CPF Syllabus and Learning Outcomes

Certificate in Python for Finance (CPF)

September 19, 2026

Program at a Glance

The Certificate in Python for Finance (CPF) is a structured, applied program in quantitative finance, Python, and modern financial engineering. The standard cohort rhythm spans 16 weeks: 12 weeks of structured curriculum followed by four weeks of practice and final-project preparation. Delegates choose one primary specialization for their first run, while retaining access to the other Core Classes and the option to return later for a second or third specialization.

Contents

1 Purpose and Audience

CPF is designed for professionals who want to connect financial theory, mathematics, data, and software into reproducible quantitative workflows. It is relevant to asset management, quantitative research, algorithmic trading, derivatives, risk management, investment analytics, and financial data or AI engineering.

This syllabus is intended to give delegates, managers, and sponsoring employers a clear view of the curriculum. It describes the program at the class, topic, and learning-outcome level. It does not attempt to reproduce the full catalog of individual videos or sessions.

2 Program Architecture

CPF combines a finance-focused Core Track with an Engineering Track that provides the technical foundations needed to implement, test, and communicate the finance material.

Layer	Role in the program	Representative content
Core Track	Develops financial and quantitative understanding through the current Core Classes.	Asset Management; Algorithmic Trading; Derivatives Analytics with Python; and Python and AI for Rates, Bonds, and Credit.
Bridge Topics	Connects the foundational theory to the specialization pathways.	Covariance geometry; Efficient Markets Hypothesis; Fundamental Theorems of Asset Pricing.
Engineering Track	Builds the Python, data, systems, and AI capabilities needed for practical implementation.	Python; Linux; data science; tools; deep learning; natural language processing.
Practice Phase	Integrates the material in applied work and a final project.	Reproducible notebooks, static data, GitHub, Google Colab, and project communication.

This syllabus describes the current CPF offering. Python and AI for Rates, Bonds, and Credit is introduced with the September 2026 cohort, which started on 7 September 2026. The class is therefore part of CPF's current development path; its detailed weekly content will be incorporated as the class materials come online. The common theory foundation, Engineering Track, and option to complete additional Core Classes in subsequent runs remain unchanged.

The first cohort run is intentionally focused. Delegates normally select one primary specialization for deeper work while sampling other available pathways. This is a sequencing decision, not a restriction on future study: the same delegate can subsequently complete a second or third Core Class and build a broader CPF portfolio over time.

3 Learning Goals

The goals progress from technical foundations, through financial reasoning and quantitative models, to machine-learning and AI methods and their responsible professional use.

By the end of the program, a successful delegate should be able to:

- use Python to acquire, clean, transform, visualize, and analyze financial data and organize reproducible data and software workflows;
- apply probability, statistics, linear algebra, and optimization to financial models, diagnostics, and decision problems;
- formulate practical finance questions computationally and explain how assumptions about uncertainty, information, and markets affect results;
- implement and evaluate portfolio, trading, pricing, hedging, and risk models while distinguishing prediction, backtesting, and causal claims;
- select appropriate machine-learning (ML), deep-learning (DL), reinforcement-learning (RL), and artificial-intelligence (AI) methods for specific financial questions;
- formulate ML problems with suitable features, labels, horizons, and constraints, then train and compare supervised or unsupervised models;
- use DL and PyTorch for financial time series, panels, and text, including neural networks, embeddings, attention, and sequence models;
- explain RL states, actions, rewards, policies, and environments, and relate them to allocation, trading, execution, hedging, and other sequential decisions;
- apply natural-language processing and generative AI (GenAI), including large language models, to financial documents, research, coding, and analysis with retrieval, provenance, and disclosure controls;
- evaluate statistical and AI models with time-aware validation, walk-forward analysis, realistic costs, constraints, stress tests, and economically meaningful benchmarks;
- diagnose leakage, overfitting, distribution shift, instability, simulator misspecification, spurious patterns, and implementation failure, while relating model outputs to theory and risk controls;
- build reproducible notebooks and Python modules, validate AI-assisted outputs, protect sensitive information, and communicate evidence, limitations, and conclusions clearly.

4 Mathematical and Quantitative Foundations

Mathematics is treated as an applied tool throughout CPF rather than as an isolated prerequisite. The following topics recur in the finance classes and are reinforced by the Mathematics Basics support class.

Foundation	Topics	Financial application
Linear algebra	Vectors, matrices, linear systems, covariance matrices, eigenvalues, projections, and least squares.	Payoff replication, portfolio variance, factor models, hedging, PCA, and risk decomposition.

Foundation	Topics	Financial application
Probability and statistics	Random variables, expectations, distributions, estimation, hypothesis tests, dependence, and time-series diagnostics.	Asset returns, market-efficiency tests, risk measurement, forecasting, and model evaluation.
Calculus and optimization	Derivatives, gradients, objective functions, constraints, and numerical optimization.	Utility maximization, portfolio construction, calibration, model training, and risk targeting.
Stochastic processes	Random walks, conditional expectations, geometric Brownian motion, trees, and simulated paths.	Dynamic pricing, forecasting, Monte Carlo, hedging, and sequential decisions.
Numerical methods	Simulation, interpolation, Fourier methods, calibration, and least-squares Monte Carlo.	Derivative valuation, implied parameters, American options, and model comparison.

The Mathematics Basics resources provide a systematic review of calculus, linear algebra, probability, statistics, optimization, and Python-based computation. Delegates can use them before the cohort or return to them when a later finance topic requires additional mathematical support.

5 Sixteen-Week Curriculum Map

The schedule below describes the default cohort rhythm. Engineering classes run in parallel with the Core Track as staged support. The engineering column describes available resources and the capabilities prioritized for the current week; it does not imply that delegates are expected to master every engineering topic simultaneously.

Week	Core finance focus	Engineering and support	Practice emphasis
1	Financial Theory with Python: finance, Python, and the two-state economy.	Python key skills first; Linux, tools, deep learning, NLP, and optional microeconomics available as staged support.	Payoff vectors, expectations, and basic reproducible notebooks.
2	Financial Theory with Python: three-state economies, incompleteness, and price bounds.	Continue Python and Linux foundations; revisit mathematics as needed.	Replication, attainable payoffs, and numerical checks.
3	Financial Theory with Python: preferences, equilibrium, static and dynamic economies.	Continue the first Engineering Track sequence.	Utility, pricing kernels, trees, and simulation.
4	Bridge: covariance geometry, EMH, and the Fundamental Theorems of Asset Pricing.	Targeted tools, Linux, deep learning, and NLP reinforcement.	Connect mathematical foundations to later specialization choices.

Week	Core finance focus	Engineering and support	Practice emphasis
5	Launch the established Core Classes: Asset Management, Algorithmic Trading, and Derivatives Analytics with Python; introduce Rates, Bonds, and Credit with the September 2026 cohort.	Python, Linux, and tools prioritized; other Engineering Track resources remain available for support.	Select a primary pathway and sample other available pathways.
6	Continue the established pathways with portfolio, trading, and derivatives foundations.	Data and software workflow development.	Build small, testable finance workflows.
7	Extend the selected pathway with deeper models and applications.	Data, tools, deep learning, NLP, and volatility support.	Compare assumptions, inputs, and diagnostics.
8	Continue specialization work and connect models to realistic data.	Reproducible engineering and applied AI.	Evaluate results with visual and statistical diagnostics.
9	Advanced pathway topics in risk, prediction, backtesting, or pricing.	Deep learning, NLP, Linux, and data workflow reinforcement.	Identify leakage, overfitting, numerical error, and model risk.
10	Extend the selected pathway toward production-quality analysis.	Python, tools, deep learning, NLP, and volatility support.	Improve code structure, testing, and reporting.
11	Integrate models, data, and financial interpretation.	Engineering Track consolidation.	Compare alternative specifications and explain trade-offs.
12	Complete the scheduled Core Track sequence.	Review and targeted special topics.	Define a project direction and consolidate evidence.
13	Practice phase: apply the selected pathway to a focused problem.	Individual support and technical consolidation.	Project design, data preparation, and proposal refinement.
14	Practice phase: implement and test the analysis.	Reproducible environments and workflow review.	Notebook, modules, tests, and diagnostics.
15	Practice phase: interpret and communicate results.	Documentation, reporting, and presentation support.	Limitations, robustness, and professionally relevant conclusions.
16	Practice phase: finalize the project.	Final workflow and delivery checks.	Submit a self-contained, reproducible final project.

5.1 Specialization progression

The main table shows the common cohort rhythm. The following compact view shows how the three established Core Classes become more specific during Weeks 5–12. Python and AI for Rates, Bonds, and Credit is introduced with the September 2026 cohort; its detailed progression will be added as the class materials come online. The delegate's primary class receives the greatest depth, while other available classes remain useful for orientation and comparison.

Core Class	Weeks 5–8	Weeks 9–12
Python and AI for Asset Management	Portfolio construction, CAPM and factor models, Black–Litterman, risk measures, and risk decomposition.	Data engineering, performance and backtesting, predictive models, AI methods, model risk, and explainability.
Python and AI for Algorithmic Trading	Market-efficiency tests, data workflows, technical strategies, predictive modelling, and vectorized backtesting.	Event-based systems, streaming, online algorithms, broker APIs, monitoring, post-trade risk, and AI-enhanced research.
Derivatives Analytics with Python	Arbitrage pricing, risk-neutral valuation, Black–Scholes–Merton, Greeks, trees, and Monte Carlo.	Fourier methods, American options, calibration, volatility models, calibrated valuation, and dynamic delta hedging.

6 Core Track: Financial Theory with Python

Financial Theory with Python provides the conceptual spine of the program. It starts with low-dimensional state-contingent models and gradually extends the same pricing logic to larger state spaces, simulation, and dynamic economies.

6.1 Topics covered

The topic sequence moves from static claims and replication to preferences, equilibrium, dynamic economies, simulation, and derivative applications.

- Payoffs as vectors, portfolios as linear combinations, and option payoffs as functions of underlying states.
- No-arbitrage, replication, state prices, Arrow–Debreu securities, and risk-neutral probabilities.
- Complete and incomplete markets, attainable payoffs, price bounds, and residual hedging risk.
- Preferences, utility, marginal utility, pricing kernels, and equilibrium prices.
- Large finite-state economies, martingale measures, and Monte Carlo pricing.
- Binomial trees, dynamic trading, geometric Brownian motion, and least-squares Monte Carlo for American options.

6.2 Skills acquired

Delegates learn to translate financial claims into mathematical objects, solve small pricing and replication problems, interpret completeness and incompleteness, and connect formal pricing results to Python simulations and diagnostics.

7 Bridge Topics

The bridge week makes three ideas explicit before delegates choose a primary specialization.

7.1 Covariance geometry

Delegates work with payoff clouds, covariance matrices, eigenvalues, portfolio variance, and geometric representations of diversification and risk. The purpose is to make the linear-algebra structure behind portfolio analysis visible.

7.2 Efficient Markets Hypothesis

Delegates formulate market efficiency as a conditional-expectation restriction and examine practical tests involving returns, autocorrelation, predictability, and statistical evidence. The topic establishes a disciplined baseline for both asset-management research and algorithmic trading.

7.3 Fundamental Theorems of Asset Pricing

Delegates study no-arbitrage, state prices, martingale measures, and market completeness in finite-state models. The low-dimensional setting makes the relationship between linear algebra, arbitrage, replication, and pricing transparent.

8 Specialization Pathways

The Core Classes are complementary rather than mutually exclusive. A delegate normally chooses one as the primary pathway during the first run, while using other available classes for structured exposure. Later cohorts or self-paced study can be used to complete the remaining Core Classes, including Python and AI for Rates, Bonds, and Credit as its materials are added.

8.1 Python and AI for Asset Management

Purpose. This pathway develops the mathematical, statistical, and computational foundations of portfolio construction, risk management, and data-driven investment research.

Topics. Asset classes and mandates; mean–variance theory; CAPM and factor models; Black–Litterman; coherent and convex risk measures; risk decomposition and risk budgeting; active risk management; data engineering; performance measurement; backtesting; linear and generalized linear models; trees and ensembles; deep and sequence models; reinforcement learning; unsupervised representation learning; model risk; explainability; deployment; and LLMs in the asset-management value chain.

Mathematics and methods. Covariance matrices, quadratic optimization, regression, factor exposure, tail-risk measures, dimensionality reduction, portfolio constraints, sequential decisions, and model evaluation.

Applied outcome. Delegates can construct, analyze, and communicate a reproducible investment workflow that connects portfolio objectives, data, models, risk controls, and performance evidence.

8.2 Python and AI for Algorithmic Trading

Purpose. This pathway develops the research and engineering workflow needed to move from a trading hypothesis to a tested and monitored systematic strategy.

Topics. EMH and hypothesis testing; Python infrastructure; scientific Python; static end-of-day data; feature panels; technical strategies; vectorized and event-based backtesting; predictive modelling; reinforcement learning; object-oriented trading systems; real-time data and ZeroMQ; online algorithms; broker APIs; reporting and monitoring; Docker and cloud deployment; logging and failure management; post-trade risk; and AI-enhanced research workflows.

Mathematics and methods. Return distributions, autocorrelation, normality and predictability tests, regression, classification, time-series features, performance attribution, transaction costs, drawdowns, and out-of-sample evaluation.

Applied outcome. Delegates can design a systematic trading research workflow, implement and test a strategy using static data, evaluate realistic risks and frictions, and identify the engineering requirements for a monitored paper-trading or production system.

8.3 Derivatives Analytics with Python

Purpose. This pathway develops the computational-finance theory and numerical methods used to value derivatives, calibrate models, analyze risk, and implement hedging workflows.

Topics. Market-based valuation; arbitrage pricing; risk-neutral valuation; Black–Scholes–Merton; Greeks; Cox–Ross–Rubinstein trees; Fourier-based pricing; American options; Monte Carlo simulation; interest-rate and volatility models; model calibration; calibrated valuation; dynamic delta hedging; and listed volatility and variance derivatives as an extension.

Mathematics and methods. Discount factors, stochastic discounting, stochastic processes, numerical integration, Fourier methods, simulation, optimization, calibration, exercise-policy regression, and sensitivity analysis.

Applied outcome. Delegates can implement and compare derivative pricing methods, calibrate model parameters, interpret numerical diagnostics, and connect model values and Greeks to hedging and risk-management decisions.

9 Engineering Track

The Engineering Track is not an unrelated technology curriculum. It supplies the practical capabilities required to make the Core Track reproducible, maintainable, and useful in a professional environment.

9.1 Python for Finance Key Skills

Python fundamentals, data structures, functions, object-oriented programming, NumPy, pandas, Matplotlib, debugging, narrative notebooks, and core machine learning workflows. The emphasis is on turning a finance question into clear, testable, documented code.

9.2 Linux Basics

Shell commands, package management, shell scripting, SSH, terminal tools, Vim, processes, and a first real-time streaming workflow. The emphasis is on working reliably across local, remote, and cloud environments.

9.3 Tools and Skills

Python environments, Docker, editors, Git, packaging, documentation, publishing, and workflow hygiene. Delegates learn the surrounding practices that make financial code portable, reviewable, and maintainable.

9.4 Data Science Basics

Data ingestion, cleaning, tabular analysis, visualization, financial time series, streaming, SQLite databases, and object-oriented design. The goal is a reusable data pipeline rather than a collection of disconnected examples.

9.5 Deep Learning Basics with PyTorch

Datasets, tensors, autograd, models, training loops, losses, optimizers, regularization, learning-rate schedules, checkpoints, convolutional models, sequences, attention, evaluation, and responsible deployment considerations.

9.6 Natural Language Processing

String operations, regular expressions, tokenization, vector representations, word embeddings, recurrent models, transformers, prompt design, LLM-assisted workflows, and finance-oriented text analytics.

10 Prerequisites and Support Resources

CPF has no programming prerequisite. The practical entry threshold is the bare minimum described in the companion computer-skills guide: delegates should be able to use a keyboard and mouse or trackpad, create and organize folders, open and save files, edit plain text, recognize common filename extensions, and open a terminal to enter simple commands such as `pwd`, `ls`, `cd`, and `mkdir`. These are basic computer-working skills, not Python skills, and can be acquired through a short period of guided practice.

Delegates do not need prior Python, Git, package-management, Docker, Linux administration, cloud, database, machine-learning, or advanced mathematics knowledge. Delegates who want additional preparation can use the Python for Finance Basics, Data Science Basics, Mathematics Basics, or Python Primer for Ramp-Up resources before and during the structured phase.

The support curriculum is deliberately revisitable. A delegate may return to linear algebra before studying covariance, review probability before Monte Carlo, or revisit Python and Linux before implementing a larger workflow. This allows the same program to serve participants with different starting points without reducing the depth of the Core Track.

11 Learning Activities and Evidence of Learning

CPF combines several forms of study:

- structured reading from the core books and supporting notes;
- slide decks for orientation and review;
- exercises that test derivations, implementation, and interpretation;
- Python notebooks and modules that turn concepts into working tools;
- static-data projects with visual and statistical diagnostics;
- coaching material that translates the weekly plan into practical work; and
- a final project that integrates a focused finance question, code, data, documentation, and conclusions.

The emphasis is on evidence that a delegate can explain assumptions, reproduce results, inspect failure modes, and communicate conclusions. The program is therefore more than passive exposure to lectures or software demonstrations.

12 Final Project and Professional Practice

The final project is a central opportunity to demonstrate integrated capability. It should be a focused research or white-paper-style project with a self-contained executable Jupyter Notebook, supporting files, static data, and a detailed `README.md`. The project should run in Google Colab without manual adjustments and should normally complete within 15 minutes.

Delegates are expected to document their methods, use appropriate Python style, explain the financial motivation, distinguish evidence from interpretation, and state limitations. Use of artificial-intelligence tools is permitted when it is disclosed clearly and the delegate remains responsible for understanding, validating, and reproducing the submitted work.

Successful completion

The official certificate is awarded after the delegate completes the structured program and submits a final project that satisfies the project requirements. Exercises and intermediate notebooks provide practice and feedback throughout the program; the final project is the primary integrated assessment. Final projects are reviewed against the standards for focus, technical quality, reproducibility, presentation, and the delegate's ability to explain methods, evidence, and limitations. The review may include feedback and a request for revision before completion.

The certificate therefore attests to the completion of an applied program in financial theory, quantitative methods, Python implementation, engineering practice, and a documented specialization project. It does not imply mastery of every Core Class: the first run is centered on one primary specialization, with additional Core Classes available in later runs.

13 Skills Acquired and Professional Relevance

The following outcomes summarize the professional value of the curriculum.

Capability	**CPF evidence**	**Professional relevance**
Financial reasoning	Explain payoffs, risk, pricing, market assumptions, and model limitations.	More precise communication between finance, risk, and technology teams.
Quantitative analysis	Apply probability, statistics, optimization, and numerical methods to financial problems.	Better model selection, validation, and interpretation.
Python research	Build notebooks, modules, data pipelines, visualizations, and diagnostics.	Faster and more reproducible investment and risk analysis.
Strategy and portfolio evaluation	Backtest, compare, stress, and document portfolio or trading ideas.	More disciplined research and decision support.
Derivative analytics	Price, calibrate, hedge, and analyze derivatives models.	Stronger valuation, risk, and model-governance capabilities.
Engineering practice	Use Linux, Git, Docker, APIs, monitoring, and documentation.	More reliable transition from analysis to maintained workflows.
AI and language workflows	Apply machine learning, deep learning, NLP, and LLM tools with explicit evaluation and guardrails.	Practical readiness for modern financial data and AI initiatives.

14 Resources and Delivery Model

Each major class is supported by a combination of books or notes, overviews, slide decks, exercises, companion code, and recorded explanations. The launchpad provides the central orientation to these materials, while the weekly study plan determines the default sequence for the cohort.

The program supports both cohort-based and self-paced study. A cohort provides weekly structure and a shared rhythm; a self-paced delegate can use the same sequence, return to support material as needed, and complete more than one Core Class in separate passes.

15 Summary

CPF combines financial theory, mathematical foundations, Python implementation, engineering practice, and modern AI methods. The September 2026 cohort introduces Python and AI for Rates, Bonds, and Credit; its detailed class materials are being incorporated as they come online. The first run is sequenced around one primary specialization so that delegates can develop depth without losing the broader view. The design also supports a longer learning path: after completing the first specialization, delegates can return for a second or third Core Class and progressively build a wider quantitative finance skill set.