Open Education and Development Group, Python Institute.
Alignment to the CTE-IT.68.9009200

Python Institute Course

Course Title	PyCrafters™ – Python for Teens
Course Year	2025
Grade Level	6th - 7th - 8th - 9th
Standards Link	https://ctepreview.cpalms.org/programs/standard-benchmark?programcip=0511020109&version=2024

Note. Coverage(s) for a "Lesson" refer to the lesson content and its associated activities and exercises, unless otherwise specified. The "Supplemental Resources" (such as demonstration videos or guided examples) are intended to introduce or reinforce the core concepts, while the main lesson text provides the complete details.

Course Description

The purpose of this course is to introduce middle school students to the fundamental concepts of computer programming and computational thinking using the Python programming language. This course aims to help learners establish a strong foundation in problem-solving, logical reasoning, and structured coding practices, while also fostering creativity through interactive and visually engaging projects.

Students will gain hands-on experience in writing, testing, and debugging code. They will learn to use variables, conditionals, loops, functions, and data structures, applying these skills to create engaging projects such as games, animations, and simple data applications.

In addition to developing technical skills, the course emphasizes essential workplace competencies, including teamwork, project planning, documentation, and both oral and written communication. These skills align with industry expectations as outlined in the CTE Coding Fundamentals standards. Students will also be introduced to the role of operating systems in supporting programming tasks and will explore how coding can be applied in business and everyday scenarios.

By the end of the course, students will have developed not only the ability to write and analyze Python programs but also the confidence to use coding as a tool for problem-solving in various domains. This course prepares students for further study in computer science, software development, or related career paths within the Information Technology field.

This course has been carefully designed to align with the **Coding Fundamentals CTE Standards (CTE-IT.68.9009200)**. Each lesson integrates the required knowledge and skills through age-appropriate coding activities, problem-solving exercises, and collaborative projects. The standards are addressed not only through direct programming instruction but also through the use of common software applications, operating systems, teamwork, project management tools, and communication tasks. This ensures that students build both technical proficiency in Python and essential workplace competencies.

The specific alignment of each standard and objective to lessons in the *PyCrafters™ – Python for Teens* course is shown on pages 3–8 of this document.

CTE Alignment Table – PyCrafters Syllabus

The table below shows how each Coding Fundamentals CTE Standard (**CTE-IT.68.9009200**) and its individual objectives are aligned with the corresponding lessons in the *PyCrafters™ – Python for Teens* course, demonstrating where and how students gain the required knowledge, practice essential skills, and apply concepts through lesson content, activities, and projects.

CTE-IT.68.9009200.1 – Demonstrate proficiency using specialized coding software. The student will be able to:	**Citation(s)**
1.1 Use specialized computer coding software to solve problems.	**Lesson 6**: First Steps in Python – running programs in IDEs. **Lesson 7**: Numbers and Text in Python – writing small problem-solving programs. **Lesson 8**: Making Choices with If Statements – applying logic to solve problems. **Lesson 9**: Repetition with Loops – solving repetitive tasks. **Lesson 10**: Organizing Code with Functions – structuring code for problem-solving. **Lessons 16, 23, 24**: Mini Projects and Final Project – solving larger, real-world style problems.
1.2 Demonstrate proficiency using specialized computer software (e.g., Swift, Python).	**Lesson 6–15**: Python Fundamentals (Hello World → Debugging Basics). **Lessons 17–22**: Visual & Interactive Programming with Turtle, Pygame, VPython. **Lesson 23**: Mini Project – Visual Creation. **Lessons 24–25**: Final Project & Presentations.

CTE-IT.68.9009200.2 – Develop an awareness of programming languages. The student will be able to:	**Citation(s)**
2.1 Identify programming language design approaches.	**Lesson 4**: Programming Languages & Careers – comparing Python, JavaScript, Swift, etc.
2.2 Explain the components of programming languages.	**Lesson 4**: Programming Languages & Careers – variables, syntax, structure. **Lesson 6**: First Steps in Python – components of a simple program.

CTE-IT.68.9009200.3 – Demonstrate proficiency using common software applications. The student will be able to:	**Citation(s)**
3.1 Compare/contrast the appropriate use of various software applications.	**Lesson 1**: What Is a Computer? **Lesson 2**: Exploring Software Applications & Data **Lesson 4**: Programming Languages & Careers **Lesson 5**: Mini Project – Computing Foundations
3.2 Demonstrate proficiency in the use of software applications.	**Lesson 2**: Exploring Software Applications & Data – spreadsheets, filters, charts. **Lesson 5**: Mini Project – Computing Foundations.
3.3 Explain why different file types exist.	**Lesson 1**: What Is a Computer? – file extensions. **Lesson 2**: Exploring Software Applications & Data – .xlsx vs .csv. **Lesson 4**: Programming Languages & Careers – workplace file types. **Lesson 14**: Saving and Loading Data – text vs CSV.

3.4 Identify the kinds of content associated with different file types.	**Lesson 1**: What Is a Computer? **Lesson 2**: Exploring Software Applications & Data **Lesson 14**: Saving and Loading Data

CTE-IT.68.9009200.4 – Demonstrate knowledge, skill, and application of information systems to accomplish job objectives. The student will be able to:	**Citation(s)**
4.1 Develop keyboarding skills to enter and manipulate text/data.	**Lesson 1**: What Is a Computer? **Lesson 2**: Exploring Software Applications & Data **Lesson 3**: Algorithms & Problem-Solving **Lesson 6+**: Python coding practice throughout
4.2 Use current/emerging technology to perform personal and business tasks.	**Lesson 2**: Exploring Software Applications & Data – spreadsheets, charts. **Lesson 7**: Numbers and Text in Python – calculators, utilities. **Lesson 16**: Mini Project – Python Fundamentals – diary, expense tracker.
4.3 Perform operations such as sorting, filtering, searching, and displaying info in charts/tables/graphs.	**Lesson 2**: Exploring Software Applications & Data **Lesson 12**: Working with Lists – sorting/searching lists **Lesson 13**: Dictionaries – storing/retrieving data **Lesson 14**: Saving and Loading Data – reading structured text

CTE-IT.68.9009200.5 – Demonstrate comprehension and communication.	**Citation(s)**

The student will be able to:	
5.1 Use listening, speaking, and nonverbal strategies to communicate.	**Lessons 3–5**: Group exercises, team research, presentations **Lesson 16**: Mini Project – Python Fundamentals **Lessons 23–25**: Mini & Final Projects
5.2 Organize and communicate oral/written messages.	**Lesson 3**: Algorithms & Problem-Solving – pseudocode, flowcharts **Lesson 4**: Programming Languages & Careers – team presentations **Lessons 5, 16, 23, 24**: Mini and Final Projects – reports, README files
5.3 Collaborate with teams.	**Lesson 3**: Group pseudocode/flowchart activities **Lesson 5**: Mini Project – Computing Foundations **Lesson 16**: Mini Project – Python Fundamentals **Lesson 24**: Final Project Development
5.4 Demonstrate awareness of project management tools.	**Lesson 5**: Mini Project – Computing Foundations (simple checklists, task trackers) **Lesson 16**: Mini Project – Python Fundamentals (sticky notes/Kanban) **Lesson 24**: Final Project Development (Kanban boards, Trello, shared docs)
5.5 Communicate appropriately online.	**Lesson 25**: Exam Prep & Presentations – sharing code and feedback online.
5.6 Recognize that multiple algorithms solve a problem.	**Lesson 3**: Algorithms & Problem-Solving **Lesson 5**: Mini Project – Computing Foundations **Lessons 7–9**: Conditionals, loops (different solutions to same problem) **Lesson 24**: Final Project Development
5.7 Create a program that implements an algorithm collaboratively.	**Lesson 16**: Mini Project – Python Fundamentals **Lesson 23**: Mini Project – Visual Creation **Lesson 24**: Final Project Development

CTE-IT.68.9009200.6 – Demonstrate knowledge of different operating systems. The student will be able to:	**Citation(s)**
6.1 Compare/contrast OSs (Windows, UNIX, Android, iOS).	**Lesson 1**: What Is a Computer?
6.2 Demonstrate proficiency using OS tools (taskbar, shortcuts, CLI).	**Lesson 1**: What Is a Computer? – file/folder management **Lesson 5**: Mini Project – Computing Foundations – CLI/GUI practice
6.3 Use iterative development/debugging in the problem domain.	**Lesson 3**: Algorithms & Problem-Solving – problem decomposition **Lessons 7–9**: Debugging logic in conditionals/loops **Lesson 15**: Debugging Basics

CTE-IT.68.9009200.7 – Demonstrate proficiency in basic programming. The student will be able to:	**Citation(s)**
7.1 Describe the structure of a simple program.	**Lesson 6**: First Steps in Python
7.2 Define “algorithm” and its relation to problem-solving.	**Lesson 3**: Algorithms & Problem-Solving **Lesson 5**: Mini Project – Computing Foundations
7.3 Describe iterative structures.	**Lesson 9**: Repetition with Loops
7.4 Describe selection structures.	**Lesson 8**: Making Choices with If Statements

7.5 Explain variables.	**Lesson 7**: Numbers and Text in Python
7.6 Write pseudocode with structured programming.	**Lesson 3**: Algorithms & Problem-Solving **Lesson 5**: Mini Project – Computing Foundations **Lesson 16**: Mini Project – Python Fundamentals
7.7 Troubleshoot/debug errors.	**Lesson 15**: Debugging Basics **Lesson 16**: Mini Project – Python Fundamentals **Lesson 25**: Exam Prep & Presentations
7.8 Create/modify/use a database (dictionaries/file I/O).	**Lesson 12**: Working with Lists **Lesson 13**: Storing Information with Dictionaries **Lesson 14**: Saving and Loading Data **Lesson 16**: Mini Project – Python Fundamentals