Introduction to Python

Global Program on Economics and Finance (Fall 2022)

Course Title	Introduction to Python		
Credit	3	**Credit Hours**	48 credit hours
Course Objectives	This course introduces core programming basics—including data types, control structures, algorithm development, and program design with functions—via the Python programming language. The course discusses the fundamental principles of Python Programming, as well as in-depth data and information processing techniques. Students will solve problems, explore real-world software development challenges, and create practical and contemporary applications.		
Course Description	Same as course objectives		

Course Requirements:
None

Teaching Methods:
Lecture and homework assignments

Course Schedule

Lecture 1:
Describe the basic features of an algorithm
Explain how hardware and software collaborate in a computer's architecture
Summarize a brief history of computing
Compose and run a simple Python program

Lecture 2:
Describe the basic phases of software development
Use strings for the terminal input and output of text
Use integers and floating-point numbers in arithmetic operations
Construct arithmetic expressions
Initialize and use variables with appropriate names

Lecture 3:
Import functions from library modules
Call functions with arguments and use returned values appropriately
Construct a simple Python program that performs inputs, calculations, and outputs
Use docstrings to document Python programs

Lecture 4:
Write a loop to repeat a sequence of actions a fixed number of times

Write a loop to traverse the sequence of characters in a string
Write a loop that counts down and a loop that counts up
Write an entry-controlled loop that halts when a condition becomes false

Lecture 5:
Use selection statements to make choices in a program
Construct appropriate conditions for condition-controlled loops and selection statements
Use logical operators to construct compound Boolean expressions
Use a selection statement and a break statement to exit a loop that is not entry-controlled

Lecture 6:
Access individual characters in a string
Retrieve a substring from a string
Search for a substring in a string
Open a text file for output and write strings or numbers to the file
Open a text file for input and read strings or numbers from the file
Use library functions to access and navigate a file system

Lecture 7:
Construct lists and access items in those lists
Use methods to manipulate lists
Perform traversals of lists to process items in the lists
Define simple functions that expect parameters and return values

Lecture 8:
Construct dictionaries and access entries in those dictionaries
Use methods to manipulate dictionaries
Determine whether a list or a dictionary is an appropriate data structure for a given application

Lecture 9:
Explain why functions are useful in structuring code in a program
Employ top-down design to assign tasks to functions
Define a recursive function
Explain the use of the namespace in a program and exploit it effectively
Define a function with required and optional parameters
Use higher-order functions for mapping, filtering, and reducing

Lecture 10:
Determine the attributes and behavior of a class of objects required by a program
List the methods, that realize the behavior of a class of objects
Choose the appropriate data structures to represent the attributes of a class of objects
Define a constructor, instance variables, and methods for a class of objects

Lecture 11:
Recognize the need for a class variable and define it

Define a method that returns the string representation of an object Define methods for object equality and comparisons Exploit inheritance and polymorphism when developing classes Transfer objects to and from files Lecture 12: Measure the performance of an algorithm by obtaining running Analyze an algorithm's performance by determining its order of complexity, using big-O notation Distinguish the common orders of complexity and the algorithmic patterns that exhibit them Design, implement and analyze search and sort algorithms
The design of class discussion or exercise, practice, experience and so on: Lecture and assignments
Grading & Evaluation: Attendance: 20% Homework assignments: 30% Midterm Examination: 20% Final Examination: 30%
Teaching Materials & References: Textbook: Fundamentals of Python: First Programs, 2nd Edition Author: Kenneth Lambert Publisher: Cengage Learning, 2017