Knight Foundation School of Computing and Information Sciences

Course Title: Python Programming II   **Date:** 6/2/2024

Course Number: COP 3045

Number of Credits: 3

Subject Area: Programming
Catalog Description: Object-oriented principles, handling modules, packages, and decorators, working with databases, data structures, and visualization tools. More complex Pythonic solutions for real-world challenges.
Textbooks: Python Object-Oriented Programming: Build robust and maintainable object-oriented Python applications and libraries by Steven F. Lott, Dusty Phillips, 4th Edition. ISBN-13: 978-1801077262. Publisher: Packt Publishing. Date: July 2, 2021.
References (for further reading):
Prerequisites Courses: COP 2047 - Python Programming I
Corequisite Courses: COP 3410 - Computational Thinking

Type: Core Course for BS in Data Science; Elective for CS and IT Majors.

Prerequisites Topics:
1. Programming fundamentals such as control structures, basic data types and structures, functions, and object-oriented paradigm in at least one programming language.
2. Experience in solving simple computational problems using coding.

Course Outcomes:
1. **Implement** an object-oriented applications utilizing principles like objects, interfaces, inheritance, and polymorphism.
2. **Make use of** Python's module system, creating, importing, and distributing packages to enhance code reusability and manage large projects.
3. **Build** Python applications that can connect to, query, and manipulate databases, and other web services.
4. **Apply** complex algorithms and data structures to solve real-world problems.
5. **Utilize** Python libraries like pandas for data processing tasks.
6. **Visualize** data using Python's matplotlib library, producing graphs and charts.
7. **Compare** advanced programming constructs like decorators, context managers, and iterators.
8. **Identify** recursion in order to enhance problem-solving skills for both computational challenges.
9. **Differentiate** the principles and applications of data structures.
10. **Examine** the significance of the principles and applications of data structures in algorithm design and optimization.

Association between Student Outcomes and Course Outcomes

BS in Computing: Student Outcomes Graduates of the program will have an ability to:	**Course Outcomes**
1) Analyze a complex computing problem and to apply principles of computing and other relevant disciplines to identify solutions.	1,2,3,4,5,6,7,8,9
2) Design, implement, and evaluate a computing-based solution to meet a given set of computing requirements in the context of the program's discipline.	1,2,3,4,5,6,7,8,9
3) Communicate effectively in a variety of professional contexts.	
4) Recognize professional responsibilities and make informed judgments in computing practice based on legal and ethical principles.	
5) Function effectively as a member or leader of a team engaged in activities appropriate to the program's discipline.	
Program Specific Student Outcomes	
6) Apply theory, techniques, and tools throughout the data science lifecycle and employ the resulting knowledge to satisfy stakeholders' needs. [DS]	4,5,6,9

Assessment Plan for the Course and how Data in the Course are used to assess Student Outcomes

Student and Instructor Course Outcome Surveys are administered at the conclusion of each offering, and are evaluated as described in the School's Assessment Plan:
https://abet.cis.fiu.edu/

Knight Foundation School of Computing and Information Sciences
COP 3045 Python Programming II

Outline

Topic	Number of Lecture Hours (Total: 37.5 hours = 15 weeks * 2 lectures/week * 1.25 hrs/lecture)	Outcome

1. Objects in Python
  1.1. Init method
  1.2. Constructors
	3	1

2. Object-oriented programming II
  2.1. Encapsulation and class methods
  2.2. Inheritance and polymorphism
	4.5	1

3. Advanced Topics in Object-oriented programming II
  3.1. Abstract classes (ABC) and Interfaces
  3.2. Method overloading
	3.75	1

4. Modules and packages
  4.1. Importing and using modules
  4.2. Creating and distributing packages
	3.75	7

5. Iterators and generators
  5.1. Iterators and the iterator protocol
  5.2. Generators and generator expressions
	3	7

6. Decorators and context managers
  6.1. Introduction to decorators and their usage
  6.2. Creating and using context managers
	3	3,4

7. Recursion, sorting, and searching in Python 7.1. Introductory topics in recursion, sorting, and searching algorithms	4.5	8
8. Data Structures 8.1. Introduction to stacks, queues, trees	3.75	4,9

9. Working with databases
  9.1. Introduction to databases and SQL
  9.2. Python and SQLite
	4.5	3,4,5,6

10. Data processing and visualization
  10.1. Introduction to data processing with pandas
  10.2. Data visualization with matplotlib
	3.75	6,9

Knight Foundation School of Computing and Information Sciences
COP 3045 Python Programming II

Performance Measures for Evaluation

All assignments are assigned through the Canvas course site. The deadlines are strictly enforced. For example, if the deadline is 11:59 PM, any assignment submitted after this time is considered late. It is also each student's responsibility to submit correct files and ensure the submission is successful before the deadline. If students are unable to submit their assignment through Canvas, they will need to send a copy of their assignment to the instructor before the stated deadline. There will be three exams and each exam will be cumulative with an emphasis on the most recently covered material. Exam details will be posted on the Canvas course site (https://canvas.fiu.edu).

Assignment	Points Each	Total Points	Percentage of Final Grade
Quizzes (11-Drop-1)	10	100	10%
Homework Assignments (3)	100	30%	30%
Exam 1	100	20%	20%
Exam 2	100	20%	20%
Exam 3	100	20%	20%
		TOTAL	100%

Letter Grade Distribution Table

Letter	Range%	Letter	Range%	Letter	Range%
A	93 or above	B	82 - 85.9	C	70 - 73.9
A-	90 - 92.9	B-	78 - 81.9	D	60 - 69.9
B+	86 - 89.9	C+	74 - 77.9	F	less than 60

Description of Possible Homework Activities

Homework 1: Object-Oriented Programming (OOP)
Description: Students are required to design a basic library management system using object-oriented programming concepts. The system should allow adding books, checking out books, and displaying all books. Implement classes for books and users, using encapsulation, inheritance, and polymorphism.

Description of Possible Rubric:

Criteria	Excellent (100)	Good (80)	Average (60)	Below Average (40)	Poor (20)	Weight
Implementation of Classes						
- 'Book' class	All required attributes are correctly implemented and function properly	Most required attributes are implemented and function properly	Some required attributes are implemented and function properly	Few required attributes are implemented and function properly	Attributes are not implemented or do not function properly	15%
- 'User' class	All required methods are correctly implemented and function properly	Most required methods are implemented and function properly	Some required methods are implemented and function properly	Few required methods are implemented and function properly	Methods are not implemented or do not function properly	15%
Application of Inheritance	Effective and sophisticated use of inheritance and polymorphism	Good use of inheritance and some use of polymorphism	Basic use of inheritance and little to no use of polymorphism	Ineffective use of inheritance and no use of polymorphism	Does not use inheritance or polymorphism	20%
System Functionality	All system functionalities work perfectly	Most system functionalities work properly	Some system functionalities work properly	Few system functionalities work properly	System functionalities do not work properly	25%
Code Quality	Code is clean, well-commented, and organized	Code is mostly clean, well-commented, and organized	Code is somewhat clean, with some comments and organization	Code is not clean, with few comments and little organization	Code is messy, without comments and organization	25%

Homework 2: Modules and Data Structures

Description: Students need to implement a module that offers operations on stacks and queues. Furthermore, they'll write a test script importing this module, showcasing all its functionalities.

Description of Possible Rubric:

Criteria	Excellent (100)	Good (80)	Average (60)	Below Average (40)	Poor (20)	Weight
Module Creation	Module perfectly defines and implements both stacks and queues	Module correctly defines and implements most of the necessary features of stacks and queues	Module defines and implements basic features of stacks and queues with some errors	Module has attempted to define and implement stacks and queues but with many errors	Module does not define or implement stacks and queues correctly	20%
Stack Operations	All operations (push, pop, display) are correctly implemented and function perfectly	Most operations are correctly implemented and function properly	Some operations are implemented correctly and function with minor issues	Few operations are implemented correctly and have many issues	Operations are not implemented correctly or have critical issues	20%

Queue Operations	All operations (enqueue, dequeue, display) are correctly implemented and function perfectly	Most operations are correctly implemented and function properly	Some operations are implemented correctly and function with minor issues	Few operations are implemented correctly and have many issues	Operations are not implemented correctly or have critical issues	20%
Test Script	Script perfectly imports the module and demonstrates each functionality correctly	Script correctly imports the module and demonstrates most functionalities	Script imports the module and demonstrates some functionalities with minor issues	Script attempts to import the module and demonstrate functionalities but has many issues	Script does not import the module correctly or demonstrate functionalities properly	20%
Code Quality	Code is clean, well-commented, and organized with no errors	Code is mostly clean, well-commented, and organized with minor errors	Code is somewhat clean, with some comments and organization, and few errors	Code is not clean, with few comments and little organization, and many errors	Code is messy, without comments and organization, and has critical errors	20%

Homework 3: Data Processing with Pandas and Visualization

Description: Students are provided with a dataset (e.g., CSV file). They will perform exploratory data analysis using pandas and visualize the results using matplotlib.

Description of Possible Rubric:

Criteria	Excellent (100)	Good (80)	Average (60)	Below Average (40)	Poor (20)	Weight
Data Cleaning	All missing values, outliers, and inconsistencies are perfectly handled	Most missing values, outliers, and inconsistencies are handled well	Some missing values, outliers, and inconsistencies are handled	Few missing values, outliers, and inconsistencies are handled	Missing values, outliers, and inconsistencies are not handled	20%
Data Exploration	Comprehensive insights obtained using a wide range of pandas operations	Good insights obtained using several pandas operations	Basic insights obtained using a few pandas operations	Limited insights obtained with minimal use of pandas operations	No or incorrect insights obtained using pandas operations	20%
Visualization	At least three different types of highly insightful and well-presented plots/graphs	At least three different types of good plots/graphs showcasing insights	At least three different types of basic plots/graphs showcasing insights	Less than three different types of plots/graphs or poorly presented insights	No plots/graphs or incorrect types of plots/graphs used	20%
Analysis Summary	Comprehensive and well-articulated summary of findings	Good summary with a clear presentation of most findings	Basic summary presenting some findings	Limited or unclear summary of findings	No summary or incorrect summary of findings	20%

Knight Foundation School of Computing and Information Sciences
COP 3045 Python Programming II

Code and Documentation Quality	Code is perfectly readable with comprehensive comments and excellent documentation	Code is mostly readable with good comments and documentation	Code is somewhat readable with basic comments and documentation	Code is poorly organized with limited comments and documentation	Code is unreadable with no comments and lacking proper documentation	20%