Tennessee Coding I (Python) Syllabus

High School - course length (150 hours)

Course Overview and Goals

The CodeHS Tennessee Coding I in Python curriculum introduces high school students to the fundamentals of computer programming using Python. Students begin with digital citizenship and cybersecurity before exploring core programming concepts including variables, conditionals, loops, functions, and data structures. The course also covers graphics programming with Python Turtle, file input/output, object-oriented programming with classes, and the software development life cycle. Upon completion, students will be able to plan and implement multistep Python programs, apply logical problem-solving strategies, and demonstrate responsible and ethical use of technology.

Learning Environment

The course utilizes a blended classroom approach. The content is fully web-based, with students writing and running code in the browser. Teachers utilize tools and resources provided by CodeHS to leverage time in the classroom and give focused 1-on-1 attention to students. Each unit of the course is broken down into lessons. Lessons consist of video tutorials, short quizzes, example programs to explore, and written programming exercises, adding up to over 100 hours of hands-on programming practice in total. Each unit ends with a comprehensive unit test that assesses students' mastery of the material from that unit as well as challenge problems where students can display their understanding of the material.

Development Environment

Students write and run Python programs in the browser using the CodeHS editor.

Prerequisites

Tennessee Coding I in Python is designed for high school students with no prior programming experience. Students should be comfortable using a computer and navigating the web, but no prior coding knowledge is required.

Technology Requirements

To complete all activities and exercises in this course, students must have access to the 3rd party sites and tools listed here: https://codehs.com/lms/course_whitelist/28659

More Information

Browse the content of this course at https://codehs.com/course/28659/explore

Course Breakdown

Module 1: Cybersecurity and You (2 weeks / 10 hours)

In this module, students explore their responsibilities as digital citizens. They examine the concept of a digital footprint, how personal data is collected and used, how to evaluate online information, and how to recognize and prevent common cyber threats including phishing and cyberbullying.

Topics Covered	• Digital Footprint and Responsibility • Personal Data Collection and Use • Evaluating Online Information • Cyber Ethics and Laws • Personal Data Security • Cybersecurity Essentials • Common Cyber Attacks and Prevention

Example Assignments	• **Reflection: Teens, Social Media, and Technology**
	  ○ Students analyze data visualizations from a Pew Research report on teen social media use and respond to structured questions about what they notice and wonder.
	• **What Is Your Digital Footprint?**
	  ○ Students reflect on why their digital footprint matters and answer questions about the impact of their online activity.
	• **Cyberbullying Case Study**
	  ○ Students read a cyberbullying scenario and apply ethical frameworks to analyze the situation and propose a responsible response.
	• **Your Personal Data and Who Is Using It**
	  ○ Students explore how technology is integrated into daily life and consider the trade-offs of sharing personal data with companies.

Module 2: Basic Python and Console Interaction (2 weeks / 10 hours)

In this module, students write their first Python programs. They learn to print output, declare variables of different types, read user input, perform mathematical and string operations, and document code with comments.

Topics Covered	• Printing in Python • Variables and Data Types • User Input • Mathematical Operators • String Operators • Comments and Collaborative Programming

Example Assignments	• **Introduce Yourself**
	  ○ Students write a program that prints their name and something they enjoy, practicing print statements and string literals.
	• **Make Some Variables!**
	  ○ Students create and initialize both string and integer variables, choosing meaningful names and printing their values.
	• **Hello <name>**
	  ○ Students write a program that reads a user's name as input and prints a personalized greeting, storing the input in a variable.
	• **Vertical Name**
	  ○ Students display their name vertically, printing one character per line to practice basic output formatting.

Module 3: Intro to Python with Turtle Graphics (3 weeks / 15 hours)

In this module, students use Python's Turtle graphics library to draw shapes and scenes on a canvas. They apply loops, functions, variables, parameters, string methods, and control structures in a visual context, building toward fully interactive programs.

Topics Covered	• Meet Tracy the Turtle • Tracy's Grid World and Positioning • Turning Tracy and Using Angles • For Loops in Graphics • Functions and Parameters • Artistic Effects and Adding Text • Variables and Strings in Graphics • User Input and Clickable Interaction • If/Else Statements and While Loops in Graphics • Putting Together Control Structures

Example Assignments	• **Stretched Slinky**
	  ○ Students program Tracy to draw a slinky with five rings, applying loops and circle-drawing commands.
	• **Caterpillar**
	  ○ Students use Tracy commands to draw a multi-segment caterpillar, practicing shape composition and positioning.
	• **Shorter Dashed Line**
	  ○ Students draw a dashed line across the x-axis, using loops to control pen-up and pen-down alternation.
	• **Coordinates Practice**
	  ○ Students read a grid image, determine coordinates of a target point, and move Tracy to the correct location.

Module 4: System Administration (3 weeks / 15 hours)

In this module, students explore how computers work at a systems level. They compare operating systems, learn about software licenses and application security, configure browser settings, use the command line interface, and research the history of computing.

Topics Covered	• Operating Systems and Types • Comparing Operating Systems • Compatibility and File Extensions • Laptops vs. Tablets • Software and Applications • Software Licenses • Application Security • Browser Configuration • Command Line Interface • Programming Languages and Computing History
Example Assignments	• **Lab: Configuring a Computer** ○ Students personalize their computer settings by completing a guided lab that walks through key system configuration options.

	• **Lab: Navigating with the CLI**
	  ○ Students use command line commands to navigate a file system, practicing essential terminal skills.
	• **Which OS Is Right for You?**
	  ○ Students take a quiz to identify a matching Linux distribution and create a comparison between operating systems to justify their choice.
	• **Operating Systems Step Inside**
	  ○ Students use the Step Inside thinking routine to explore the perspective and role of an operating system in a computer.

Module 5: Conditionals and Looping (3 weeks / 15 hours)

In this module, students add decision-making and repetition to their Python programs. They write boolean expressions, if/else statements, logical and comparison operators, while loops, for loops, break and continue statements, and nested control structures.

Topics Covered	• Booleans and Boolean Expressions • If Statements and If/Else Statements • Comparison Operators • Logical Operators • Floating Point Numbers and Rounding • While Loops • For Loops • Break and Continue • Nested Control Structures

Example Assignments	• **Old Enough to Vote?**
	  ○ Students write a program that reads a user's age and reports whether they are eligible to vote, applying comparison operators and if/else logic.
	• **Positive, Zero, or Negative?**
	  ○ Students prompt the user for a number and classify it as positive, zero, or negative using conditional statements.
	• **Divisibility**
	  ○ Students use a while loop to validate user input and prevent division by zero before computing a result.
	• **Average Test Score**
	  ○ Students ask the user for three test scores and compute and report the average using arithmetic expressions.

Module 6: Functions and Exceptions (1 week / 5 hours)

In this module, students define and call their own functions. They explore parameters, return values, variable scope and namespaces, default parameter values, and how to handle runtime errors gracefully using exception handling.

Topics Covered	• Defining and Calling Functions • Parameters and Arguments • Return Values

	• Namespaces and Variable Scope • Default Parameter Values • Exceptions and Error Handling

Example Assignments	• **Weather**
	  ○ Students write a program that asks for the weather and uses a function to suggest appropriate footwear based on the conditions.
	• **Print Product**
	  ○ Students define a function that accepts two integer arguments and prints their product, practicing parameter passing.
	• **Print Multiple Times**
	  ○ Students write a function that takes a string and an integer and prints the string the specified number of times.
	• **Area of a Square with Default Parameters**
	  ○ Students build a function with a default side length parameter that computes and prints the area of a square.

Module 7: Data Security (2 weeks / 10 hours)

In this module, students explore how data is stored, protected, and potentially exploited. They learn about databases, SQL injection, cross-site scripting, environmental controls, vulnerability scanning, and risk management strategies.

Topics Covered	• Data as a Resource • Using Databases and Database Management Systems • Security in Coding: SQL Injection • Dev Tools Capture the Flag • Environmental Controls • Checking for Vulnerabilities • Risk Management

Example Assignments	• **Benefits of Data Response**
	  ○ Students analyze a video about how companies use user data and respond to questions about the benefits and risks of data collection.
	• **Field Trip: Data Center**
	  ○ Students virtually explore a Microsoft Data Center and answer reflection questions about physical data infrastructure.
	• **States of Data Creation**
	  ○ Students create a visual representation of the different states that data passes through during its lifecycle.
	• **Dev Tools Capture the Flag**
	  ○ Students use browser developer tools to find hidden flags in web page source code, learning how publicly visible code can be a security vulnerability.

Module 8: Project: Engineering Design Process (2 weeks / 10 hours)

In this module, students apply the engineering design process to a real-world problem. They work through empathy, definition, ideation, prototyping, and testing phases, then present their design to the class.

Topics Covered	• Engineering Design Process Overview • Research and Empathy • Define and Point-of-View Statements • Ideation Techniques • Prototyping • User Testing • Showcasing and Presenting

Example Assignments	• **Engineering Design Notebook**
	  ○ Students document each phase of their design process in an engineering notebook, tracking decisions and iterations throughout the project.
	• **Accessibility: Designing for ALL**
	  ○ Students reflect on the meaning of accessibility and the importance of designing products that can be used by everyone.
	• **User Interview**
	  ○ Students interview a classmate or community member using structured empathy techniques to gather data for their design.
	• **Showcase Your Creation!**
	  ○ Students present their finalized prototype and explain how user testing shaped their design decisions.

Module 9: Data Structures (2 weeks / 10 hours)

In this module, students store and manipulate collections of data using Python's built-in data structures. They work with tuples, lists, list methods, 2D lists, list comprehensions, packing and unpacking, and dictionaries.

Topics Covered	• Tuples • Lists and List Methods • For Loops and Lists • 2D Lists • List Comprehensions • Packing and Unpacking • Dictionaries

Example Assignments	• **Word Ladder**
	  ○ Students build a word ladder — a list of words where each word differs by one letter — applying list creation and manipulation techniques.
	• **How Many Names?**
	  ○ Students write a program that handles names of varying length (first, middle, last) by storing them in a list and iterating over it.
	• **Checkerboard**
	  ○ Students build a program across three parts that creates a 2D list and populates it with an alternating 0/1 checkerboard pattern.
	• **Tic Tac Toe**
	  ○ Students complete a program that simulates a game of tic tac toe, using a 2D list to track board state and implement game logic.

Module 10: File I/O (1 week / 5 hours)

In this module, students learn to read from and write to files in Python. They work with file pointers, read characters, lines, and full file contents, write output to files, and apply file I/O to process real-world text data.

Topics Covered	• What Is File I/O? • Reading Characters and Lines from Files • Reading All Lines from a File • Writing to Files • Moving the File Pointer

Example Assignments	• **Validating Tweet Length**
	  ○ Students write a function that reads a text file and determines whether its contents are within Twitter's character limit.
	• **Counting Lines in a File**
	  ○ Students write a program that opens a story file and counts the number of lines it contains.
	• **Summing Numbers from File**
	  ○ Students read a file of numbers (one per line) and compute their sum using file iteration.
	• **Formatting Movie Titles**
	  ○ Students read a list of movie titles from a file, apply string formatting, and write the cleaned results to an output file.

Module 11: Software Development and Careers (1 week / 5 hours)

In this module, students explore the professional side of computing. They study the software development life cycle, plan and prototype a small app, conduct user testing, and learn about roles on a software development team and career pathways in computer science.

Topics Covered	• Software Development Life Cycle • Planning, Prototyping, and Rapid Iteration • User Testing • Software Deployment and Licensing • Roles on a Software Development Team • Computer Science Careers

Example Assignments	• **Create a Paper Prototype**
	  ○ Students sketch a rapid prototype of their app idea on paper, applying design thinking principles before writing any code.
	• **Program Your App — Version 1**
	  ○ Students implement the first version of their app from pseudocode, meeting a defined set of functional requirements.
	• **Program Your App — Final!**
	  ○ Students incorporate feedback from user testing to improve and finalize their app, then deploy and document it.

Module 12: Karel in Python (3 weeks / 15 hours)

In this module, students solidify foundational programming concepts by programming Karel, a robot that navigates a grid world. They write functions, use for loops, while loops, if/else statements, top-down decomposition, and develop debugging and algorithmic thinking skills.

Topics Covered	• Introduction to Programming with Karel • Functions in Karel • Top-Down Design and Decomposition • Commenting Your Code • Abstraction and Super Karel • For Loops and While Loops in Karel • If Statements and If/Else Statements • Control Structures Example • Debugging Strategies and Algorithms

Example Assignments	• **Your First Karel Program**
	  ○ Students write a program to move Karel to a tennis ball and pick it up, learning the basic movement and pick-up commands.
	• **Short Stack**
	  ○ Students program Karel to move one space and place a short stack of tennis balls, practicing sequencing and put-down commands.
	• **Make a Tower**
	  ○ Students write a program that has Karel build a tower of tennis balls and end up on top of the tower.
	• **Pyramid of Karel**
	  ○ Students program Karel to construct a pyramid with three balls on the first row, two on the second, and one on the third, applying loops and conditionals.

Module 13: Strings (1 week / 5 hours)

In this module, students deepen their understanding of Python strings. They practice indexing, slicing, iterating with for loops, checking membership with the in keyword, and applying built-in string methods to process and transform text.

Topics Covered	• String Indexing • String Slicing • String Immutability • Strings and For Loops • The in Keyword • String Methods

Example Assignments	• **Spelling Bee**
	  ○ Students use string methods and iteration to spell out the letters of a word, one per line, as if in a spelling bee.
	• **Remove All From String**
	  ○ Students write a program that takes two strings from the user and removes all occurrences of one string from the other.
	• **Find the Error**
	  ○ Students identify which of two string operations will cause a runtime error, reinforcing their understanding of immutability and valid index ranges.

Module 14: Classes and Objects (2 weeks / 10 hours)

In this module, students learn object-oriented programming in Python. They define classes, create instances, write methods, overload operators, distinguish class from instance variables, apply inheritance, and work with modules and namespaces.

Topics Covered	• Classes and Objects • Instance Methods • Built-In Methods • Operator Overloading • Class Variables vs. Instance Variables • Inheritance • Hidden Attributes and Namespaces • Modules

Example Assignments	• **The Rectangle Class, Part 1**
	  ○ Students define a Rectangle class with instance variables for width and height, practicing class definition and object instantiation.
	• **The Rectangle Class, Part 2**
	  ○ Students extend their Rectangle class by adding parameters to the constructor so width and height can be set on creation.
	• **The Rectangle Class, Part 3**
	  ○ Students add a get_area method to their Rectangle class that computes and returns the area.
	• **The Rectangle Class, Part 4**
	  ○ Students complete their Rectangle class by adding a get_perimeter method and testing both methods with multiple objects.

Module 15: Social Media and Internet Safety (2 weeks / 10 hours)

In this concluding module, students examine the impact of social media on mental health, behavior, and society. They study persuasive design, digital permanency, misinformation, deepfakes, and how to use social media responsibly and for positive impact.

Topics Covered	• Objectives / Topics Covered • Finding Balance with Technology • Digital Footprint and Permanency • How Social Media Impacts Behavior • Algorithms, Feeds, and Persuasive Design • Using Social Media Safely • Evaluating Misinformation and Deepfakes • Using Social Media for Good

Example Assignments	• **My Digital Balance Plan**
	  ○ Students apply what they've learned about attention design and reward loops to create a personal plan for healthy and balanced technology use.
	• **Spotting Persuasive Design**
	  ○ Students analyze real social media platforms to identify specific examples of persuasive design features such as infinite scroll, notifications, and recommendation algorithms.
	• **Right to Be Forgotten?**
	  ○ Students watch a video and respond to questions analyzing the

	concept of the "right to be forgotten" and whether individuals should be able to erase their digital history. ● **Building a Positive Digital Footprint** ○ Students reflect on their own and their peers' social media activity and write about how they will intentionally build a positive online presence.