CST 362	PROGRAMMING IN PYTHON	Category	L	T	P	Credit	Year of Introduction
		PEC	2	1	0	3	2019

Preamble: The objective of the course is to equip the learners to develop multi-module software solutions for real world computational problems using Python. It encompasses the Python programming environment, syntax, data representations, intermediate level features, GUI programming, Object Oriented Programming and data processing. This course lays the foundation to develop modular software solutions including complex interactive applications, network applications, and data-driven intelligent applications.

Prerequisite: Basic knowledge in Computational Problem Solving, A course in any programming language.

Course Outcomes: After the completion of the course the student will be able to

CO1	Write, test and debug Python programs **(Cognitive Knowledge level: Apply)**
CO2	Illustrate uses of conditional (if, if-else and if-elif-else) and iterative (while and for) statements in Python programs. **(Cognitive Knowledge level: Apply)**
CO3	Develop programs by utilizing the Python programming constructs such as Lists, Tuples, Sets and Dictionaries. **(Cognitive Knowledge level: Apply)**
CO4	Develop graphical user interface for solutions using Python libraries. **(Cognitive Knowledge level: Apply)**
CO5	Implement Object Oriented programs with exception handling. **(Cognitive Knowledge level: Apply)**
CO6	Write programs in Python to process data stored in files by utilizing Numpy, Matplotlib, and Pandas. **(Cognitive Knowledge level: Apply)**

Mapping of course outcomes with program outcomes

	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12
CO1	✔	✔	✔		✔							✔
CO2	✔	✔	✔									✔
CO3	✔	✔	✔	✔	✔							✔
CO4	✔	✔	✔	✔	✔							✔
CO5	✔	✔	✔	✔	✔							✔
CO6	✔	✔	✔	✔	✔	✔						✔

Abstract POs defined by National Board of Accreditation

#PO	Broad PO	#PO	Broad PO
PO1	Engineering Knowledge	PO7	Environment and Sustainability
PO2	Problem Analysis	PO8	Ethics
PO3	Design/Development of solutions	PO9	Individual and team work
PO4	Conduct investigations of complex problems	PO10	Communication
PO5	Modern tool usage	PO11	Project Management and Finance
PO6	The Engineer and Society	PO12	Life long learning

Assessment Pattern

Bloom's Category	Test 1 (*Marks in percentage*)	Test 2 (*Marks in percentage*)	End Semester Examination Marks
Remember	20	20	20

Understand	35	35	35
Apply	45	45	45
Analyse			
Evaluate			
Create			

Mark Distribution

Total Marks	CIE Marks	ESE Marks	ESE Duration
150	50	100	3

Continuous Internal Evaluation Pattern:

Attendance   : 10 marks

Continuous Assessment Test   : 25 marks

Continuous Assessment Assignment : 15 marks

Internal Examination Pattern:

Each of the two internal examinations has to be conducted out of 50 marks. The first series test shall be preferably conducted after completing the first half of the syllabus and the second series test shall be preferably conducted after completing the remaining part of the syllabus. There will be two parts: Part A and Part B. Part A contains 5 questions (preferably, 2 questions each from the completed modules and 1 question from the partly completed module), having 3 marks for each question adding up to 15 marks for part A. Students should answer all questions from Part A. Part B contains 7 questions (preferably, 3 questions each from the completed modules and 1 question from the partly completed module), each with 7 marks. Out of the 7 questions, a student should answer any 5.

End Semester Examination Pattern:

There will be two parts; Part A and Part B. Part A contains 10 questions with 2 questions from each module, having 3 marks for each question. Students should answer all questions. Part B

contains 2 questions from each module of which a student should answer any one. Each question can have a maximum of 2 sub-divisions and carries 14 marks.

SYLLABUS

Module -1 (Programming Environment and Python Basics) (6 hours)

Getting started with Python programming – Interactive shell, IDLE, iPython Notebooks, Detecting and correcting syntax errors, How Python works. The software development process – A case study. Basic coding skills – strings, assignment, and comments, Numeric data types and character sets, Expressions, Using inbuilt functions and modules. Control statements – Iteration with for/while loop, Formatting text for output, A case study, Selection structure (if-else, switch-case), Conditional iteration with while, A case study, Testing control statements, Lazy evaluation.

Module -2 (Building Python Programs) (8 hours)

Strings and text files – Accessing characters, substrings, Data encryption, Strings and number system, String methods, Text files, A case study on text analysis. Design with Functions – Functions as Abstraction Mechanisms, Problem solving with top-down design, Design with recursive functions, Managing a program's namespace, Higher-Order Functions. Lists - Basic list Operations and functions, List of lists, Slicing, Searching and sorting list, List comprehension. Work with tuples. Sets. Work with dates and times, A case study with lists. Dictionaries - Dictionary functions, dictionary literals, adding and removing keys, accessing and replacing values, traversing dictionaries, reverse lookup. Case Study – Data Structure Selection.

Module -3 (Graphics) (7 hours)

Graphics – Terminal-based programs, Simple Graphics using Turtle, Operations, 2D Shapes, Colors and RGB Systems, A case study. Image Processing – Basic image processing with inbuilt functions. Graphical User Interfaces – Event-driven programming, Coding simple GUI-based programs : Windows, Labels, Displaying images, Input text entry, Popup dialog boxes, Command buttons, A case study.

Module -4 (Object Oriented Programming) (7 hours)

Design with classes - Objects and Classes, Methods, Instance variables, Constructor, Accessor and Mutator, Data-Modeling Examples, Structuring classes with inheritance and polymorphism. Abstract classes, Interfaces, Exceptions - Handle a single exception, handle multiple exceptions.

Module -5 (Data Processing) (9 hours)

The os and sys modules, NumPy - Basics, Creating arrays, Arithmetic, Slicing, Matrix Operations, Random numbers. Plotting and visualization. Matplotlib - Basic plot, Ticks, Labels, and Legends. Working with CSV files. – Pandas - Reading, Manipulating, and Processing Data. Introduction to Micro services using Flask.

Text Books:

1. Kenneth A Lambert., Fundamentals of Python : First Programs, 2/e, Cengage Publishing, 2016

2. Wes McKinney, Python for Data Analysis, 2/e, Shroff / O'Reilly Publishers, 2017

3. Flask: Building Python web services, Jack Stouffer, Shalabh Aggarwal, Gareth Dwyer, PACKT Publishing Limited, 2018

Reference Books:

1. Zed A Shaw, Learn Python 3 The Hard Way, Addison-Wesley, 2017

2. Allen B. Downey, Think Python: How to Think Like a Computer Scientist, 2/e, Schroff, 2016

3. Michael Urban and Joel Murach, Python Programming, Shroff/Murach, 2016

4. Charles Severance. Python for Informatics: Exploring Information,

Sample Course Level Assessment Questions

Course Outcome1(CO1):

1. What is type conversion? How is it done in Python?

2. Write a note on the Python editors.

Course Outcome 2(CO2):

1. Write a Python program which takes a positive integer **n** as input and finds the sum of cubes all positive even numbers less than or equal to the number.

2. What is printed when the below code is executed?

```
mysum = 0
   for i in range(5, 11, 2):
            mysum += i
            if mysum == 5:
                    break
            mysum += 1
   print(mysum)
```

What would be the output if 'break' is replaced with 'continue' in the above code fragment?

Course Outcome 3(CO3):

1. Given is a list of of words, *wordlist*, and a string, *name.* Write a Python function which takes *wordlist* and *name* as input and returns a tuple. The first element of the output tuple is the number of words in the *wordlist* which have *name* as a substring in it. The second element of

the tuple is a list showing the index at which the *name* occurs in each of the words of the *wordlist* and a 0 if it doesn't occur.

2. What is the value of L after you run the code below?

```
L = ["life", "answer", 42, 0]
for thing in L:
        if thing == 0:
          L[thing] = "universe"
        elif thing == 42:
            L[1] = "everything"
```

Course Outcome 4(CO4):

1. A bouncy program is defined as follows – The program computes and displays the total distance traveled by a ball, given three inputs—the initial height from which it is dropped, its bounciness index, and the number of bounces. Given the inputs write a GUI-based program to compute the total distance traveled.

2. Write a Python program to find the quadrant of a point, say (x,y).

Course Outcome 5(CO5):

1. Write a Python program to implement the addition, subtraction, and multiplication of complex numbers using classes. Use constructors to create objects. The input to the program consist of real and imaginary parts of the complex numbers.

2. Explain inheritance in Python using suitable examples.

Course Outcome 6(CO6):

1. Given a file "auto.csv" of automobile data with the fields *index, company, body-style, wheel-base, length, engine-type, num-of-cylinders, horsepower, average-mileage,* and *price,* write python code to

    1. Clean and Update the CSV file

    2. Print total cars of all companies

    3. Find the average mileage of all companies

    4. Find the highest priced car of all companies.

2. Given two matrices A and B, write a program to find the product of A and B^T.

Model Question Paper

QP CODE: PAGES:

Reg No:_____

Name:_____

APJ ABDUL KALAM TECHNOLOGICAL UNIVERSITY SIXTH SEMESTER B.TECH DEGREE EXAMINATION, MONTH & YEAR

Course Code: CST 362

Course name : PROGRAMMING IN PYTHON

Max Marks: 100   **Duration: 3 Hours**

PART-A

(Answer All Questions. Each question carries 3 marks)

1. Write a Python program to reverse a number and also find the sum of digits of the number. Prompt the user for input.

2. Explain the concept of scope and lifetime of variables in Python programming language, with a suitable example.

3. Illustrate format specifiers and escape sequences with examples.

4. Compare tuples, lists, and dictionaries with examples.

5. Describe the following dictionary methods with an example.
   i. *get( )*   ii. *Keys( )*   iii. *pop( )*   iv. *update( )*   v. *values( )*   vi. *items( )*

6. Differentiate the terminal-based and GUI-based programming in Python.

7. What is polymorphism? Give an example in the context of OOP in Python.

8. How is exception handling accomplished in Python programs?

9. Explain the **os** and **os.path** modules in Python with examples. Also, discuss the *walk( )* and *getcwd( )* methods of the **os** module.

10. What are the important characteristics of CSV file format.

PART-B

(Answer any one full question from each module)

11. (a) Write a Python code to check whether a given year is a leap year or not (6)
[An year is a leap year if it's divisible by 4 but not divisible by 100 except for those divisible by 400].

(b) What are the possible errors in a Python program. Write a Python (8)
program to print the value of $2^{2n}+n+5$ for n provided by the user.

OR

12. (a) Write a Python program to find the value for $sin(x)$ up to n terms using the (6)
series

$$\sin(x) = \frac{x}{1!} - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \cdots \qquad \text{where } x \text{ is in degrees}$$

(b) Write a Python code to determine whether the given string is a Palindrome (8)
or not using slicing. Do not use any string function.

13. (a) Write a Python code to create a function called *list_of_frequency* that takes a (5)
string and prints the letters in non-increasing order of the frequency of their occurrences. Use dictionaries.

(b) Write a Python program to read a list of numbers and sort the list in a non- (9)
decreasing order without using any built in functions. Separate function should be written to sort the list wherein the name of the list is passed as the parameter.

OR

14. (a) Illustrate the following Set methods with an example. (6)

i. *intersection( )* ii. *Union( )* iii. *Issubset( )* iv. *Difference( )* v. *update( )* vi. *discard( )*

(b) Write a Python program to check the validity of a password given by the (8)
user.

The Password should satisfy the following criteria:

1. Contains at least one letter between **a** and **z**

2. Contains at least one number between **0** and **9**

3. Contains at least one letter between **A** and **Z**

4. Contains at least one special character from **$, #, @**

5. Minimum length of password: **6**

15. (a) Write a program to draw a hexagon using turtle. (5)

    (b) Write a note on the image processing function in Python. (9)

OR

16. (a) Describe the features of event driven programming. (4)

    (b) Write a GUI-based program that allows the user to convert temperature values between degrees Fahrenheit and degrees Celsius. The interface should have labeled entry fields for these two values. These components should be arranged in a grid where the labels occupy the first row and the corresponding fields occupy the second row. At start-up, the Fahrenheit field should contain 32.0, and the Celsius field should contain 0.0. The third row in the window contains two command buttons, labeled >>>> and <<<<. When the user presses the first button, the program should use the data in the Fahrenheit field to compute the Celsius value, which should then be output to the Celsius field. The second button should perform the inverse function. (10)

17. (a) How can a class be instantiated in Python? Write a Python program to express the instances as return values to define a class RECTANGLE with parameters *height, width, corner_x,* and *corner_y* and member functions to find center, area, and perimeter of an instance. (10)

    (b) Explain inheritance in Python. Give examples for each type of inheritance. (4)

OR

18. (a) Write a Python class named ***Circle*** constructed by a radius and two methods which will compute the area and the perimeter of a given circle (6)

    (b) Write Python program to create a class called as **Complex** and implement *__add__( )* method to add two complex numbers. Display the result by overloading the + Operator. (8)

19. (a) Write a Python program to add two matrices and also find the transpose of the resultant matrix. (8)

    (b) Given a file "auto.csv" of automobile data with the fields *index, company, body-style, wheel-base, length, engine-type, num-of-cylinders, horsepower, average-mileage,* and *price,* write Python codes using Pandas to (6)

        1) Clean and Update the CSV file

        2) Print total cars of all companies

3) Find the average mileage of all companies

4) Find the highest priced car of all companies.

OR

20. (a) Write Python program to write the data given below to a CSV file. (5)

SN	Name	Country	Contribution	Year
1	Linus Torvalds	Finland	Linux Kernel	1991
2	Tim Berners-Lee	England	World Wide Web	1990
3	Guido van Rossum	Netherlands	Python	1991

(b) Given the sales information of a company as CSV file with the following (9)
fields *month_number, facecream, facewash, toothpaste, bathingsoap,
shampoo, moisturizer, total_units, total_profit*. Write Python codes to
visualize the data as follows

1) Toothpaste sales data of each month and show it using a scatter plot

2) Face cream and face wash product sales data and show it using the
bar chart

Calculate total sale data for last year for each product and show it using a
Pie chart.

(14X5=70)

Teaching Plan

Module 1: Programming Environment and Python Basics		(6 hours)
1.1	Getting started with Python programming – Interactive shell, IDLE, iPython Notebooks, Detecting and correcting syntax errors, How Python works.	**1 hour**
1.2	The software development process – A case study.	**1 hour**
1.3	Basic coding skills – strings, assignment, and comments, Numeric data types and character sets	**1 hour**
1.4	Expressions, Using inbuilt functions and modules.	**1 hour**
1.5	Control statements – Definite Iteration with for loop, Formatting text for output, Selection structure (if-else, switch-case), Conditional iteration with	**1 hour**

	while loop, A case study	
1.6	Testing the control statements, Lazy evaluation.	**1 hour**
Module 2: Building Python Programs		**(8 hours)**
2.1	Strings – Accessing characters, substrings, Data encryption, Strings and number system, String methods,	**1 hour**
2.2	Text files, A case study on text analysis.	**1 hour**
2.3	Design with Functions – Functions as Abstraction Mechanisms, Problem solving with top-down design,	**1 hour**
2.4	Design with recursive functions, Managing a program's namespace, Higher-Order Functions.	**1 hour**
2.5	Lists - Basic list Operations and functions, List of lists, Slicing, Searching and sorting list, List comprehension.	**1 hour**
2.6	Work with tuples. Sets. Work with dates and times, A case study with lists.	**1 hour**
2.7	Dictionaries - Dictionary functions, dictionary literals, adding and removing keys, accessing and replacing values, traversing dictionaries, reverse lookup.	**1 hour**
2.8	Case Study - Data Structure Selection.	**1 hour**
Module 3: Graphics		**(7 hours)**
3.1	Graphics – Simple Graphics using Turtle, Operations, 2D Shapes,	**1 hour**
3.2	Colors and RGB Systems, A case study.	**1 hour**
3.3	Image Processing – Basic image processing with inbuilt functions.	**1 hour**
3.4	Graphical User Interfaces – Event-driven programming	**1 hour**
3.5	Coding simple GUI-based programs : Windows, Labels, Displaying images,	**1 hour**
3.6	Coding simple GUI-based programs : Input text entry, Popup dialog boxes, Command buttons	**1 hour**
3.7	A case study - GUI	**1 hour**

Module 4: Object Oriented Programming		**(7 hours)**
4.1	Design with classes : Objects and Classes, Methods, Instance Variables	**1 hour**
4.2	Constructor, Accessors, and Mutators	**1 hour**
4.3	Structuring classes with Inheritance	**1 hour**
4.4	Polymorphism	**1 hour**
4.5	Abstract Classes	**1 hour**
4.6	Interfaces	**1 hour**
4.7	Exceptions : Handle a single exception, handle multiple exceptions	**1 hour**
Module 5: Data Processing		**(9 hours)**
5.1	The *os* and *sys* modules, NumPy : Basics, Creating arrays, Arithmetic, Slicing	**1 hour**
5.2	Matrix Operations, Random numbers.	**1 hour**
5.3	Matplotlib : Basic plot, Ticks, Labels, and Legends	**1 hour**
5.4	Working with CSV files	**1 hour**
5.5	Pandas : Reading, Manipulating	**1 hour**
5.6	Pandas : Processing Data and Visualize.	**1 hour**
5.7	Introduction to Microservices using Flask	**1 hour**
5.8	Introduction to Microservices using Flask	**1 hour**
5.9	Introduction to Microservices using Flask	**1 hour**