MARYLAND APPLIED GRADUATE ENGINEERING

PYTHON PROGRAMMING FOR CLOUD ENGINEERING
ENPM818Q

TENTATIVE SYLLABUS – SUBJECT TO CHANGE

Term: *Spring 2026*
Professor: Seyedmohammad, Shams
Course Dates: From September 2nd, 2025 – December 12th, 2025
Course Times: TBD
Classroom: TBD

Course Description

Python Programming for Cloud Engineering provides a comprehensive and practical introduction to Python programming, specifically tailored for applications in cloud-centric environments. The course begins with foundational programming concepts such as data types, control structures, functions, and modular code design, establishing a strong base in problem-solving with Python. It then progresses to more advanced topics including object-oriented programming (OOP), exception handling, and file operations, enabling students to build structured and maintainable code. Students will explore Python's scientific libraries—Numpy, Scipy, and Matplotlib—to perform data manipulation, analysis, and visualization tasks. The course also introduces essential skills in database programming and network development using TCP/UDP sockets. In the final phase of the course, students gain hands-on experience developing dynamic, full-stack web applications using frameworks like Flask and Django, while learning how to integrate these applications with cloud infrastructure and services.

Prerequisites

None.

Learning Outcomes

After successfully completing this course, you will be able to:

1. **Understand and apply core Python programming concepts**, including variables, data types, control structures, functions, and modular programming, to solve computational problems.
2. **Design and implement efficient, reusable software** using object-oriented programming principles such as classes, inheritance, and encapsulation in Python.
3. **Analyze and visualize data** by leveraging Python's scientific libraries, including Numpy for numerical computing, Scipy for advanced analysis, and Matplotlib for graphical representation
4. **Develop skills in database programming**, using Python to connect with and manipulate SQL and NoSQL databases, perform CRUD operations, and manage persistent data storage.
5. **Create basic networked applications using Python sockets**, demonstrating understanding of TCP/UDP communication, client-server interactions, and real-time data transfer.
6. **Build and deploy dynamic web applications** using Python frameworks such as Flask and Django, incorporating form handling, routing, and templates to support user interaction and application logic.

Course Assignments

- **In-class Problem Solving Activities (5% of the student's grade)**
  Short, formative exercises completed during class to reinforce core concepts and encourage active engagement. These may include debugging tasks, code tracing, mini-challenges, or collaborative coding.
- **Weekly Programming Assignments (30% of the student's grade)**
  Hands-on assignments designed to apply weekly topics, covering Python fundamentals, object-oriented programming, file operations, network programming, and interaction with databases and web frameworks.
- **Midterm Exam (20% of the student's grade)**
  A cumulative assessment at the midpoint of the semester evaluating students' understanding of Python programming, including syntax, control structures, OOP, and file operations.
- **Group Capstone Project (30% of the student's grade)**
  A 4–5 week group project where students collaboratively design, develop, and deploy a cloud-based web application using Python and frameworks such as Flask or Django. Project milestones include proposal, development, testing, documentation, and a final presentation.
- **Final Exam (20% of the student's grade)**
  A comprehensive summative exam assessing students' understanding of advanced topics such as cloud integration, web frameworks, and Python-based system development, with practical coding scenarios.

Course Topics

Week	Topic
1	Introduction to Python • *Focus*: Understanding Python's syntax, basic data types, and programming constructs.
2	Control Structures and Data Types • *Focus*: Exploring control flows and advanced data handling in Python.
3	Functions and Modular Programming • *Focus*: Functions, modules, and importing libraries; laying the groundwork for reusability and modular coding.
4	Object-Oriented Programming in Python • *Focus*: Introduction to classes, objects, inheritance, and other OOP concepts.
5	Advanced Object-Oriented Programming • *Focus*: Deep dive into polymorphism, encapsulation, and advanced object-oriented design patterns.
6	File Operations and Exception Handling • *Focus:* Reading/writing files and robust error handling techniques.
7	Scientific Computing and Data Visualization with Python • *Focus*: Introduction to Numpy, Scipy, and Matplotlib for data analysis and visualization.
8	Database Management with Python • *Focus*: Basic database operations with Python, introduction to SQL and NoSQL.
9	First Midterm Exam and Network Programming Basics • *Focus*: Building network applications using Python’s socket library.

	• **Midterm Exam:** covering Python fundamentals (Weeks 1–8).
10	Web Development with Flask • *Focus*: Developing simple web applications using the Flask framework. • *Start of Group Capstone Project*: Teams begin to design and plan their cloud-based applications.
11	Advanced Web Development with Django • *Focus*: Using Django for more complex web development projects.
12	Integrating Python Applications with Cloud Services • *Focus*: Deploying Python applications to cloud platforms, integrating with cloud APIs.
13 - 14	Capstone Project Work • *Focus*: Teams continue to develop, test, and refine their cloud-based applications.
15	Capstone Project Presentations • *Final presentations of group projects*.
16	Final Exam Review and **Final Exam** • *Review session for the Final Exam*.

Note: This is a tentative schedule, and subject to change as necessary – monitor ELMS-Canvas for current deadlines. In the unlikely event of a prolonged university closing, or an extended absence from the university, adjustments to the course schedule, deadlines, and assignments will be made based on the duration of the closing and the specific dates missed.