UNIVERSITY OF GLOBAL VILLAGE

PYTHON PROGRAMMING

PART-1

COURSE PLAN

4th Semester

Department of CSE

CLO & OUTLINE DESIGN

Mohammad Rony
Lecturer
Department of CSE
University of Global Village

CLASS CONTENT DESIGN

Galib Jaman
Lab Instructor
Department of CSE
University of Global Village

DATA STRUCTURES

Course Learning Outcomes (CLO)

1. **Learn Python Basics:** Understand Python syntax and basic programming constructs.
2. **Understand Data Types & Variables**: Work with data types and variables and learn various use cases of data types and variables
3. **Understand Control Structures**: Implement decision-making structures like `if-else` and loops to control the flow of programs based on conditions.
4. **Working with Functions & Modules**: Write reusable code using functions, understand parameter passing.
5. **Learn to Handle Files & Exceptions**: Learn how to perform file operations such as reading, writing, and appending data.
6. **Learn the Steps of Problem Solving**: Apply logical thinking and structured approaches to design and implement solutions for real-world programming problems.
7. **Implement Data Structures & Recursion:**Explore fundamental data structures and understand the concept of recursion.
8. **Understand OOP Concepts with Python**: Understand object-oriented principles to build modular and scalable applications.
9. **Write Efficient and Readable Code:** Develop algorithms to solve problems efficiently, optimize code for performance, and apply best practices.

CLO MAPPING OF

17-Class Course Plan

WEEK	TOPIC	ASSESSMENT STRATEGY	CLO
01	Installing Python, exploring syntax	Practice, Quiz	01
02	Data Types and Variables	Practice, Review	01
03	Implementing Logic	Practice, Quiz	02
04	Loops and Iteration	Practice, Assignment	02
05	Writing and using functions	Practice, Quiz	03
06	Creating & Using modules	Review, Assignment	03
07	Reading and writing to files	Practice, Review	04
08	Exception Handling	Practice, Assignment	05
09	Structuring simple programs	Practice, Review	06
10	Combining loops & conditionals	Practice, Group Work	06
11	Non-Primitive Data Types	Practice, Group Work	07
12	Basics of recursion	Quiz	07
13	Object Oriented Programming	Quiz, Practice	08
14	Encapsulation, inheritance	Quiz	08
15	Basics of Problem Solving	Practice, Review	09
16	Writing efficient Python code	Quiz, Group Work	09
17	Evaluate concepts learned	Assignment, Group Work	09

WEEK-01 | CLO-01

Installing Python & Exploring syntax

Outcome:

- Successfully install Python and set up the development environment.
- Understand the basic syntax and structure of Python programs.

Discussion Topics:

- ☐ How to download and install Python on different operating systems.
- ☐ Introduction to Python & Interpreted Languages.
- ☐ Setting up an Integrated Development Environment.
- ☐ Writing and running your first Python script.
- ☐ Python syntax essentials like Indentation and comments.
- ☐ Introduction to print() and taking input using input().

Questions:

1. Explain the difference between compiled and interpreted programming languages.

Lab Practice:

1. Install Python on your machine and verify the installation using the command line (python --version).
2. Write a program that prints "Hello, Python World!" to the console.
3. Create a Python script that takes a user's name as input and displays a personalized greeting.
4. Write a program to demonstrate the use of single-line and multi-line comments.
5. Use Python's built-in help() function to explore any Python function (e.g., print or input).

WEEK-02 | CLO-01

Data Types and Variables

Outcome:

- Understand the role and usage of variables and data types in Python.
- Learn arithmetic operations through comprehensive practice.

Discussion Topics:

- ☐ Primitive and non-primitive data types in Python.
- ☐ What are variables, and why are they important in programming?
- ☐ Declaration, initialization, and dynamic typing of variables.
- ☐ Variable naming rules and conventions.
- ☐ The concept of mutability and immutability in Python data types.
- ☐ Arithmetic operators in Python and their usage.

Questions:

1. What is the difference between mutable and immutable data types?
2. How is variable declaration different in Python compared to statically-typed languages?

Lab Practice:

1. Write a program to declare variables of different data types and display their types using ***`type()`***.
2. Write a program to calculate the sum of two numbers.
3. Create a program to calculate the area of a rectangle using variables for length and width.
4. Write a program to compute the perimeter and area of a circle with a given radius.
5. Write a program to convert specified days into years, weeks and days.
6. Write a program to convert specified seconds into hours, minutes and seconds.

WEEK-03 | CLO-02

Implementing Logics

Outcome:

- Understand the concept of conditional statements and decision-making in C programming.
- Learn how to implement ***`if`***, ***`if-else`*** and ***`elif`*** statements.

Discussion Topics:

- ☐ What are conditional statements, and why are they important?
- ☐ Comparison operators in Python and their usage in conditionals.
- ☐ Syntax and usage of ***`if`***, ***`if-else`***, and ***`elif`*** statements.
- ☐ The ***`elif`*** ladder and its role in multi-condition scenarios.
- ☐ Nested conditional statements and the importance of proper indentation.

Lab Practice:

1. Write a program to accept two integers and check whether they are equal or not.
2. Write a program to check whether a given number is even or odd.
3. Write a program to check whether a given number is positive or negative.
4. Write a program to find whether a given year is a leap year or not.
5. Write a program to find the largest of three numbers.
6. Write a program to accept a coordinate point in an XY coordinate system and determine in which quadrant the coordinate point lies.
7. Write a program to check whether a triangle can be formed with the given values for the angles.
8. Write a program to check whether a character is an alphabet, digit or special character.
9. Write a program to accept a grade and declare the equivalent description using multiple conditional statements.

WEEK-04 | CLO-03

Loops and Iteration

Outcome:

- Understand the concept and purpose of loops in programming.
- Learn to use different types of loops to solve repetitive tasks efficiently.

Discussion Topics:

- ☐ What are loops, and why are they used in programming?
- ☐ Introduction to ***`range()`*** in Python and its usage.
- ☐ Types of loops in Python: for and while.
- ☐ Syntax and differences between ***`for`*** and ***`while`*** loops in Python.
- ☐ Usage and examples of nested loops in Python.
- ☐ The role of ***`break`*** and ***`continue`*** statements in controlling loop flow.

Lab Practice:

1. Write a Python program to print numbers from 1 to 10 using a for loop and a while loop.
2. Write a Python program to display the first n natural numbers and their sum.
3. Write a Python program to display the multiplication table for a given integer.
4. Write a Python program to check if a number is a prime number or not.
5. Write a Python program to reverse a given number using a loop.
6. Write a Python program to count the number of digits in a given integer using a loop.
7. Write a Python program to display the cube of numbers up to a given integer.
8. Write a Python program to calculate the factorial of a given number.

9. **Draw the following patterns using loops:**

```
*              1            a                *             1
* *            1 2          b c             * *           2 3
* * *          1 2 3        d e f          * * *         4 5 6
* * * *        1 2 3 4      g h i j       * * * *       7 8 9 10
```

WEEK-05 | CLO-03

Writing and using functions

Outcome:

- Understand the importance of functions in programming.
- Learn to define, call, and use functions effectively in Python.
- Explore parameter passing, default arguments, and return values.

Discussion Topics:

- ☐ What are functions, and why are they used in programming?
- ☐ Syntax of defining and calling a function in Python.
- ☐ Parameter types: Positional, default, and keyword arguments.
- ☐ Returning values from functions and the ***None*** return type.
- ☐ The concept of scope: Local and global variables in functions.

Lab Practice:

1. Write a function that takes two numbers as arguments and returns their sum.
2. Create a function to calculate the factorial of a number provided by the user.
3. Write a function that accepts a list of numbers and returns the maximum and minimum values.
4. Implement a program with a function that checks if a given number is prime.
5. Create a program to calculate the area of a rectangle using a function with two parameters (length and width).
6. Write a function to convert temperatures from Celsius to Fahrenheit and vice versa.
7. Implement a function that takes a string as input and returns the number of vowels and consonants.
8. Write a program that demonstrates the use of a function with default arguments.
9. Create a program to demonstrate the use of a function to calculate the nth Fibonacci number.