Python for Beginners

Learn Python from scratch and build real projects, ideal for those new to coding.

Python Instructor: Chris Taylor • 5 sections • 48 lessons • 17.00h total

View this course online Browse all courses

About this course

Starting to learn programming can feel daunting, especially with so many languages and resources available. This course cuts through the noise, focusing on Python, one of the most accessible and versatile languages for beginners. You'll build a solid foundation without any prior experience. Through a series of video lessons and hands-on projects, you'll learn Python syntax, data structures, control flow, and how to work with libraries. By the end, you'll be able to write your own scripts and applications, ready to tackle more advanced topics.

Course curriculum

Foundations and Preparation

- Setting Up Your Python Environment (12m)
  Step-by-step guide to installing Python and setting up a code editor on your system.
- Understanding Variables and Data Types (15m)
  Learn how to store and manipulate data using variables, integers, strings, and booleans.
- Basic Operators and Expressions (10m)
  Use arithmetic, comparison, and logical operators to perform calculations and make decisions.
- Working with Strings and String Methods (18m)
  Manipulate text with string operations, slicing, and common methods like split and join.
- Control Flow with If Statements (14m)
  Write code that makes decisions using if, elif, and else blocks for different outcomes.
- Looping with For and While (16m)
  Repeat actions efficiently using loops, including iterating over lists and using break statements.
- Introduction to Functions (20m)
  Define reusable blocks of code with functions, parameters, and return values for better organization.

Core Concepts

- Working with Lists and Tuples (22m)
  Store and access collections of data using lists and tuples, including indexing and slicing.
- Using Dictionaries and Sets (19m)
  Manage key-value pairs and unique items with dictionaries and sets for efficient data handling.
- File Input and Output (17m)
  Read from and write to files on disk to save and load data in your programs.
- Error Handling with Try-Except (15m)
  Handle runtime errors gracefully using try-except blocks to prevent program crashes.
- Modules and Packages (13m)
  Extend Python's functionality by importing modules and organizing code into packages.
- List Comprehensions and Generators (18m)
  Write concise code to create lists and iterate over data with comprehension syntax.
- Object-Oriented Programming Basics (25m)
  Understand classes and objects to structure code around real-world entities for reusability.
- Classes and Objects in Depth (20m)
  Define methods, attributes, and constructors to build custom classes for your applications.
- Inheritance and Polymorphism (22m)
  Extend existing classes and use polymorphism to create flexible and scalable code designs.
- Working with Standard Libraries (16m)
  Use built-in libraries like math, random, and datetime to perform common tasks efficiently.
- Introduction to Debugging (14m)
  Find and fix errors in your code using print statements and basic debugging techniques.

Deep Dive

- Regular Expressions for Pattern Matching (28m)
  Search and manipulate text using regex patterns for tasks like data validation and extraction.
- Handling Dates and Times (15m)
  Work with the datetime module to manage dates, times, and time zones in your programs.
- Building GUIs with Tkinter (30m)
  Create simple graphical user interfaces using Tkinter for desktop applications.
- Database Integration with SQLite (25m)
  Store and retrieve data from databases using Python's sqlite3 module for persistent storage.
- Web Scraping Basics (22m)
  Extract data from websites using libraries like BeautifulSoup and requests for automated gathering.
- Introduction to APIs (18m)
  Connect to external services and fetch data using APIs with Python's requests library.
- Working with JSON and CSV Files (16m)
  Read, write, and parse data in JSON and CSV formats for data interchange and storage.
- Project: Build a Simple Calculator (32m)
  Apply concepts to create a calculator app that handles basic arithmetic operations.

Advanced Application

- Project: Todo List Application (28m)
  Develop a command-line todo list app to practice data management and user interaction.
- Project: Web App with Flask Basics (35m)
  Build a simple web application using Flask to understand routing and templates.
- Data Analysis with Pandas Introduction (30m)
  Use pandas to load, manipulate, and analyze tabular data for insights.
- Automation Scripts for Daily Tasks (25m)
  Write scripts to automate repetitive tasks like file organization or email sending.
- Working with Images Using Pillow (20m)
  Process and edit images programmatically with the Pillow library for basic graphics work.
- Introduction to Machine Learning with Scikit-learn (32m)
  Get started with machine learning by building simple models for classification or regression.
- Project: Simple Chatbot (27m)
  Create a basic chatbot that responds to user input using logic and string manipulation.
- Version Control with Git Basics (18m)
  Track changes in your code using Git for collaboration and project management.
- Testing with Pytest (22m)
  Write and run tests to ensure your code works correctly and catches bugs early.
- Performance Optimization Tips (15m)
  Improve code speed and efficiency with techniques like profiling and algorithm choices.

Implementation & Next Steps

- Capstone Project Planning (20m)
  Outline a larger project, define requirements, and plan the structure for your final build.
- Capstone Project: Build a Personal Blog Engine (45m)
  Apply all skills to create a blog engine with user posts, comments, and basic styling.
- Deploying Python Applications (25m)
  Learn how to deploy your apps to the cloud or local servers for real-world use.
- Career Paths in Python (18m)
  Explore job roles like web developer, data analyst, or automation engineer using Python.
- Contributing to Open Source Projects (22m)
  Get involved in open source communities to improve skills and collaborate on projects.
- Staying Updated with Python Community (15m)
  Follow blogs, forums, and events to keep up with Python updates and best practices.
- Advanced Libraries Overview (20m)
  Survey libraries for areas like web development, data science, and machine learning.
- Python in Data Science (28m)
  Dive into data science workflows with Python, including visualization and modeling.
- Python in Web Development (25m)
  Expand on web frameworks like Django and Flask for full-stack development.
- Python for Automation and Scripting (18m)
  Use Python to automate system tasks, network operations, and more for efficiency.
- Final Review and Q&A (30m)
  Recap key concepts, address common questions, and solidify your understanding.
- Course Wrap-Up and Resources (12m)
  Summarize the course and provide recommendations for further learning and practice.