CS 3270 Course Syllabus

For additional course information, including prerequisites, corequisites, and course fees, please refer to the Catalog: https://catalog.uvu.edu/

Semester: Spring
Year: 2025
Course Prefix: CS
Course and Section #: CS 3270 – X02
Course Title: Python Software Development
Credits: 3

Course Description

Covers the features of the Python programming language. Includes scripting, dynamic typing, data types (sequences, sets, mappings, files, etc.), loops, iterators, generators, functions, coroutines, classes and objects, modules, packages and scope, runtime services, data wrangling, concurrent programming, etc.

Course Attributes

This course has the following attributes:
☐ General Education Requirements
☐ Global/Intercultural Graduation Requirements
☐ Writing Enriched Graduation Requirements
☒ Discipline Core Requirements in Program
☐ Elective Core Requirements in Program
☒ Open Elective
Other: This is not a GE course. It is required for Computational Data Science majors and elective for others.

Instructor Information

Instructor Name: Robert Kumar

Student Learning Outcomes

Upon successful completion, students should be able to:
- Apply the scripting language paradigm
- Explain the advantages and drawbacks of dynamically typed programming languages
- Apply Python's built-in data structures to programming projects
- Apply the principles and techniques of the functional programming paradigm as supported by Python
- Apply the principles and techniques of the object-oriented programming paradigm as supported by Python
- Implement suitable software architectures using Python's module and package features
- Apply the core modules in the Python library to solutions to programming problems

Course Materials and Texts

Python for Geeks by Mohammad Asif
ISBN: 9781801070119

Course Requirements

Course Assignments, Assessments, and Grading Policy
Grading Scale

The following grading standards will be used in this class:

Grade											
A	A-	B+	B	B-	C+	C	C-	D+	D	D-	E
93-100	90+	87+	83+	80+	77+	73+	70+	67+	63+	60+	0+

Assignment Categories and Points

Project Phases (10)	100 each
Module Questions (10)	10 each
Final Project	100
Total	1200

Late Work Statement
Partial credit for points can be deceptive on software assignments, because it is possible to get a "good grade" for software that doesn't work or where a student never learns all the stages needed for completion but passes. If you find yourself in a situation where you cannot complete a phase on time, it is better to submit something complete a little late than to routinely submit partially complete phases always on time.
Late Penalty: 5% per day
Last day to submit anything other than the final phase: Last Day of Classes
Last day to submit final project phase: Last Day of Semester

Module Questions
Module Questions are a set of mostly reflective questions to be answered at the end of each module. They may come from many sources, not just the reading or the phase required--but they will be related to the module and the phase, not out-of-the-blue. The accuracy and quality are scored out of 10 points for each module. Partial scoring is possible. Late penalties, if any apply to that latest submission.

Exams
The only exam is the final project submission.

Discussions
Please use Teams Channel for any course-related questions
Since the course is project based, discussions won't be used.

Assessments
The final project is the only summative assessment in the course.
Project phases are formative and summative: Formative in that they may be resubmitted and regraded based on feedback received, summative in that in aggregate they are the substance of the final project submission.
Module Questions are formative in that a student may answer the questions, do some additional study if needed, and resubmit new answers for a new score.

Required or Recommended Reading Assignments
Each of the following modules have required reading, discussion post and a programming assignment.

Module 1: Optimal Python Development Lifecycle
Module 2: Modularization
Module 3: Advanced Object-Oriented Programming
Module 4: Libraries for Advanced Programming
Module 5: Automated Testing
Module 6: Functional Programming Paradigm
Module 7: Multiprocessing and Asynchronous Programming
Module 8: Using Clusters
Module 9: Web Development
Module 10: Machine Learning
Module 11: Final Project

General Description of the Subject Matter of Each Lecture or Discussion
Module 1: Optimal Python Development Lifecycle
This module introduces the Pythonista community and culture, emphasizing its influence on how Python developers write and share code. It outlines the phases of a professional Python project, covering workflows, documentation, naming conventions, and best practices for testing and deploying code. The primary goal is to establish an effective development and submission environment.

Module 2: Modularization
This module delves into the importance of modularization in software development. It explores modules, packages, and standard methods for sharing and installing software, focusing on their significance in large-scale projects beyond small assignments.

Module 3: Advanced Object-Oriented Programming
Students will deepen their understanding of OOP principles, including composition versus inheritance, duck typing, and alternatives to OOP in Python. The goal is to enhance Python OOP knowledge and explore Pythonic alternatives for writing efficient code.

Module 4: Libraries for Advanced Programming
This module reviews collection data types, iterators, generators, and advanced programming libraries. Students will learn about logging, error handling, file operations, and advanced library usage for building complex Python projects.

Module 5: Automated Testing
This module focuses on the importance of automated testing as a continuous process in software development. It highlights the benefits of test automation for saving time and resources while ensuring software quality and performance.

Module 6: Functional Programming Paradigm
Students revisit functional programming concepts, including nested functions, lambda expressions, decorators, and nested dictionaries. The module emphasizes patterns that make code clearer and less prone to errors, enhancing functional programming skills.

Module 7: Multiprocessing and Asynchronous Programming
This module introduces concurrency and parallel processing using Python's built-in support. Students will learn to design responsive systems and explore asynchronous programming to overcome the limitations of sequential algorithms on single processors.

Module 8: Using Clusters
This module focuses on processing large data volumes using clusters of machines. It introduces PySpark as a tool for efficient parallel data processing, particularly useful for tasks like real-time streaming data.

Module 9: Web Development
Students will explore web development using the Flask framework. The module covers building web applications, interacting with databases, and creating REST APIs or web services, focusing on the multi-tier architecture of web applications.

Module 10: Machine Learning
This module provides an introduction to machine learning by incorporating models to analyze data and make predictions. Students will use a popular ML library to train models on weather data and generate reasonable predictions, focusing on practical applications rather than advanced details.

Module 11: Final Project
Students will demonstrate their knowledge by completing a semester project. This module emphasizes finalizing and refining the project, incorporating feedback, and ensuring code quality. For an added challenge, students can explore authentication and encryption, although these are optional.

Required Course Syllabus Statements

Generative AI
ChatGPT (and similar Tools) in This Course: Use ChatGPT as a learning assistant, not as a crutch. If you use it, cite it at the top of your code. You are responsible to make sure that any code or content does what it is supposed to do and says what you want it to say. Don't accept anything it generates at face value without checking it critically. These days potential employers will expect you to know how to use tools like ChatGPT to generate code, so it is a skill we need to teach you. If it helps you learn some things faster, GREAT because we can spend class time on more interesting topics. Just remember: If you REALLY want to be good, work for it.

Does your instructor REALLY expect you to use GEN AI in this class? REALLY? Yes!
Suggestions for using it responsibly:

If you don't have a clue, use AI to get a clue.
If you don't understand a concept, ask AI for an explanation with examples.
If some code isn't working, ask AI for help on that snippet, including broken APIs.
If you want help on improving your code, ask AI how you might improve some function or section of your code.
Tell AI to guide you toward a solution rather than giving you a solution immediately.
If you use AI, remember to note you've used it in your module docstrings and in your project submission document.

If you feel like you need smaller exercises or practice with some concept before working on some part of your project, use AI to generate exercises for you.
Whatever the AI generates, don't turn in code you could not, would not, or should not have written. That will be penalized heavily. The instructor is the judge of that.

Using Remote Testing Software
☒ This course does not use remote testing software.

☐ This course uses remote testing software. Remote test-takers may choose their remote testing locations. Please note, however, that the testing software used for this may conduct a brief scan of remote test-takers' immediate surroundings, may require use of a webcam while taking an exam, may require the microphone be on while taking an exam, or may require other practices to confirm academic honesty. Test-takers therefore shall have no expectation of privacy in their test-taking location during, or immediately preceding, remote testing. If a student strongly objects to using test-taking software, the student should contact the instructor at the beginning of the semester to determine whether alternative testing arrangements are feasible. Alternatives are not guaranteed.

Required University Syllabus Statements

Accommodations/Students with Disabilities
Students needing accommodations due to a permanent or temporary disability, pregnancy or pregnancy-related conditions may contact UVU Accessibility Services at accessibilityservices@uvu.edu or 801-863-8747.

Accessibility Services is located on the Orem Campus in BA 110.

Deaf/Hard of Hearing students requesting ASL interpreters or transcribers can contact Accessibility Services to set up accommodations. Deaf/Hard of Hearing services can be contacted at DHHservices@uvu.edu

DHH is located on the Orem Campus in BA 112.

Academic Integrity
At Utah Valley University, faculty and students operate in an atmosphere of mutual trust. Maintaining an atmosphere of academic integrity allows for free exchange of ideas and enables all members of the community to achieve their highest potential. Our goal is to foster an intellectual atmosphere that produces scholars of integrity and imaginative thought. In all academic work, the ideas and contributions of others must be appropriately acknowledged and UVU students are expected to produce their own original academic work.

Faculty and students share the responsibility of ensuring the honesty and fairness of the intellectual environment at UVU. Students have a responsibility to promote academic integrity at the university by not participating in or facilitating others' participation in any act of academic dishonesty. As members of the academic community, students must become familiar with their rights and responsibilities. In each course, they are responsible for knowing the requirements and restrictions regarding research and writing, assessments, collaborative work, the use of study aids, the appropriateness of assistance, and other issues. Likewise, instructors are responsible to clearly state expectations and model best practices.

Further information on what constitutes academic dishonesty is detailed in UVU Policy 541: *Student Code of Conduct*.

Equity and Title IX

Utah Valley University does not discriminate on the basis of race, color, religion, national origin, sex, sexual orientation, gender identity, gender expression, age (40 and over), disability, veteran status, pregnancy, childbirth, or pregnancy-related conditions, citizenship, genetic information, or other basis protected by applicable law, including Title IX and 34 C.F.R. Part 106, in employment, treatment, admission, access to educational programs and activities, or other University benefits or services. Inquiries about nondiscrimination at UVU may be directed to the U.S. Department of Education's Office for Civil Rights or UVU's Title IX Coordinator at 801-863-7999 – TitleIX@uvu.edu – 800 W University Pkwy, Orem, 84058, Suite BA 203.

Religious Accommodation

UVU values and acknowledges the array of worldviews, faiths, and religions represented in our student body, and as such provides supportive accommodations for students. Religious belief or conscience broadly includes religious, non-religious, theistic, or non-theistic moral or ethical beliefs as well as participation in religious holidays, observances, or activities. Accommodations may include scheduling or due-date modifications or make-up assignments for missed class work.

To seek a religious accommodation, a student must provide written notice to the instructor and the Director of Accessibility Services at accessibilityservices@uvu.edu. If the accommodation relates to a scheduling conflict, the notice should include the date, time, and brief description of the difficulty posed by the conflict. Such requests should be made as soon as the student is aware of the prospective scheduling conflict.

While religious expression is welcome throughout campus, UVU also has a specially dedicated space for meditation, prayer, reflection, or other forms of religious expression.