

- A) **Course Code** : **2418305(T2418305, P2418305,S2418305)**
- B) **Course Title** : Python Programming
(CE, CSE, AIML, ME, ME (Auto), ELX, ELX (R), MIE, FTS, CRE, CHE, TE, CACDDM, GT, RE)
- C) **Pre-requisite Course(s)** :
- D) **Rationale** :

Python programming has emerged as a popular programming language across wide range of application segments from Scientific to Machine Learning to mobile app development, and so on. Python is a high-level general-purpose programming language.

Because code is automatically compiled to byte code and executed, Python is suitable as a scripting language, Web application implementation language, etc.

In Python there are multiple levels of organizational structure: functions, classes, modules, and packages. These assist in organizing code. An excellent and large example is the Python standard library.

The Object-oriented Python provides a consistent way to use objects: in Python it is easy to implement new object types (called classes in object-oriented programming).

This introductory course to learn basic Python programming features which can be used as building blocks to develop different kind of applications using Python 3.

- E) **Course Outcomes (COs):** After the completion of the course, teachers are expected to ensure the accomplishment of following course outcomes by the learners. For this, the learners are expected to perform various activities related to three learning domains (Cognitive, Psychomotor and Affective) in classroom/ laboratory/ workshop/ field/ industry.

After completion of the course, the students will be able to-

- CO-1** Use various data types and operators in formation of expressions.
- CO-2** Write and execute programs using control statements.
- CO-3** Perform relevant operations on Sequence data types
- CO-4** Create functions in modules
- CO-5** Use numpy in writing python programs
- CO-6** Handle data files and exceptions.

- F) **Suggested Course Articulation Matrix (CAM):**

Course Outcomes (COs)	Programme Outcomes(POs)							Programme Specific Outcomes* (PSOs)	
	PO-1 Basic and Discipline Specific Knowledge	PO-2 Problem Analysis	PO-3 Design/ Development of Solutions	PO-4 Engineering Tools	PO-5 Engineering Practices for Society, Sustainability and Environment	PO-6 Project Management	PO-7 Life Long Learning	PSO-1	PSO-2
CO-1	1	-	1	-	-	-	-		
CO-2	1	2	2	1	-	1	-		
CO-3	1	2	2	1	-	1	-		
CO-4	1	2	2	1	-	1	2		
CO-5	1	2	2	1	-	1	-		
CO-6	1	2	2	1	-	1	1		

Legend: High (3), Medium (2), Low (1) and No mapping (-)

* PSOs will be developed by respective programme coordinator at institute level. As per latest NBA guidelines, formulating PSOs is optional

G) Teaching & Learning Scheme:

Board of Study	Course Code	Course Title	Scheme of Study (Hours/Week)					
			Classroom Instruction (CI)		Lab Instruction (LI)	Notional Hours (TW+ SL)	Total Hours (CI+LI+TW+SL)	Total Credits (C)
			L	T				
	2418305	Python programming	03	-	04	02	09	06

Note: Prefix will be added to Course Code if applicable (T for theory Paper, P for Practical Paper and S for Term work)

Legend:

CI: Classroom Instruction (Includes different instructional/implementation strategies i.e. Lecture (L), Tutorial (T), Case method, Demonstrations, Video demonstration, Problem based learning etc. to deliver theoretical concepts)

LI: Laboratory Instruction (Includes experiments/practical performances /problem-based experiences in laboratory, workshop, field or other locations using different instructional/Implementation strategies)

Notional Hours: Hours of engagement by learners, other than the contact hours for ensuring learning.

TW: Term Work (includes assignments, seminars, micro projects, industrial visits, any other student activities etc.)

SL: Self Learning, MOOCs, spoken tutorials, online educational resources etc.

C: Credits = (1 x CI hours) + (0.5 x LI hours) + (0.5 x Notional hours)

Note: TW and SL have to be planned by the teacher and performed by the learner under the continuous guidance and feedback of teacher to ensure outcome of learning.

H) Assessment Scheme:

Board of Study	Course Code	Course Title	Assessment Scheme (Marks)						Total Marks (TA+TWA+LA)
			Theory Assessment (TA)		Term Work & Self-Learning Assessment (TWA)		Lab Assessment (LA)		
			Progressive Theory Assessment (PTA)	End Theory Assessment (ETA)	Internal	External	Progressive Lab Assessment (PLA)	End Laboratory Assessment (ELA)	
	2418305	Python programming	30	70	20	30	20	30	200

Note: Prefix will be added to Course Code if applicable (T for theory Paper, P for Practical Paper and S for Term work)

Legend:

PTA: Progressive Theory Assessment in class room (includes class test, mid-term test and quiz using online/offline modes)

PLA: Progressive Laboratory Assessment (includes process and product assessment using rating Scales and rubrics)

TWA: Term work & Self Learning Assessment (Includes assessment related to student performance in assignments, seminars, micro projects, industrial visits, self-learning, any other student activities etc.)

Note:

- ETA & ELA are to be carried out at the end of the term/ semester.
- Term Work is to be done by the students under the guidance of internal faculty but its assessment will be done **internally (40%)** as well as **externally (60%)**. Assessment related to planning and execution of Term Work activities like assignment, micro project, seminar and self-learning is to be done by internal faculty (Internal Assessment) whereas assessment of output/product/ presentation related to these activities will be carried out by external faculty/expert (External Assessment). However, criteria of internal as well as external assessment may vary as per the requirement of respective course. For valid and reliable assessment, the internal faculty should prepare checklist & rubrics for these activities.

- I) **Course Curriculum Detailing:** This course curriculum detailing depicts learning outcomes at course level and session level and their attainment by the students through Classroom Instruction (CI), Laboratory Instruction (LI), Term Work (TW) and Self Learning (SL). Students are expected to demonstrate the attainment of Theory Session Outcomes (TSOs) and Lab Session Outcomes (LSOs) leading to attainment of Course Outcomes (COs) upon the completion of the course. While curriculum detailing, NEP 2020 related reforms like green skills, Sustainability, Multidisciplinary aspects, Society connect, Indian Knowledge System (IKS) and others must be integrated appropriately.

J) **Theory Session Outcomes (TSOs) and Units: T2418305**

Major Theory Session Outcomes (TSOs)	Units	Relevant COs Number(s)
<p><i>TSO 1a.</i> Differentiate between Procedure Oriented P and Object Oriented Programming approach with example.</p> <p><i>TSO 1b.</i> Use the concept of Lvalue and Rvalue</p> <p><i>TSO 1c.</i> Write python program using various data types and operators</p>	<p>Unit 1: Fundamentals of Python Programming Syntax</p> <p>1.1 Introduction to Python Character Set, Python Tokens, Variables, Lvalue and Rvalue Concepts, and the Use of Comments.</p> <p>1.2 Overview of Data Types:</p> <ul style="list-style-type: none"> • Number Types: Integer, Floating Point, Complex • Boolean Type • Sequence Types: String, List, Tuple • None Type • Mapping Type: Dictionary • Distinction between Mutable and Immutable Data Types <p>1.3 Understanding Operators:</p> <ul style="list-style-type: none"> • Arithmetic Operators • Relational Operators • Logical Operators • Assignment Operator • Augmented Assignment Operators • Expressions and Statements • Type Conversion and Input/Output Mechanisms • Precedence of Operators • Expression Evaluation 	CO-1
<p><i>TSO 2a.</i> Write Python program using decision making statements</p> <p><i>TSO 2b.</i> Write Python program using loop structure to solve iterative problems</p>	<p>Unit-2.0 Conditional and Iterative statements</p> <p>2.1 Conditional statements:</p> <ul style="list-style-type: none"> • simple if statement • if- else statemen • if-elif-else statement <p>2.2 Iterative statements:</p> <ul style="list-style-type: none"> • while loop • for loop • range function • break and continue statements • nested loops 	CO-2

Major Theory Session Outcomes (TSOs)	Units	Relevant COs Number(s)
<p><i>TSO 3a.</i> Perform various operations on string using string operators and methods</p> <p><i>TSO 3b.</i> Perform various operations on List using list operators and methods</p> <p><i>TSO 3c.</i> Perform various operations on tuples using tuples operators and methods</p> <p><i>TSO 3d.</i> Perform various operations on set using set methods</p> <p><i>TSO 3e.</i> Perform various operations on dictionary using dictionary methods</p>	<p>Unit-3.0 String, List, Tuples, set and Dictionary</p> <p>3.1 String:</p> <ul style="list-style-type: none"> Indexing string operations (concatenation, repetition, membership & slicing) traversing a string using loops built-in functions. <p>3.2 Lists:</p> <ul style="list-style-type: none"> Introduction Indexing in list list operations: concatenation, repetition, membership & slicing, traversing a list, built- in list functions, linear search on list of numbers and counting the frequency of elements in a list <p>3.3 Tuples: Creating, initializing, accessing elements, tuple assignment, performing operations on tuples, tuple methods and built-in functions, nested tuples</p> <p>3.4 Set: Creating set, traversing, adding, removing data in set, performing set operations like join, Union intersection, difference</p> <p>3.5 Dictionary: accessing items in a dictionary using keys, mutability of dictionary: adding a new item, modifying an existing item, built-in dictionary functions.</p>	CO-3
<p><i>TSO 4a.</i> Create and use user defined functions to implement modular programming approach</p> <p><i>TSO 4b.</i> Differentiate variable scope with example.</p> <p><i>TSO 4c.</i> Import and use Python modules, libraries</p>	<p>Unit-4.0 Python Functions, Modules and packages</p> <p>4.1 Functions: types of function (built- in functions, functions defined in module, user defined functions), creating user defined function, arguments and parameters, default parameters, positional parameters, Lambda functions, returning value, scope of a variable: global scope, local scope</p> <p>4.2 Modules and Packages: Importing module using 'import' Regular Expressions, Exception Handling, PyPI Python Package Index, Pip Python package manager, Importing Libraries and Functions</p>	CO-4
<p><i>TSO 5a.</i> Write simple Python programs using numpy</p> <p><i>TSO 5b.</i> Use Numpy array in python program</p> <p><i>TSO 5c.</i> Use Numpy to solve linear algebra problem.</p>	<p>Unit-5.0 Numpy</p> <p>5.1 Introduction to NumPy</p> <p>5.2 Installation of NumPy</p> <p>5.3 NumPy Arrays:</p> <ul style="list-style-type: none"> Understanding the NumPy array 	CO-5

Major Theory Session Outcomes (TSOs)	Units	Relevant COs Number(s)
	<ul style="list-style-type: none"> The fundamental data structure in NumPy. Creation of arrays using different methods: np.array(), np.zeros(), np.ones(), etc. Exploring array attributes like shape, size, and dimensions. <p>5.4 Array Indexing and Slicing:</p> <ul style="list-style-type: none"> Accessing elements and subarrays in NumPy arrays using indexing and slicing. Demonstration of the difference between one-dimensional and multi-dimensional array indexing. <p>5.5 Array Operations:</p> <ul style="list-style-type: none"> Performing element-wise operations on NumPy arrays. Exploring universal functions (ufuncs) for mathematical operations. <p>5.6 Linear Algebra with NumPy:</p> <ul style="list-style-type: none"> Introduction to linear algebra operations using NumPy. Matrix multiplication, determinant, inverse, and solving linear equations. <p>5.7 File input and output with Numpy</p> <p>5.8 Broadcasting in Numpy</p>	
<p><i>TSO 6a.</i> Explain different types of Exceptions in python</p> <p><i>TSO 6b.</i> Write Python programs for exception handling in Python</p> <p><i>TSO 6c.</i> Differentiate different modes of file opening.</p> <p><i>TSO 6d.</i> Perform read, Write, Append operations in files</p>	<p>Unit 6: Exception and File Handling in Python</p> <p>6.1 Exception Handling: syntax errors, exceptions, need of exception handling, user-defined exceptions, raising exceptions, handling exceptions, catching exceptions, Try - except - else clause, Try - finally clause, recovering and continuing with finally, built-in exception classes.</p> <p>6.2 File Handling: text file and binary file, file types, open and close files, reading and writing text files, reading and writing binary files, file access modes</p>	CO-6

Note: One major TSO may require more than one Theory session/Period.

K) Suggested Laboratory (Practical) Session Outcomes (LSOs) and List of Practical: P2418305

Practical/Lab Session Outcomes (LSOs)	S. No.	Laboratory Experiment/Practical Titles	Relevant COs Number(s)
<p><i>LSO 1.1.</i> Write, execute and debug simple Python program using Integrated Development and Learning Environment (IDLE)</p> <p><i>LSO 1.2.</i> Write and execute simple 'C' program using variables, arithmetic expressions.</p>	1.	<p>a) Download and Install IDLE.</p> <p>Write and execute Python program to-</p> <p>b) Calculate the Area of a Triangle where its three sides a, b, c are given. $s=(a+b+c)/2$, Area=square root of $s(s-a)(s-b)(s-c)$ (write program without using function)</p> <p>c) Swap Two Variables</p> <p>d) Solve quadratic equation for real numbers.</p>	CO-1
<p><i>LSO 2.1.</i> Write and execute python programs using conditional statements.</p> <p><i>LSO 2.2.</i> Write and execute python programs using various types of Loop statements</p>	2.	<p>Write and execute Python program to-</p> <p>a) Check if a Number is Positive, Negative or zero.</p> <p>b) Check whether the given year is a Leap Year.</p> <p>c) Print all Prime Numbers in an Interval.</p> <p>d) Display the multiplication Table based on the given input.</p> <p>e) Print the Fibonacci sequence.</p> <p>f) Find the Factorial of a Number.</p>	CO-2
<i>LSO 3.1.</i> Write and execute Python program to perform various operations on string using string operators and methods	3.	<p>Write and execute Python program to-</p> <p>a) Check whether the string is Palindrome</p> <p>b) Reverse words in a given String in Python</p> <p>c) identify in a strings the name, position and counting of vowels.</p> <p>d) Count the Number of matching characters in a pair of string (set)</p> <p>e) Python program for removing i-th character from a string</p>	CO-2, CO-3
<i>LSO 4.1.</i> Write and execute Python program to perform various operations on List using List operators and methods	4.	<p>Write and execute Python program to-</p> <p>a) find largest number in a given list without using max().</p> <p>b) find the common numbers from two lists.</p> <p>c) create a list of even numbers and another list of odd numbers from a given list.</p> <p>d) To find number of occurrences of given number without using built-in methods.</p>	CO-2, CO-3
<i>LSO 5.1.</i> Write and execute Python program to perform various operations on Tuple using Tuple operators and methods.	5.	<p>Write and execute Python program to-</p> <p>a) find the index of an item of a tuple.</p> <p>b) find the length of a tuple.</p> <p>c) to reverse a tuple.</p> <p>d) Write a Python program to sort a list of tuple by its float element. Sample data: [('item1', '12.20'), ('item2', '15.10'), ('item3', '24.5')] Expected Output: [('item3', '24.5'), ('item2', '15.10'), ('item1', '12.20')]</p>	CO-2, CO-3

Practical/Lab Session Outcomes (LSOs)	S. No.	Laboratory Experiment/Practical Titles	Relevant COs Number(s)
<i>LSO 6.1.</i> Write and execute Python program to perform various operations on sets using set methods.	6.	Write and execute Python program to- a) create an intersection of sets. b) create a union of sets. c) create set difference. d) check if two given sets have no elements in common.	CO-2, CO-3
<i>LSO 7.1.</i> Write and execute Python program to perform various operations on Dictionary using Dictionary methods	7.	Write and execute Python program to- a) Write a Python script to concatenate two dictionaries to create a new one b) Write a Python script to merge two Python dictionaries. c) Write a Python program to combine two dictionary adding values for common keys. d1 = {'a': 100, 'b': 200, 'c':300} d2 = {'a': 300, 'b': 200, 'd':400} Sample output: d({'a': 400, 'b': 400, 'd': 400, 'c': 300})	CO-2, CO-3
<i>LSO 8.1.</i> Write and execute Python program to create user defined functions and call them.	8.	Write and execute Python program to- a) Write a Python function for reversing a string and call it. b) Write a Python function for calculating compound interest and call it. c) Write a Python function for calculating the factorial of a number and call it to calculate $n!/(r)!(n-r)!$ where symbol " $!$ " stands for factorial.	CO-2, CO-4
<i>LSO 9.1.</i> Write and execute Python program to define a numpy array. <i>LSO 9.2.</i> Develop and execute Python program Using various types of Numpy operation.	9.	a) Write a python program to create a Numpy array filled with all zeros b) Write a python program to check whether a Numpy array contains a specified row c) Write a python program to Remove rows in Numpy array that contains non-numeric values d) Write a python program to Find the number of occurrences of a sequence in a NumPy array e) Write a python program to Find the most frequent value in a NumPy array f) Write a python program to Combine a one and a two-dimensional NumPy Array g) Write a python program to Flatten a Matrix in Python using NumPy h) Write a python program to Interchange two axes of an array	CO-2, CO-5
<i>LSO 10.1.</i> Develop and execute Python program to handle various type of exceptions. <i>LSO 10.2.</i> Develop and execute Python program to perform file operations.	10.	a) Using exception handling feature such as try...except, try finally- write minimum three programs to handle following types of exceptions. i. Type Error ii. Name Error	CO-6, CO-1, CO-2,

Practical/Lab Session Outcomes (LSOs)	S. No.	Laboratory Experiment/Practical Titles	Relevant COs Number(s)
		iii. Index Error iv. Key Error v. Value Error vi. IO Error vii. Zero Division Error b) Write Python program to demonstrate file operations.	

Note: in addition to above listed practical, students are suggested to practice all the examples covered by the teacher during theory sessions.

L) Suggested Term Work and Self Learning: S2418305 Some sample suggested assignments, micro project and other activities are mentioned here for reference.

a. **Assignments:** Questions/Problems/Numerical/Exercises to be provided by the course teacher in line with the targeted COs.

b. **Micro Projects:**

1. Create a shop billing system
2. Create income tax calculation system.
3. Develop number guessing game (random integer will be selected by the system and the user has to guess that integer in the minimum number of guesses. Maximum 5 guess allowed.)
4. Assign numbers to alphabet a-z as (1-26). User will input a word. System will convert it to a number by adding all the individual alphabet of that word.
5. Design a basic calculator program that performs arithmetic operations like addition, subtraction, multiplication, and division based on user input.
6. Any other micro-projects suggested by subject faculty on similar line.

(Students may use file and sequence data types to develop above listed applications)

c. **Other Activities:**

1. Seminar Topics:
 - Tkinter widgets in python
 - Python date/time module and its applications
 - wxPython and its applications

M) Suggested Course Evaluation Matrix: The course teacher has to decide and use appropriate assessment strategy and its weightage in theory, laboratory and Term Work for ensuring CO attainment. The response/performance of each student in each of these designed activities is to be used to calculate **CO attainment**.

COs	Course Evaluation Matrix						
	Theory Assessment (TA)**		Term Work Assessment (TWA)			Lab Assessment (LA)#	
	Progressive Theory Assessment (PTA) Class/Mid Sem Test	End Theory Assessment (ETA)	Term Work & Self Learning Assessment			Progressive Lab Assessment (PLA)	End Laboratory Assessment (ELA)
			Assignments	Micro Projects	Other Activities*		
CO-1	10%	10%	15%	16%	16%	10%	16%

CO-2	15%	15%	15%	16%	16%	15%	16%
CO-3	25%	25%	20%	18%	18%	25%	18%
CO-4	15%	15%	15%	16%	16%	15%	16%
CO-5	25%	25%	25%	18%	18%	25%	18%
CO-6	10%	10%	10%	16%	16%	10%	16%
Total Marks	30	70	20	20	10	20	30
			50				

Legend:

*: Other Activities include self- learning, seminar, visits, surveys, product development, software development etc.

** : Mentioned under point- (N)

: Mentioned under point-(O)

Note:

- The percentage given are approximate
- In case of Micro Projects and End Laboratory Assessment (ELA), the achieved marks will be equally divided in all those COs mapped with total experiments.
- For CO attainment calculation indirect assessment tools like course exit survey need to be used which comprises of questions related to achievement of each COs.

N) Suggested Specification Table for End Semester Theory Assessment: Specification table represents the reflection of sample representation of assessment of cognitive domain of full course.

Unit Title and Number	Total Classroom Instruction (CI) Hours	Relevant COs Number(s)	Total Marks	ETA (Marks)		
				Remember (R)	Understanding (U)	Application & above (A)
Unit-1.0 Basics of Python Programming syntax	4	CO-1	7	3	2	2
Unit-2.0 Conditional and Iterative statements	6	CO-2	10	3	3	4
Unit-3.0 3.0 String, List, Tuples, set and Dictionary	12	CO-3	18	5	3	10
Unit-4.0 Python Functions, Modules and packages	7	CO-4	10	3	3	4
Unit-5.0 Numpy	12	CO-5	18	4	5	9
Unit-6.0 Exception and File Handling in Python	7	CO-6	7	2	2	3
Total	48	-	70	20	18	32

Note: Similar table can also be used to design class/mid-term/ internal question paper for progressive assessment.

O) Suggested Assessment Table for Laboratory (Practical):

S. No.	Laboratory Practical Titles	Relevant COs Number(s)	PLA/ELA		
			Performance		Viva-Voce (%)
			PRA* (%)	PDA** (%)	
1.	Write and execute Python program to- a) Calculate the Area of a Triangle where its three sides a,b,c are given. $s=(a+b+c)/2$, Area=square root of $s(s-a)(s-b)(s-c)$ (write program without using function) b) Swap Two Variables c) Solve quadratic equation for real numbers.	CO-1	40	50	10
2.	Write and execute Python program to- a) Check if a Number is Positive, Negative or zero. b) Check whether the given year is a Leap Year.	CO-2	40	50	10

S. No.	Laboratory Practical Titles	Relevant COs Number(s)	PLA/ELA		
			Performance		Viva-Voce (%)
			PRA* (%)	PDA** (%)	
	c) Print all Prime Numbers in an Interval. d) Display the multiplication Table based on the given input. e) Print the Fibonacci sequence. f) Find the Factorial of a Number.				
3.	Write and execute Python program to- a) Check whether the string is Palindrome b) Reverse words in a given String in Python c) identify in a strings the name, position and counting of vowels. d) Count the Number of matching characters in a pair of string (set) e) Python program for removing i-th character from a string	CO-2, CO3	40	50	10
4.	Write and execute Python program to- a) find largest number in a given list without using max(). b) find the common numbers from two lists. c) create a list of even numbers and another list of odd numbers from a given list. d) To find number of occurrences of given number without using built-in methods.	CO-2, CO-3	40	50	10
5.	Write and execute Python program to- a) find the index of an item of a tuple. b) find the length of a tuple. c) to reverse a tuple. d) Write a Python program to sort a list of tuple by its float element. Sample data: [('item1', '12.20'), ('item2', '15.10'), ('item3', '24.5')] Expected Output: [('item3', '24.5'), ('item2', '15.10'), ('item1', '12.20')]	CO-2, CO-3	40	50	10
6.	Write and execute Python program to- a) create an intersection of sets. b) create a union of sets. c) create set difference. d) check if two given sets have no elements in common.	CO-2, CO-3	40	50	10
7.	Write and execute Python program to- a) Write a Python script to concatenate two dictionaries to create a new one b) Write a Python script to merge two Python dictionaries. c) Write a Python program to combine two dictionary adding values for common keys. d1 = {'a': 100, 'b': 200, 'c': 300} d2 = {'a': 300, 'b': 200, 'd': 400} Sample output: d({'a': 400, 'b': 400, 'd': 400, 'c': 300})	CO-2, CO-3	40	50	10
8.	Write and execute Python program to- a) Write a Python function for reversing a string and call it. b) Write a Python function for calculating compound interest and call it.	CO-2, CO-4	40	50	10

S. No.	Laboratory Practical Titles	Relevant COs Number(s)	PLA/ELA		
			Performance		Viva-Voce (%)
			PRA* (%)	PDA** (%)	
	c) Write a Python function for calculating the factorial of a number and call it to calculate $n/(n-r)!$ where symbol "!" stands for factorial.				
9.	a) Write a python program to create a Numpy array filled with all zeros b) Write a python program to check whether a Numpy array contains a specified row c) Write a python program to Remove rows in Numpy array that contains non-numeric values d) Write a python program to Find the number of occurrences of a sequence in a NumPy array e) Write a python program to Find the most frequent value in a NumPy array f) Write a python program to Combine a one and a two-dimensional NumPy Array g) Write a python program to Flatten a Matrix in Python using NumPy Write a python program to Interchange two axes of an array	CO-2, CO-5	40	50	10
h)	Using exception handling feature such as try...except, try finally-write minimum three programs to handle following types of exceptions. viii. TypeError ix. NameError x. IndexError xi. KeyError xii. ValueError xiii. IOError xiv. ZeroDivisionError	CO-2, CO-6	40	50	10
i)	Write and execute Python program to- a) Calculate the Area of a Triangle where its three sides a,b,c are given. $s=(a+b+c)/2$, Area=square root of $s(s-a)(s-b)(s-c)$ (write program without using function) b) Swap Two Variables c) Solve quadratic equation for real numbers.	CO-1	40	50	10

Legend:

PRA*: Process Assessment

PDA**: Product Assessment

Note: This table can be used for both end semester as well as progressive assessment of practical. Rubrics need to be prepared by the course teacher for each experiment/practical to assess the student performance.

P) Suggested Instructional/Implementation Strategies: Different Instructional/ Implementation Strategies may be appropriately selected, as per the requirement of the content/outcome. Some of them are Improved Lecture, Tutorial, Case Method, Group Discussion, Industrial visits, Industrial Training, Field Trips, Portfolio Based, Learning, Role Play, Live Demonstrations in Classrooms, Lab, Field Information and Communications Technology (ICT)Based Teaching Learning, Blended or flipped mode, Brainstorming, Expert Session, Video Clippings, Use of Open Educational Resources (OER), MOOCs etc.

Q) List of Major Laboratory Equipment, Tools and Software:

S. No.	Name of Equipment, Tools and Software	Broad Specifications	Relevant Experiment/Practical Number
1.	Computer system	Processor Intel Core i5, 4 GB RAM, 15 GB free disk space	All
2.	Integrated Development and Learning Environment (IDLE)	S/w to be downloaded for python 3.11.3 or higher	All

R) Suggested Learning Resources:**(a) Books:**

S. No.	Titles	Author(s)	Publisher and Edition with ISBN
1.	Introduction to Computing and Problem-Solving using Python	E. Balagurusamy	McGraw Hill Education (India) Pvt. Ltd. 1 st Edition /2016
2.	Learning Python Programming	Jeffrey Elkner, Allan B. Downey, Chris Meyers	Samurai Media Limited. 2016
3.	Python Programming	Ashok Namdev Kamthane and Amit Ashok Kamthane	McGraw Hill Education (India) Pvt. Ltd. 2020, 2 nd Edition
4.	Programming in Python	Dr. Pooja Sharma	BPB Publications 2017

(b) Online Educational Resources:

1. <https://docs.python.org/3/tutorial/>
2. <https://www.w3schools.com/python/>
3. <https://www.tutorialspoint.com/python/index.htm>

Note: Teachers are requested to check the creative commons license status/ financial implications of the suggested, online educational resources before use by the students.
