Curriculum Guide

Introduction to Computer Science Fundamentals

Course Overview

Introduction to Python is a year-long course that covers the basics of the Python programming language, with a special emphasis on artificial intelligence (AI) applications and data analysis. Students will learn about the fundamentals of computer science (CS) and how it is used in various applications in the world around us, particularly those built using Artificial Intelligence and Data Analysis techniques. They will learn how to code using the programming language Python and understand best practices while completing a variety of exercises, assessments, and projects. Students will also learn data analysis and data visualization techniques as well as the ethical impacts of computing.

Target Learner

The target learner is a 9th grader with no coding experience or CS background.

Duration

150-175 hours

Learning Targets

By the end of this course, students will understand:

- Computer science fundamentals and computational thinking
- The Python programming language
- Data analysis and visualization techniques
- Current computer science and AI applications, as well as their impacts on society and the individual

Course Structure and Delivery

Modes of Instruction and Evaluation

The course consists of instructional videos, online and offline activities, practice questions, formative and summative assessments, and projects.

Course Division

The course is divided into 9 **units**, each consisting of several **lessons**. Each lesson consists of steps.

Each step consists of an instructional video as well as a practice activity for the student. Each step is designed to take no longer than 8-10 minutes to complete and each lesson is designed to take no more than 60 minutes to complete.

Additionally, there are **projects** after every 2-3 units, which allow students to build an application tying together concepts they have learned so far.

There are also **Kernels of Curiosity** sprinkled throughout the course**,** which cover computer science topics beyond programming. Additionally, they discuss more ways Computer Science shows up in society, the impacts of computers on society, and their implications on data storage, security, and privacy concerns

Assessments and Projects

Every lesson concludes with a lesson assessment, and every unit concludes with a unit assessment. There is also a midterm, final, and three projects spread throughout the course. T

Course Content Outline

Unit 1: Fundamentals of Communicating with a Computer	Motivations & Applications
	Print Statements and Commenting
	Data Types
	Variables
	Bugs and Debugging
	User Input and Output
Unit 1 Assessment	
Unit 2: Decision Making with Computers using If-else Statements	If Statements and Operators
	Decision Trees and Flow Charts
	If-Else Statements
	Elif Statements
	Nested if statements
Unit 2 Assessment	
Kernel of Curiosity 1: Data Under the Hood	
Unit 3. Expanding Capabilities with Functions and Libraries	Functions
	Functions with Return Values
	Built-in Functions
	Using Modules and Libraries
Unit 3 Assessment	
Mini Project 1: Sticks Game Implementation	

Unit 4. Storing Data with Lists	Creating Lists
	Indexing into Lists and Changing Elements in Lists
	Appending to and Removing from Lists
	Tuples and Nested Lists
Unit 4 Assessment	
Midterm Exam	
Unit 5. Repetition and Iteration with Loops	While Loops
	For Loops
	Looping over Strings and Lists
	Nested Loops
	Loops for Data Scientists
Unit 5 Assessment	
Kernel of Curiosity 2: Theory of Computing	
Unit 6. Storing Data with Dictionaries	Creating a Dictionary
	Accessing Elements in a Dictionary
	Changing and Adding Elements in a Dictionary
	Complex data structures
Unit 6 Assessment	
Kernel of Curiosity 3: Computer Systems and Networks	
Mini Project 2: Rap Generation	
Kernel of Curiosity 4: Access and Ownership	

Unit 7: Creating Custom Data Types with Classes	Creating Classes
	Adding Behavior to Classes with Methods
Unit 7 Assessment	
Unit 8: Data Analysis Life Cycle	What is the Data Analysis Life Cycle?
	Importing and Exploring Datasets
	Cleaning Datasets
	Uncovering Information from Data
Unit 8 Assessment	
Unit 9: Data Visualization	Purpose of Data Visualizations
	Creating Bar Plots
	Creating Line Plots and Scatter Charts
	Using Visualizations to Find Trends in Data
Unit 9 Assessment	
Kernel of Curiosity 5: Community and Access	
Final Project: Most Popular Movie Prediction	
Final Exam	

Standards Coverage

The course covers both the Tennessee State Standards for High School Computer Science (page 22) as well as AP CS Principles Standards.

These standards are reinforced throughout the course. The table below cites one example of where each standard is met."

Tennessee State Standard	One time where standard appears
CS.AT Algorithmic Thinking	
Use lists to simplify solutions, generalizing computational problems instead of repeatedly using simple variables.	Unit 4
Systematically design and develop programs for broad audiences by incorporating feedback from users.	Final Project
Create prototypes that use algorithms to solve computational problems by leveraging prior student knowledge and personal interests.	Mini Project 1
Use effective communication and accurate computer science terminology to explain problem solving when completing a task.	Mini Project 1
CS.DA Data Analysis	
Create computational models that represent the relationships among different elements of data collected from a phenomenon or process.	Unit 8
Utilize data to answer a question using a variety of computing and data visualization methods.	Unit 9
Use data analysis tools and techniques to identify patterns in data representing complex systems.	Unit 9

CS.NI Networking and the Internet	
Explain the tradeoffs when selecting and implementing cybersecurity recommendations.	Kernel of Curiosity 3
Identify laws regarding the use of technology and their consequences and implications.	Kernel of Curiosity 4
Evaluate strategies to manage digital identity and reputation with awareness of the permanent impact of actions in a digital world.	Kernel of Curiosity 5
Demonstrate how to apply techniques to mitigate effects of user tracking methods.	Kernel of Curiosity 4
Show an understanding of the ramifications of end-user license agreements and terms of service associated with granting rights to personal data and media to other entities.	Kernel of Curiosity 4
Recommend security measures to address various scenarios based on factors such as efficiency, feasibility, and ethical impacts.	Kernel of Curiosity 4
Demonstrate a fundamental understanding of API (Application Programming Interface).	Kernel of Curiosity 3
CS.PC Programming Concepts	
Choose and apply an appropriate iterative design process to systematically test and refine a program to increase performance.	Mini Project 1
Develop a plan to manage and assign data values of different types (strings, numeric, character, integer, and date) to a variable	Unit 1
Create and refine programs with Boolean conditionals to demonstrate the use of branches and logical operators.	Unit 2
Design and develop iterative programs that combine control structures, including nested loops and compound conditionals.	Unit 5
Create parameters to organize a program to make it easier to follow, test, and debug.	Unit 3

Incorporate existing code, media, and libraries into original programs, and give proper attribution.	Unit 3
Debug (identify and fix) errors in an algorithm or program that includes sequences and simple and complex loops following a two-step debugging process.	Unit 6
CS.IC Impacts of Computing	
Discuss the ethical ramifications of hacking and its impact on society.	Kernel of Curiosity 5
Explain the privacy concerns related to the collection and generation of data through automated processes that may not be evident to users	Unit 8
Explain the positive and negative consequences that intellectual property laws can have on innovation.	Kernel of Curiosity 4
Use tools and methods for collaboration on a project to increase connectivity of people in different cultures and career fields.	Projects
Research the impact of computing technology on possible education and career pathways.	Unit 1
Predict how computational innovations that have revolutionized aspects of our culture might evolve.	Kernel of Curiosity 5

Unit 1: Fundamentals of Communicating with a Computer

Students will explore the world of computer science, focusing on data science, artificial intelligence, and programming. They will learn about algorithms, data types like strings and integers, and how computers process data. Students will be introduced to the Python programming language and will practice writing print statements. They will also understand how AI systems and humans learn from information, and the significance of AI ethics and safety. Furthermore, students will delve into the usage of variables in programming, making code more efficient and flexible, and learn the rules for naming variables and using f-strings. Finally, they will discover how to create interactive programs by getting input from users, working with different data types, and even build a trivia chatbot.

Learning Targets

Students will know:

- Terminology: algorithm, computer science, data science, artificial intelligence, programming language, command, program, character, bug, syntax
- Different data types including: strings, integers, floats
- Different subfields of computer science
- Syntax for print commands, input commands, variables, and f-strings

Students will understand that:

- Computer science has applications in many different areas
- Data science involves computers analyzing information to solve problems and artificial intelligence involves using information to make decisions
- Humans write computer programs using a programming language
- There are many different kinds of data types that a computer uses
- It is important for a computer to have accurate input data to produce correct results

Students will be able to:

- Write commands using print statement, f-strings, and variables
- Use the string, integer, and float data types and convert between them
- Create interactive programs using the input() command
- Debug print commands, variables, and input() commands

Unit 2: Decision Making with Computers using If-Else Statements

Students will learn about conditional statements, decision trees, and loops. They will explore the 'if-else' statement, booleans, comparison and logical operators, and practice fixing buggy code. The lesson will also teach students about creating multiple execution paths with if-else statements and visualizing those paths with decision trees and flow charts. Finally, they will discover while loops for automating repetitive tasks, using break statements, and avoiding infinite loops. Students will learn and practice these concepts through the real-world example of autonomous vehicles.

Learning Targets

Students will know:

- Terminology: decision tree, decision tree flowchart, conditional
- Syntax for boolean expressions, mathematical operators, if-else statements, elif-statements, nested if statements, and while loops

Students will understand that:

- Boolean statements can be used with mathematical operators to determine if a statement is true or false
- Decision tree flowcharts are used to visually represent the process of rule-based decision making
- An If-else statement is used to represent a decision tree flowchart with code
- While loops can be used together with if-else statements to repeat a block of code until a certain condition is reached

Students will be able to:

- Write programs with boolean expressions and operators
- Write if-else-elif statements and nested if statements
- Write while loops with if-else statements to repeat code until some condition is met
- Debug broken control flow logic consisting of boolean, if-else-statements, and while loops

Unit 3: Expanding Capabilities with Functions and Libraries

Students will explore the fundamentals of functions in programming, including purpose, structure, input parameters, output results, and abstraction. They will compare programming functions to mathematical functions and gain hands-on experience creating, debugging, and using functions. Students will also learn about return statements, decomposition, global and local variables, and practice problem-solving with interactive coding exercises. Students will discover Python modules, focusing on the Python standard library, math module, and random module, and understand how to import and use their functions. The lesson highlights libraries, documentation, and abstraction.

Learning Targets

Students will know:

- Terminology: function, void, non-void, calling functions
- Syntax for functions and importing Python libraries

Students will understand that:

- Functions organize and reuse code that performs a specific task
- Functions take input and return output
- Python libraries include functions that have already been written to perform a variety of tasks

Students will be able to:

- Write their own functions to perform specific tasks
- Call functions with different inputs to product different outputs
- Import and use functions from Python libraries
- Debug bugs related to writing and calling functions

Unit 4: Storing Data with Lists

Students will explore lists in Python, learning how to store, organize, and access multiple pieces of data in a single variable. They will practice creating lists, accessing elements using indexes, and updating items within lists. Students will also discover list slicing and how strings can be treated as collections of characters. They will learn about the CRUD (Create, Read, Update, Delete) operations in Python, using methods like append, pop, and insert. Students will delve into nested lists and tuples, which help manage data in table-like structures and store immutable information, respectively. Through various activities, students will develop skills in working with lists and tuples to create functions and games.

Learning Targets

Students will know:

- Terminology: data structure, list, tuple, index, CRUD (create, remove, update, delete)
- Syntax to create lists, index into lists and tuples to access elements, add elements to lists, remove elements from lists

Students will understand that:

- Lists and tuples are used to store many pieces of data
- Data can be accessed, updated, added, and removed from lists
- Data in lists can be changed but data in tuples cannotTuples store data that cannot be changed

Students will be able to:

- Create lists and tuples
- Access, update, append, and remove elements to and from lists
- Create nested lists
- Debug errors that result in working with lists and tuples in computer programs

Unit 5: Repetition and Iteration with Loops

Students will master while and for loops in Python, learning to create, avoid mistakes, and solve real-world problems. They'll explore indexing, the enumerate function, and data cleaning while iterating through lists and strings. Students will be introduced to advanced looping techniques such as nested loops and looping over nested data structures with practical examples. By the end of the lessons, students will have a deeper understanding of how loops enhance programming efficiency and solve complex problems.

Learning Targets

Students will know:

- Terminology: infinite loop, finite loop, for loop, while loop
- Syntax for for loops, while loops, range function, incrementing and decrementing while iterating

Students will understand that:

- For loops are often used with lists and nested for loops are often used with nested lists
- For loops are used primarily used to iterate through a list of data and while loops are used primarily to repeat code until some condition is met
- There is a risk of running an infinite loop if the condition for a while loop is not properly defined

Students will be able to:

- Use and implement while loops and for loops with lists and strings
- Iterate over a nested list with a for loop
- Iterate “forwards” vs “backwards” in a list using a for loop
- Debug broken loop logic

Unit 6: Storing Data with Dictionaries

Students will explore Python dictionaries and lists, learning to create, access, and modify them. They'll understand mutable and immutable objects, use loops like for loops with the range() function, and work with nested data structures. Focusing on real-life scenarios like building a music playlist, students will create algorithms for tasks such as counting song play counts and identifying popular artists. They'll gain valuable experience in Python programming through hands-on exercises and practical applications.

Learning Targets

Students will know:

- Terminology: Dictionary, key-value pair
- Syntax to create dictionaries, index into dictionaries to access and change a value for a given key, add key-value pairs to dictionaries, remove key-value pairs from dictionaries

Students will understand that:

- Dictionaries are used to store data in key-value pairs where looking up data with a single key can be helpful
- Each key in a dictionary must be unique but the same value can exist for two different keys
- Key-value pairs in dictionaries can be accessed, changed, added, or removed

Students will be able to:

- Create dictionaries
- Access, update, append, and remove key-value pairs to and from dictionaries
- Create and use complex data structures combining lists, dictionaries, and loops
- Debug errors that result in working with dictionaries in computer programs

Unit 7: Creating Custom Data Types with Classes

Students will explore Python classes, constructors, and methods, learning to create custom data types and understand their real-world applications. They'll develop coding skills by defining classes, creating constructors, and exploring method types in object-oriented programming. Through hands-on activities, students will gain experience in creating unique objects and using methods effectively, providing them with a strong foundation in Python classes and their practical uses.

Learning Targets

Students will know:

- Terminology: class, object, static method, instance method, class method
- Syntax to create a class, class objects, and various methods

Students will understand that:

- Classes are used to build custom data types
- Class methods can change class variable values that would apply to all objects of a class and they enable different kinds of class objects for different use cases
- Instance methods can change instance variable values that would apply to a specific instance or object of the class
- Static methods cannot change either the instance or class variables

Students will be able to:

- Create and instantiate a class to create a custom data type
- Create instance methods, class methods, and static methods
- Call various methods on class objects
- Debug creating and using classes

Unit 8: Data Analysis Life Cycle

Students will learn the data analysis lifecycle, working with the Pandas library in Python to manage data sets. They'll practice importing, exploring, selecting, and manipulating data, as well as handling missing values and using data visualization techniques. By mastering data formatting, cleaning, and analyzing techniques such as filtering, sorting, and groupby aggregations, students will be well-equipped to make informed decisions based on accurate, consistent data analysis

Learning Targets

Students will know:

- Terminology: frequency, mean, median, mode
- Syntax to use Pandas to import, process, modify and analyze data

Students will understand that:

- Analyzing and processing data accurately is a key part of building reliable AI systems
- Analyzing data can help us understand patterns and key features of a dataset that can be used to make conclusions and predictions

Students will be able to:

- Use Pandas to import datasets
- Use Pandas to format and prepare datasets for analysis
- Use Pandas to modify a dataset using python libraries (e.g. add/remove columns, remove outliers, etc.)
- Analyze key features of a data set, including max, min, frequency, mean, median, mode

Unit 9: Data Visualization

Students will explore data visualization and analysis using Python libraries. They'll learn to create bar, line, and scatter charts for different data types, and customize plots for readability. By analyzing rainfall data, students will understand the usefulness of data analysis in real-life scenarios. The lesson covers four analysis types: descriptive, diagnostic, predictive, and prescriptive, highlighting their role in understanding trends, making predictions, and informing decisions.

Learning Targets

Students will know:

- Terminology: bar plot, scatter plot, linear plot, correlation, causation
- Syntax to use Matplotlib and Seaborn to graph visualizations

Students will understand that:

- Visualizing data can help us understand key features and patterns of a dataset that can be used to make conclusions and predictions about the data
- Having incorrect data can lead to misleading visualizations and incorrect conclusions about the data
- Correlation between data does not necessarily mean causation

Students will be able to:

- Use the Python libraries Matplotlib and Seaborn to create visualizations of various features and patterns within a dataset
- Customize graphs and plots to include titles and axis labels
- Plot multiple lines in one plot to better compare trends between different data

Projects Overview

The projects are intended to enable students to create relevant and interesting applications by combining concepts they have learned so far in the course. They are designed to give students the opportunity to work with larger codebases in a collaborative way that will be similar to what they may have to work with in a real-world industry environment.

Mini Project 1: Sticks

Students will create a bot that plays the classic hand game Sticks, translating game rules and strategies into code. They will explore the concepts of game states, valid moves, and exceptions in programming. Students will also learn about solved games, game complexity, and game theory The project requires knowledge of variables, conditionals, functions, and while loops.

Mini Project 2: If Wikipedia could rap

Students will create a rap lyric generator using data from Wikipedia. First, students will retrieve text from Wikipedia articles. Then, they will combine their knowledge of Python with functions from external Python libraries to find rhyming words, word counts, and syllable counts to configure their rap lyrics. The project covers concepts such as nested loops, conditional statements,libraries, and lists.

Final Project: Movie Prediction Machine

Students will leverage their Python and data science skills to analyze a large data set about movies. Students will use Python library Pandas to process the data set and create data visualizations to understand what makes movies successful. Students will then use those insights to pitch an original movie and feed their pitch into a generative AI model to create a poster of their idea.

Kernels of Curiosity Overview

The kernels of curiosity cover concepts beyond programming that are critical to their understanding of how computer science shows up in the real world in different ways, and the risks and limitations associated with it.

Kernel of Curiosity 1: Data under the Hood

Students will explore the world of binary numbers. They will learn to translate decimal numbers into binary and understand the difference between base-10 and base-2 number systems. Students will also discover how computers use bits to store digital information, and they will discuss the causes of integer overflow and floating point imprecision. Students will also explore the concept of data compression, which helps to reduce the size of digital data.

Kernel of Curiosity 2: Theory of Computing

Students will learn about algorithms, computational complexity, and Big-O notation as a measure of algorithm efficiency. They'll explore linear and binary search methods, using a card deck and phone book as examples. Students will learn about undecidable problems in computer science, like the halting problem and the game completion problem, that can't be solved in every instance using computers. They'll also understand the efficiency of algorithms and how it determines the feasibility of solving problems.

Kernel of Curiosity 3: Computer Systems and Networks

Students will explore how the Internet works and how interconnected computer devices transfer data. They'll learn about data and metadata fields of Internet packets using protocols like IP, TCP, and UDP. The lesson also covers parallel and distributed computing, focusing on how humans and computers can work together to solve problems at scale.

Kernel of Curiosity 4: Privacy and Ownership

Students will learn about the privacy implications of many common technologies and why privacy protections are valuable. Students will learn about the question of intellectual ownership in software and how it affects engineer responsibility and public policy. Students will discuss the importance of addressing legal and ethical concerns in computing.

Kernel of Curiosity 5: Community and Access

In this lesson, students learn how parallelism helps both computers and humans find solutions at scale. Students will conceptually learn a parallel approach to Python for-loops and practice using formulas to compare the runtime of parallel and sequential computing solutions. Distributed approaches to performing tasks exist for not only computers, but also humans. In particular, a lot of data processing and creation is performed in parallel by many people doing small tasks across the world. While this sharing of resources greatly expands the scope of computing and engineering, it also raises new legal and ethical challenges around intellectual ownership, engineer responsibility, and equitable access to technology.