

**Cython Programming**

**Course Number:** PYTH-284
**Duration:** 2 days

**Overview**

[Cython](https://cython.org/) is a programming language that extends Python with C-like syntax. It allows programmers to write Python code that can be compiled into C code, resulting in significant performance improvements. Cython is often used for high-performance computing, scientific computing, and machine learning applications. The Cython programming training course teaches attendees how to write highly performant code with Cython and use NumPy arrays with Python, C, and Cython.

**Prerequisites**

This Cython course is for students who have completed a C and Python training class or are self-taught programmers with many years of continuous Python and C experience.

**Materials**

All Cython training students receive comprehensive courseware.

**Software Needed on Each Student PC**

* Any Windows, Linux, or Mac OS X operating system
* Standard Python 3.x or Anaconda Python 3.x
* Visual Studio Code (other editors may be used)

**Objectives**

* Set up a C programming environment
* Explore how to program and run C programs
* Learn code organization through functions and include files
* Understand static typing and dynamic memory allocation
* Practice writing C code called from Python

**Outline**

* Introduction
* Development Environment
	+ Overview of Virtual Machine for Class
	+ Visual Studio Code Extensions for C
	+ Debugging C with VS Code
* C Programming Language
* Development Tools
	+ Makefile
	+ CMake
	+ CPP Check
* Program Structure
	+ Main Function
	+ Include Files
	+ Header Files
	+ Compile to Executable
	+ Compile to Shared Library
	+ Using Libraries
	+ Common Compile Options
* Data Types
	+ Numbers
	+ Strings (Arrays of Characters)
	+ Arrays
	+ Structs
	+ Type Defs
	+ Type Casting
	+ Static Typing vs. Dynamic Typing
	+ Loose Typing vs. Strong Typing
	+ Static Types
* Control Flow
	+ If
	+ Switch
	+ While
	+ Do-While
	+ For
	+ Goto
* Functions
	+ Define a Function
	+ Call a Function
	+ Passing Arguments/Parameters
	+ Return Values
	+ Function References
* Pointers
	+ Pass By Value vs. Pass By Reference
	+ Dynamically Allocating Memory on the Heap
	+ Principles and Practices of Dynamic Memory Allocation
	+ Linked Lists
	+ Untyped Pointers
* Python and C Integration
	+ Numpy Arrays
	+ Ctypes FFI
	+ Python Struct and Structure
	+ Create a Simple C-Extension for Python
	+ Debug a C-Extension for Python
	+ Review C Code generated by Cython
	+ Basic Profiling of C functions
* Conclusion