CH/AD/CD/CE/ CM/CS/EC/EE/ IT/ME/QC153	COMPUTATIONAL THINKING AND PROBLEM SOLVING WITH PYTHON LABORATORY Semester I [First Year]	L	T	P	C	Int	Ext
		-	-	3	1.5	40	60

COURSE OUTLINE:

* This course introduces first-year engineering students to computational thinking and its translation into working Python programs, establishing the problem-solving foundation that underlies all subsequent computing work.
* It develops the discipline of decomposing problems, recognizing patterns, abstracting details, and designing algorithms and flowcharts before writing code.
* Building on this foundation, the course covers Python control structures, core data structures, functions and recursion, file and exception handling, and a first exposure to data analysis and visualization through a capstone project.
* The course is common to all branches and feeds directly into higher-semester courses in data structures, object-oriented programming, and branch specific computational and data - analytics electives.

COURSE OBJECTIVES:

1. To introduce the fundamental pillars of computational thinking - decomposition, pattern recognition, abstraction, and algorithmic thinking - for systematic problem-solving.
2. To develop skills in designing algorithms, pseudocode, and flowcharts as precursors to programming.
3. To enable proficiency in Python programming using control structures and core data structures such as strings, lists, tuples, and dictionaries.
4. To build the ability to write modular, reusable Python code using functions, recursion, and built-in library modules.
5. To develop competence in file handling, exception handling, and basic data analysis and visualization using Python, culminating in an integrated capstone project.

COURSE OUTCOMES:

After the successful completion of the course,students are able to

CO	Course Outcome	RBTL
CO1	understand the pillars of computational thinking and apply them to decompose problems, design algorithms, and construct flowcharts.	Apply (BL3)
CO2	apply Python control structures - conditionals, loops, and loop-control statements - to implement solution logic for a given problem	Apply (BL3)
CO3	apply appropriate Python data structures - strings, lists, tuples, and dictionaries - to organize, access, and manipulate data.	Apply (BL3)
CO4	apply functions, recursion, and modular programming techniques, including built-in library modules, to build reusable Python code	Apply (BL3)
CO5	apply file handling, exception handling, and basic data-analysis tools in Python to complete an integrated end-to-end programming project.	Apply (BL3)

CO - PO / PSO Mapping

CO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PSO1	PSO2
CO1	3	2	1	1	2	-	-	-	-	-	1	-	-
CO2	3	3	2	1	3	-	-	1	-	-	1	-	-
CO3	3	2	2	1	3	-	-	1	-	-	1	-	-
CO4	3	2	2	1	3	-	-	1	1	-	2	-	-
CO5	2	3	2	2	3	1	-	1	2	-	2	-	-

EXPERIMENTS:

1. Draw a flowchart and write pseudocode to check whether a given number is even or odd.
2. Write a Python program to swap two numbers without using a third variable.
3. Design an algorithm and flowchart to convert temperature from Celsius to Fahrenheit
4. Write a Python program to check whether a given year is a leap year.
5. Write a program to print the multiplication table of a number using a for loop.
6. Write a program that uses break and continue to find the first prime number greater than a given input.
7. Write a Python program to count the frequency of each word in a sentence using a dictionary.
8. Write a program to find the second-largest number in a list without using built-in sort.
9. Write a program that uses a tuple to store and display the coordinates of a set of points.
10. Write a recursive Python function to generate the first n terms of the Fibonacci series.
11. Write a function with default arguments to compute the area of a rectangle or square.
12. Write a modular program that uses at least three separate functions to validate, process, and display user input.
13. Write a Python program that reads numbers from a file and handles exceptions for non-numeric entries.
14. Write a program using try-except-finally to validate user input for a simple calculator.
15. Using a small sample CSV dataset, write a program to compute column-wise averages with Pandas and plot the result with Matplotlib.

Note:**

A minimum of 10(Ten) experiments have to be performed and recorded by the candidate to attain eligibility for Semester End Practical Examination.