
44) Evaluate whether a program meets its intended purpose and user requirements.
45) Revise code to improve correctness, efficiency, readability, and reliability.
46) Document testing procedures, bugs, and revisions.
47) Apply quality assurance practices such as version notes, test evidence, and performance checks.
48) Assess how faulty code could affect security, privacy, reliability, efficiency, or performance in a networked system.

8. Number Systems, Logic, and Computer Operations

Objective: *Students will explain how computers and networks represent and process information using binary, hexadecimal, logical structures, and digital communication concepts.*

49) Convert numbers between base-10, binary, and hexadecimal.
50) Explain why binary representation is essential to digital systems.
51) Interpret how data, instructions, and signals are processed by computing devices.
52) Relate number systems to storage, memory, and communication in computing systems.
53) Analyze how digital logic supports processors, storage, networking, and automated responses in computing systems.
54) Apply number system knowledge to practical networking, addressing, and digital communication examples.

9. Applied Coding Project

Objective: *Students will create and present a beginning-level programming project that solves a networking, systems support, or security-related problem.*

55) Plan a project using a defined problem statement, criteria, and constraints.
56) Design a program that addresses a networking, systems support, or security challenge.
57) Develop code that uses input, processing, output, decision-making, and repetition to support a networking or security-related function.
58) Test and refine the program based on evidence from trial runs.
59) Document the design process, code structure, and testing outcomes.
60) Present the final product and justify design decisions using technical vocabulary.
61) Evaluate the effectiveness of the project and propose logical next steps for improvement.

10. Leadership

Objective: *Students will demonstrate leadership, teamwork, and professional skills through participation in classroom, CTSO, and career-connected activities.*

62) Demonstrate effective communication, collaboration, and problem-solving skills in individual and team settings.
63) Apply leadership strategies such as goal setting, decision-making, conflict resolution, and responsible follow-through.
64) Participate in CTSO or leadership development experiences that strengthen professionalism, civic responsibility, and career readiness.

11. Work-Based Learning

Objective: *Students will connect classroom learning to career pathways through work-based learning experiences, industry engagement, and reflection on workplace expectations.*

65) Identify the purpose and value of work-based learning experiences such as guest speakers, job shadows, industry-sponsored projects, internships, apprenticeships, or supervised workplace experiences.
66) Demonstrate workplace readiness skills including attendance, time management, communication, professionalism, and safe work practices.
67) Reflect on work-based learning experiences to evaluate personal strengths, growth areas, and next steps for career and postsecondary planning.

12. Career and Technical Student Organizations

Objective: *Using the CTSO Resource Document, chapter activities will provide some of the best opportunities CTSO members will have to learn by doing. A successful program of work creates a positive learning atmosphere in the classroom. CTSO members learn how to accept responsibility, work as a team, manage a budget, and handle success and failure. Per the CTE Admin Handbook, CTSOs are embedded into curriculum.*

Professional Development

Prepare each member for entry into the workforce and provide a foundation for success in a career. Becoming a professional does not stop with acquiring a skill, but involves an increased

awareness of the meaning of good citizenship and the importance of labor and management in the world of work.

Community Service

Promote and improve goodwill and understanding among all segments of the community through services donated by CTSO chapters, and to instill in its members a lifetime commitment to community service.

Business and Industry Connections

Increase awareness of quality job practices and attitudes, and increase the opportunities for employer engagement and eventual employment.

Financial Leadership

Plan and participate in activities to allow all members to carry out the chapter's projects. Educate members on their own personal finance for current and future success.

Public Relations

Make the general public aware of the good work that students in career and technical education are doing to better themselves and their community, state, nation, and world.

Social Activities

To increase cooperation in the school and community through activities that allow CTSO members to get to know each other in something other than a business or classroom setting.

Academic Standards Alignment

Colorado CTE courses are required to integrate academic and essential skills into course content. The following academic standards have been identified as aligned to this course and should be integrated throughout instruction.

CO CTE Academic Standards Resources (Rubrics, integration strategies, etc)
CDE Colorado Academic Standards Online (Online standards lookup tool)

Note: Mathematics and Science standards will be added in the future. Visit the Colorado Academic Standards page for the most current version.

Reading, Writing and Communicating Crosswalk

Course: Coding 1
Colorado Academic Standards – Reading, Writing, and Communicating

Standard	Prepared Graduate Competency	Description	Course Evidence / Alignment
Standard 1: Oral Expression and Listening	Prepared Graduate 1 – Collaboration	Collaborate effectively as group members or leaders.	Students collaborate on coding challenges, discuss logic and errors, and provide and receive peer feedback.
Standard 1: Oral Expression and Listening	Prepared Graduate 2 – Oral Presentations	Deliver effective oral presentations for varied audiences.	Students explain code functionality, present projects or applications, and verbally walk through algorithms.
Standard 2: Reading for All Purposes	Prepared Graduate 4 – Informational Texts	Read a wide range of informational texts.	Students read programming documentation, tutorials, examples, problem statements, and specifications.
Standard 2: Reading for All Purposes	Prepared Graduate 5 – Language in Context	Understand how language functions in different contexts.	Students learn and apply domain-specific language such as syntax, commands, and logic, and interpret meaning across programming contexts.

Standard 3: Writing and Composition	Prepared Graduate 7 – Informational Writing	Craft informational/explanatory texts.	Students write code comments, explain algorithms and solutions, and document program behavior.
Standard 3: Writing and Composition	Prepared Graduate 9 – Writing Process	Demonstrate mastery of their own writing process.	The course reinforces drafting, testing, debugging, and refining, resulting in clean, correct, and readable code and explanations.
Standard 4: Research Inquiry and Design	Prepared Graduate 10 – Research and Ethics	Gather information from a variety of sources.	Students research solutions, syntax, and structures, evaluate sources for accuracy and relevance, and apply information ethically.

Free Resource Links

1. Career Exploration and Postsecondary Opportunities

- Bureau of Labor Statistics Computer and Information Technology Occupations (IT careers job outlook education pathways)
  https://www.bls.gov/ooh/
- CompTIA Career Paths (IT roles certifications technical support networking cybersecurity)
  https://www.comptia.org/content/it-careers-path-roadmap
- CareerOneStop Information Technology Careers (career exploration certifications education planning)
  https://www.careeronestop.org
- Code.org Career Connections (computer science careers workplace skills future jobs)
  https://code.org/careers

2. Computing, Coding, and Information Technology Applications

- Code.org Computer Science (coding fundamentals, programming concepts, real world applications)
  https://code.org
- Code.org Student Learning (coding fundamentals, programming concepts, real world applications)
  https://code.org/en-US/students
- Arduino Resources for Beginner Coders (algorithms, flow control, data types, beginner coding practice)
  https://www.arduino.cc/education/resources-for-beginner-coders/
- Scratch MIT Programming (intro coding, logic, simulations, beginner projects)
  https://scratch.mit.edu

3. Ethical, Safe, and Responsible Programming

- Common Sense Digital Citizenship (ethics, online safety, responsible technology use)
  https://www.commonsense.org/education
- CodeHS Cybersecurity and Ethics (ethical programming, data privacy, security awareness)
  https://codehs.com
- Stanford Copyright and Fair Use (copyright law, intellectual property, fair use basics)
  https://fairuse.stanford.edu
- Google Be Internet Awesome (internet safety, responsible online behavior)
  https://beinternetawesome.withgoogle.com

4. Computational Thinking and Program Design

- Google Computational Thinking (problem solving, decomposition, abstraction, algorithms)
  https://edu.google.com
- Lucidchart Flowchart Guide (flowcharts, diagramming, algorithm planning)
  https://www.lucidchart.com/pages/flowchart

- Khan Academy Algorithms (algorithms, logic, problem solving basics) https://www.khanacademy.org/computing
- CS Unplugged Activities (computational thinking, offline activities, problem solving) https://csunplugged.org

5. Programming Fundamentals and Code Construction

- W3Schools Python Tutorial (syntax, variables, loops, conditionals, basics) https://www.w3schools.com/python
- Replit Coding Environment (online coding, projects, collaboration, debugging) https://replit.com
- CodeHS Programming Course (coding exercises, practice, structured learning) https://codehs.com
- FreeCodeCamp Programming (projects, coding tutorials, debugging skills) https://www.freecodecamp.org

6. Data, Files, and Information Management

- Khan Academy Data Basics (data storage, variables, arrays, organization) https://www.khanacademy.org
- GeeksforGeeks File Handling (file input, output, data processing, programming) https://www.geeksforgeeks.org/file-handling-python
- Google Sheets Training (data organization, spreadsheets, analysis skills) https://support.google.com/docs
- Code.org Data Systems (data storage, management, computing systems) https://studio.code.org

7. Testing, Debugging, and Quality Assurance

- MIT Scratch Debugging (debugging strategies, troubleshooting, logic errors) https://scratch.mit.edu
- Guru99 Software Testing (testing concepts, debugging, quality assurance basics) https://www.guru99.com/software-testing.html
- FreeCodeCamp Debugging Guide (debugging techniques, troubleshooting, coding errors) https://www.freecodecamp.org/news
- Khan Academy Testing Concepts (testing, validation, logic correctness, verification) https://www.khanacademy.org

8. Number Systems, Logic, and Computer Operations

- Khan Academy Binary and Hexadecimal (number systems, conversions, computing basics) https://www.khanacademy.org
- Cisco Binary Game (binary practice, interactive learning, computing logic) https://www.netacad.com
- All About Circuits Logic Gates (logic gates, computing circuits, AND OR NOT) https://www.allaboutcircuits.com
- Crash Course Computer Science (computer systems, data processing, basics) https://www.youtube.com

9. Applied Coding Project

- Code.org Project Builder (project planning, coding workflows, development) https://code.org
- MIT App Inventor Projects (app development, user interface design, problem solving activities) https://appinventor.mit.edu/explore/resources
- Code.org Project Builder (project planning, coding workflows, development) https://code.org
- Replit Projects (software projects, coding practice, application development) https://replit.com

10. Leadership

- Texas A&M Center for Teaching Excellence Group Work Guide (teamwork, collaboration, communication, professionalism) https://cte.tamu.edu/resources/practical-guide-effective-group-work.html
- Cornell Teaching Innovation Team Projects Guide (team roles, project collaboration, accountability, conflict resolution) https://teaching.cornell.edu/teaching-resources/active-collaborative-learning/collaborative-learning/preparing-students-work
- Duke Center for Teaching and Learning Teams Resource (leadership, teamwork, shared responsibility, project management) https://ctl.duke.edu/resources/art-and-science-of-teaching/use-student-teams-effectively/
- University of Minnesota Teamwork and Leadership (leadership, collaboration, conflict management, career readiness) https://cla.umn.edu/undergraduate-students/cla-core-competencies/career-readiness-cla/teamwork-and-leadership

11. Work-Based Learning

- Colorado Workforce Development Council Work-Based Learning (Colorado continuum, industry engagement, career pathways, workforce readiness) https://cwdc.colorado.gov/strategies/work-based-learning
- U.S. Department of Labor Career Preparation and Work-Based Learning (career readiness, workplace skills, job exploration, planning) https://www.dol.gov/agencies/odep/program-areas/individuals/youth/career-preparation
- Colorado State Agency Work-Based Learning Definitions (job shadowing, internships, career exploration, project-based learning) https://dhr.colorado.gov/work-based-learning-skills-based-hiring/state-agency-work-based-learning-definitions
- IES Work-Based Learning Checklist (student experiences, employer partnerships, reflection, quality indicators) https://ies.ed.gov/rel-northwest/2025/12/work-based-learning-checklist

Technology Student Association – National Competitive Events

Coding

Objective: *Students will demonstrate coding knowledge and develop a software solution to an on-site programming challenge aligned with TSA Coding competition requirements.*

- Study coding concepts such as syntax, data structures, control flow, and object-oriented programming.
- Prepare for an individual coding-concepts test used to determine semifinalist placement.
- Analyze an on-site coding problem, evaluation criteria, required tools, and programming constraints.
- Design and develop a software solution within the allotted competition time.
- Use an approved programming language, operating system, or API specified for the challenge.
- Test and debug the program to improve accuracy, functionality, and efficiency.
- Present the completed solution for objective testing and judge evaluation.

Resources

- Technology Student Association – Coding competition guidelines
  https://tsaweb.org/competitions-programs/tsa/themes-problems
- Code.org – Computer science and programming lessons
  https://code.org/
- freeCodeCamp – Programming tutorials and practice
  https://www.freecodecamp.org/
- W3Schools – Programming references and examples
  https://www.w3schools.com/

Technology Student Association – National Competitive Events

Software Development

Objective: *Students will design, develop, document, test, and present a software application that responds to the annual TSA Software Development challenge.*

- Analyze the annual software development challenge and identify the intended users, purpose, and functional requirements.
- Plan the software application using appropriate design tools such as pseudocode, flowcharts, diagrams, or wireframes.
- Develop a functioning software product using programming practices appropriate to the selected platform.
- Create documentation that explains the software purpose, development process, features, testing, and revisions.
- Apply quality assurance practices through testing, debugging, and refinement.
- Prepare a presentation or demonstration that explains the software solution and development decisions.
- Evaluate the final software using TSA criteria for functionality, design quality, documentation, usability, and presentation.

Resources

- Technology Student Association – Software Development competition guidelines
  https://tsaweb.org/competitions-programs/tsa/themes-problems
- GitHub Docs – Version control and project collaboration
  https://docs.github.com/
- Replit – Browser-based coding environment
  https://replit.com/
- freeCodeCamp – Software development tutorials
  https://www.freecodecamp.org/

Technology Student Association – National Competitive Events

Webmaster

Objective: *Students will design, build, launch, and explain a multi-page website that addresses the annual TSA Webmaster design challenge.*

- Research the annual Webmaster topic and identify the website's purpose, audience, content needs, and design requirements.
- Plan the website structure using a sitemap, wireframes, page layouts, or other web design planning tools.
- Develop a multi-page website that communicates the team's solution to the annual theme and problem.
- Apply web design practices related to navigation, accessibility, usability, visual consistency, and responsive layout.
- Launch the website on a publicly accessible web server according to TSA submission requirements.
- Test the website for functionality, links, layout, readability, and access issues before submission.
- Participate in an interview explaining the website, design choices, research, and development process.

Resources

- Technology Student Association – Webmaster competition guidelines https://tsaweb.org/competitions-programs/tsa/themes-problems
- Mozilla Developer Network – HTML, CSS, and JavaScript documentation https://developer.mozilla.org/
- W3Schools – Web development tutorials https://www.w3schools.com/
- GitHub Pages – Website hosting for student projects https://pages.github.com/