NOIDA INSTITUTE OF ENGINEERING AND TECHNOLOGY GREATER NOIDA-201306

(An Autonomous Institute)

School of Computer Science in Emerging Technologies

Subject Name: **Advanced Python**	**L-T-P [0-0-6]**
Subject Code: **BCSE0252**	**Applicable in Department: B. Tech.- Second Semester CSE/CSE-R/IT/CS/IOT/M.Tech.(Int.)**
Pre-requisite of Subject: Basic Python Programming & Python Concepts.	
Course Objective: To become familiar with Python's Object-Oriented Concepts, functional programming and create GUI application and to gain the knowledge of Python libraries.	

Course Outcomes (CO)

Course outcome: After completion of this course students will be able to:		**Bloom's Knowledge Level(KL)**
CO1	Implement classes and create instances in python	K3
CO 2	Implement GUI based Python application	K3
CO 3	Use Python libraries for data handling.	K3
CO 4	Analyse data using visualization libraries.	K4
CO 5	Analyse web scraping application for real world data	K4

Syllabus						
Unit No	**Module Name**	**Topic covered**	**Pedagogy**	**Lecture Required (L+P)**	**Practical/ Assignment/ Lab Nos**	**CO Mapping**
Unit 1	**Classes and Objects**	Introduction: Python Classes and objects, User- Defined Classes, Class Variables and Instance Variables, Instance methods, Class method, static methods, constructor in python, parametrized constructor, Magic Methods in python, Object as an argument, Instances as Return Values, namespaces, Introduction to inheritance and polymorphism, Abstract Class, Introduction to Abstraction and Encapsulation.	Smart board, Hands-on exercise	10+10	Program1-32	CO1
Unit 2	**Functionaland GUI Programming**	Functional Programming: Immutability, Closures and Decorators, generators, Co-routines, iterators, Declarative programming, GUI Programming: Introto GUI Programming, Settling widgets in the window's interior, Numeric Widgets, Boolean Widgets, Selection Widgets, String Widgets, Date Picker, Color Picker, Container Widgets, Creating a GUI Application, Tkinter, button, canvas.	Smart board, Hands-on exercise	4+10	Program33-78	CO2
Unit 3	**Libraries for Data Handling**	NumPy: Basic Operation, Indexing, slicing and Iterating, Multidimensional arrays, NumPy Data types, Reading and writing data on Files	Smart board, Hands-on exercise	5+8	Program79-116	CO3

		SciPy: Introduction to SciPy, Create function,modules of SciPy. Pandas: Series and Data Frames, Grouping, aggregation, Merge Data Frames, Generate summary tables, Group data into logical pieces, Manipulation of data				
Unit 4	**Libraries in Data Visualization**	Matplotlib: Scatter plot, Bar charts, histogram, Stack charts, Legend title Style, Figures andsubplots, Plotting function in pandas, Labelling and arranging figures, Save plots. Seaborn: style function, color palettes, heatmaps,distribution plots, category plot, regressionplotPlotly: Line plots, Area plots, Scatterplots, Bubble plots, Stacked bar charts, Grouped bar charts, Pie charts, Tables, Dashboards.	Smart board, Hands-on exercise	5+8	Program117-174	CO4
Unit 5	**Web Scraping with Python**	WebScraping: Introduction, Web Crawling v/s WebScraping, Uses of Web Scraping, Components of a Web Scraper, working of a Web Scraper, Crawl, Parse and Transform Store the Data. BeautifulSoup: Introduction to Beautiful Soup library, Accessing Tags, Navigable Strings, Navigating and searching with Beautiful Soup, Web Scraping. Example: Scraping Flipkart Website Introduction to	Smart board, Hands-on exercise	4+8	Program175-208	CO5

		GitHub.				
Total				**72**		

Lab Practical		
Course Objective: To enhance students' proficiency in advanced Python features, including object-oriented programming,functional programming, python packages and libraries for data science, web development.		
Course Outcomes (CO)		
Course outcome: After completion of this course students will be able to:		**Bloom's Knowledge Level(KL)**
CO1	Apply classes and instances in real world problems.	K3
CO2	Implement GUI based Python application	K3
CO3	Design python packages, libraries and web scraping application	K6
List of Practical		
Sr No	**Program Title**	**CO Mapping**
1.	Write a program illustrating class definition and accessing class members.	CO1
2.	Write a program to implement default constructor, parameterized constructor, and destructor.	CO 1

3.	Create a Python class named Rectangle constructed by a length and width. a. Create a methodcalled area which will compute the area of a rectangle.	CO 1
4.	Create a class called Numbers, which has a single class attribute called MULTIPLIER, and aconstructor which takes the parameters x and y (these should all be numbers). a. Write an instance method called add which returns the sum of the attributes x and y. b. Write a class method called multiply, which takes a single number parameter a and returns theproduct of a and MULTIPLIER.	CO 1
5.	Create a class named as Student to store the name and marks in three subjects. Use List to storethe marks. a. Write an instance method called compute to compute total marks and average marks of astudent. b. Write a method called display to display student information.	CO 1
6.	Create a Python class named Circle constructed by a radius. Use a class variable to define the valueof constant PI. a. Write two methods to be named as area and circum to compute the area and the perimeter ofa circle respectively by using class variable PI. b. Write a method called display to print area and perimeter.	CO 1

7.	Write a program that has a class called Fraction with attributes numerator and denominator. a. Write a method called getdata to enter the values of the attributes. b. Write a method show to print the fraction in simplified form.	CO 1
8.	Write a program that has a class Numbers with a list as an instance variable. a. Write a method called insert_element that takes values from user. b. Write a class method called find_max to find and print largest value in the list.	CO 1
9.	Create a class called Complex. Write a menu driven program to read, display, add and subtract twocomplex numbers by creating corresponding instance methods.	CO 1
10.	Write a program that has a class Point with attributes x and y. a. Write a method called midpoint that returns a midpoint of a line joining two points. b. Write a method called length that returns the length of a line joining two points.	
11.	Write a Python program to create a class called "Rectangle" with attributes length and width.Include methods to calculate the perimeter and area of the rectangle.	CO 1
12.	Implement a Python class called "BankAccount" with attributes account number, account holder	CO 1

	name, and balance. Include methods to deposit and withdraw money from the account.	
13.	Write a Python program to create a class called "Student" with attributes roll number, name, and marks in three subjects. Include a method to calculate the average marks of the student.	CO 1
14.	Implement a Python class called "Car" with attributes make, model, and year. Include methods to start the car, stop the car, and display its details.	CO 1
15.	Write a Python program to create a class called "Book" with attributes title, author, and price. Include methods to calculate the discounted price of the book based on a discount percentageprovided.	CO 1
16.	Implement a Python class called "Bank" with attributes bank name and branch. Include methods to add a new account, display all accounts, and search for an account based on the accountnumber.	CO 1
17.	Write a Python program to create a class called "Rectangle" with attributes length and width. Include a method to check if the rectangle is a square or not.	CO 1
18.	Implement a Python class called "Employee" with attributes name, designation, and experience. Include methods to promote an employee to a higher designation based on their experience.	CO 1
19.	Write a Python program to create a class called "Employee" with attributes name, employee ID, and salary. Include a method to display the employee details.	CO 1
20.		CO 1

	Write a program to illustrate the use of following built-in methods: a. hasattr(obj,attr) b. getattr(object, attribute_name [, default]) c. setattr(object, name, value) d. delattr(class_name, name)	
21.	Write a Program to illustrate the use of__str__(),__repr__(),__new__,__doc__,__dict__, __name__and__bases__methods.	CO 1
22.	Write a program to create class Employee. Display the personal information and salary details of 5employees using single inheritance.	CO 1
23.	WAP that extends the class Employee. Derive two classes Manager and Team Leader from Employee class. Display all the details of the employee working under a particular Manager and Team Leader.	CO 1
24.	Write a program that has a class Point. Define another class Location which has two objects (Location and destination) of class Point. Also, define a function in Location that prints the reflection on the y-axis.	CO 1
25.	Write a program that create a class Distance with members km and metres. Derive classes Schooland office which store the distance from your house to school and office along with other details.	CO 1

26.	Write a program to create an abstract class Vehicle. Derive three classes Car, Motorcycle and Truck from it. Define appropriate methods and print the details of vehicle	CO 1
27.	Write a program to demonstrate hybrid inheritance and show MRO for each class.	CO 1
28.	Write a program to overload + operator to multiply to fraction object of fraction class which contain two instance variable numerator and denominator. Also, define the instance method simplify() to simplify the fraction objects.	CO 1
29.	26. Write a program to compare two-person object based on their age by overloading > operator. .	CO 1
30.	Write a program to overload in operator.	CO 1
31.	WAP to create a Complex class having real and imaginary as it attributes. Overload the +,-,/,* and += operators for objects of Complex class	CO 1
32.	Design a fundamental banking system where users can create accounts, deposit money, withdraw money, and check their balance.	CO1
33.	WAP to Show the concept of inner function.	CO2
34.	WAP to create closure.	CO2

35.	WAP to create a decorator which will convert a string into upper case string.	CO2
36.	WAP to show the concept of nested decorator.	CO2
37.	WAP to decorate a function with arguments.	CO2
38.	WAP to decorate instance method	CO2
39.	WAP to calculate sum of 1,2,3,4,5 using reduce function.	CO2
40.	WAP to generate numbers from 1 to 10 using generator.	CO2
41.	WAP to decide number is even or odd using generator.	CO2
42.	WAP to generate square of 1,2,3,4,5,6,7,8,9,10 using generator.	CO2
43.	WAP to generate square of even number upto 10 using generator and save in list.	CO2

44.	WAP to make a co-routine which will print all name with prefix Dear.	CO2
45.	WAP to close a co-routine.	CO2
46.	WAP to iterate tuple using iter() and next() method.	CO2
47.	WAP to iterate a string using iter and next method.	CO2
48.	WAP to print numbers from 1 to 20 using iterator and generate StopIteration exception once wereach limit.	CO2
49.	Hello World: Display a simple "Hello, World!" message box.	CO 2
50.	Button: Create a button that displays a message when clicked.	CO 2
51.	Entry: Create a text entry field and display the entered text.	CO 2
52.	Check button: Create a checkbox and display the selected options	CO 2

53.	Radio button: Create radio buttons and display the selected option.	CO 2
54.	List box: Create a list box and display the selected items.	CO 2
55.	Text: Create a text area and display the entered text.	CO 2
56.	Menu: Create a menu with different options.	CO 2
57.	Message: Display a message in a dialog box.	CO 2
58.	Progress bar: Create a progress bar that updates over time python	CO 2
59.	Scale: Create a scale widget and display the selected value.	CO 2
60.	Spin box: Create a spin box and display the selected value.	CO 2
61.	Canvas: Create a canvas and draw shapes on it.	CO 2
62.	Label Frame: Create a labeled frame with widgets inside.	CO 2
63.	Scrollbar: Add a scrollbar to a widget like a text area or list box	CO 2

64.	Frame: Create a frame and place widgets inside it.	CO 2
65.	Tree view: Create a tree view widget to display hierarchical data	CO 2
66.	Notebook: Create a notebook widget with tabs.	CO 2
67.	File Dialog: Open a file dialog to select a file.	CO 2
68.	Color Dialog: Open a color dialog to select a color.	CO 2
69.	Button Counter: Create a button that increments a counter when clicked.	CO 2
70.	Checkbox List: Display a list of checkboxes and show selected options.	CO 2
71.	Dropdown Menu: Create a dropdown menu with multiple options.	CO 2
72.	Slider Value Display: Display the current value of a slider widget.	CO 2
73.	Text Input and Button: Take user input in a text box and display it when a button is clicked.	CO 2
74.	Radio Buttons: Present a set of options as radio buttons and display the selected option.	CO 2
75.	Progress Bar: Show the progress of a task using a progress bar widget.	CO 2
76.	Password Input: Create a password input field that hides the entered characters.	CO 2
77.	File Uploader: Enable users to upload files and display the selected file name.	CO 2
78.	Implement a Student class where students can enroll in courses and the class keeps track of total enrolments.	CO2
79.	Creating Arrays: Create NumPy arrays using various methods like np.array(), np.zeros(), np.ones(),	CO 3

	np.arange(), etc.	
80.	Array Shape and Size: Get the shape and size of a NumPy array using the shape and size attributes.	CO 3
81.	Array Indexing: Access and modify individual elements of a NumPy array using indexing	CO 3
82.	Array Slicing: Extract a subset of elements from a NumPy array using slicing.	CO 3
83.	Array Reshaping: Change the shape of a NumPy array using the reshape() function.	CO 3
84.	Array Arithmetic: Perform basic arithmetic operations (addition, subtraction, multiplication, division) on NumPy arrays.	CO 3
85.	Array Broadcasting: Perform element-wise operations on arrays with different shapes usingbroadcasting rules.	CO 3
86.	Array Aggregation: Calculate aggregate values on arrays, such as sum(), min(), max(), mean(), etc. using NumPy	CO 3
87.	Array Transposition: Transpose a NumPy array using the transpose() function.	CO 3
88.	Write a program that demonstrates advanced array indexing techniques, such as indexing with boolean arrays or using fancy indexing to select specific elements or subsets of an array.	CO3
89.	Write a program using NumPy to perform data manipulation tasks, such as sorting arrays, removing duplicates, or finding unique elements in an array.	CO3
90.	Array Sorting: Sort the elements of a NumPy array using the sort() function.	CO 3
91.	Array Filtering: Filter elements in a NumPy array based on a condition using boolean indexing.	CO 3

92.	Array Statistics: Calculate statistical measures like mean, median, standard deviation using functions like np.mean(), np.median(), np.std().	CO 3
93.	Array Randomization: Generate random numbers or arrays using functions from the np.random module.	CO 3
94.	Array Dot Product: Compute the dot product of two NumPy arrays using the dot() function.	CO 3
95.	Array Matrix Operations: Perform matrix operations like matrix multiplication, matrix inverse using functions from the np.linalg module.	CO 3
96.	Array File I/O: Save and load NumPy arrays from files using functions like np.save() and np.load().	CO 3
97.	Array Masking: Create a mask array to select or manipulate specific elements of a NumPy array based on a condition.	CO 3
98.	Array Broadcasting: Understand and utilize broadcasting rules in NumPy for efficient computations.	CO 3
99.	Write a program to finds the cube root of values using scipy library.	CO 3
100.	Write a program to computes the 10**x element-wise using scipy library .	CO 3
101.	Write a SciPy program to calculate Permutations and Combinations.	CO 3
102.	Write a SciPy program to calculates the inverse of any square matrix.	CO 3
103.	Write a SciPy program to calculates the Eigenvalues and Eigenvector.	CO 3
104.	Read and Load a CSV File into a Pandas DataFrame using pandas.read_csv.	CO 3
105.	Access and Display the First N Rows of a DataFrame using DataFrame.head(N).	CO 3

106.	Access and Display the Last N Rows of a DataFrame using DataFrame.tail(N).	CO 3
107.	Retrieve Basic Information about a DataFrame using DataFrame.info.	CO 3
108.	Perform Descriptive Statistics on a DataFrame using DataFrame.describe.	CO 3
109.	Filter Rows of a DataFrame based on a Condition using Boolean Indexing.	CO 3
110.	Rename Columns in a DataFrame using DataFrame.rename.	CO 3
111.	Group Data in a DataFrame using DataFrame.groupby.	CO 3
112.	Perform Aggregation on Grouped Data using GroupBy.agg.	CO 3
113.	Sort a DataFrame by One or Multiple Columns using DataFrame.sort_values.	CO 3
114.	Perform Basic Arithmetic Operations on Columns of a DataFrame.	CO 3
115.	Apply a Function to Each Element or Column of a DataFrame using DataFrame.apply or DataFrame.applymap.	CO 3
116.	Reshape Data using Pivot Tables using DataFrame.pivot_table.	CO 3
117.	Perform Data Visualization using pandas.plotting or matplotlib.pyplot.	CO 3
118.	Save a DataFrame to a CSV File using DataFrame.to_csv.	CO 3
119.	Perform Data Sampling or Random Selection using DataFrame.sample.	CO 3
120.	Find the roots of a mathematical equation using SciPy's root-finding functions, such as scipy.optimize.root.	CO 3
121.	Fit a polynomial function to a set of data points using SciPy's curve fitting functions, such as	CO 3

	scipy.optimize.curve_fit	
122.	Perform linear regression on a dataset using SciPy's linear regression functions, such as scipy.stats.linregress.	CO 3
123.	Calculate the Fast Fourier Transform (FFT) of a signal using SciPy's FFT functions, such as scipy.fft.fft.	CO 3
124.	Solve a system of linear equations using SciPy's linear algebra functions, such as scipy.linalg.solve.	CO 3
125.	Perform numerical integration using SciPy's integration functions such as scipy.integrate.quad.	CO 3
126.	Calculate the eigenvalues and eigenvectors of a square matrix using SciPy's linear algebra functions, such as scipy.linalg.eig.	CO 3
127.	Load a CSV file of the temperature readings and analyze the data using grouping, aggregation, and merging data frames.	CO3
128.	Create a Simple Line Plot using matplotlib.pyplot.plot.	CO 4
129.	Create a Scatter Plot using matplotlib.pyplot.scatter.	CO 4
130.	Create a Bar Chart using matplotlib.pyplot.bar.	CO 4
131.	Create a Histogram using matplotlib.pyplot.hist.	CO 4
132.	Create a Pie Chart using matplotlib.pyplot.pie.	CO 4
133.	Create a Box Plot using matplotlib.pyplot.boxplot.	CO 4
134.	Create a Heatmap using matplotlib.pyplot.imshow.	CO 4
135.	Customize Plot Labels and Titles using matplotlib.pyplot.xlabel, matplotlib.pyplot.ylabel, and	CO 4

	matplotlib.pyplot.title.	
136.	Customize Plot Colors, Line Styles, and Marker Styles using matplotlib.pyplot.plot parameters.	CO 4
137.	Add Gridlines to a Plot using matplotlib.pyplot.grid.	CO 4
138.	Add Legends to a Plot using matplotlib.pyplot.legend.	CO 4
139.	Create Subplots using matplotlib.pyplot.subplots.	CO 4
140.	Save a Plot as an Image File using matplotlib.pyplot.savefig.	CO 4
141.	Create 3D Plots using mpl_toolkits.mplot3d module.	CO 4
142.	Create Error Bars on a Plot using matplotlib.pyplot.errorbar.	CO 4
143.	Customize Axis Ticks and Tick Labels using matplotlib.pyplot.xticks and matplotlib.pyplot.yticks.	CO 4
144.	Create a Bar Plot with Stacked Bars using matplotlib.pyplot.bar and the bottom parameter.	CO 4
145.	Create a Scatter Plot using seaborn.scatterplot.	CO 4
146.	Create a Line Plot using seaborn.lineplot.	CO 4
147.	Create a Bar Plot using seaborn.barplot.	CO 4
148.	Create a Histogram using seaborn.histplot.	CO 4
149.	Create a Box Plot using seaborn.boxplot.	CO 4
150.	Create a Violin Plot using seaborn.violinplot.	CO 4
151.	Create a Heatmap using seaborn.heatmap.	CO 4

152.	Create a Pair Plot using seaborn.pairplot.	CO 4
153.	Create a Joint Distribution Plot using seaborn.jointplot.	CO 4
154.	Create a KDE (Kernel Density Estimate) Plot using seaborn.kdeplot.	CO 4
155.	Create a Categorical Scatter Plot using seaborn.stripplot.	CO 4
156.	Create a Categorical Bar Plot using seaborn.countplot.	CO 4
157.	Create a Facet Grid using seaborn.FacetGrid.	CO 4
158.	Customize Plot Colors and Styles using seaborn.set_palette and seaborn.set_style.	CO 4
159.	Add Error Bars to a Plot using seaborn.barplot or seaborn.pointplot with the ci parameter.	CO 4
160.	Create a Clustered Heatmap using seaborn.clustermap.	CO 4
161.	Create a Regression Plot using seaborn.regplot.	CO 4
162.	Create a Pairwise Relationship Plot using seaborn.pairplot or seaborn.scatterplot with multiple variables.	CO 4
163.	Create a Boxen Plot using seaborn.boxenplot.	CO 4
164.	Create a Stacked Bar Plot using seaborn.barplot with the hue parameter.	CO 4
165.	Write a program to draw a line chart using Plotly	CO 4
166.	Write a program to draw a Bar chart using Plotly	CO 4
167.	Write a program to draw a Histogram chart using Plotly	CO 4

168.	Write a program to draw a scatter plot using Plotly	CO 4
169.	Write a program to draw a Bubble chart using Plotly	CO 4
170.	Write a program to draw a pie chart using Plotly	CO 4
171.	Write a program to draw a Boxplot using Plotly	CO 4
172.	Write a program to draw Violin Plots using Plotly	CO 4
173.	Write a program to draw a Gant chart using Plotly	CO 4
174.	As a data analyst working on a project to analyse sales data for a retail company. Your task is tovisualize various aspects of the data using Matplotlib, Seaborn, and Plotly to gain insights and communicate findings effectively.	CO4
175.	Write a Python program to find the title tags from a given html document.	CO 5
176.	Write a Python program to retrieve all the paragraph tags from a given html document.	CO 5
177.	Write a Python program to get the number of paragraph tags of a given html document.	CO 5
178.	Write a Python program to extract the text in the first paragraph tag of a given html document.	CO 5
179.	Write a Python program to find the length of the text of the first <h2> tag of a given html document.	CO 5
180.	Write a Python program to find the text of the first <a> tag of a given html text.	CO 5
181.	Write a Python program to find the href of the first <a> tag of a given html document.	CO 5
182.	Write a Python program to a list of all the h1, h2, h3 tags from the webpage python.org.	CO 5

183.	Write a Python program to extract all the text from a given web page.	CO 5
184.	Write a Python program to print the names of all HTML tags of a given web page going through the document tree.	CO 5
185.	Write a Python program to retrieve children of the html tag from a given web page.	CO 5
186.	Write a Python program to retrieve all descendants of the body tag from a given web page.	CO 5
187.	Write a Python program to print content of elements that contain a specified string of a given web page.	CO 5
188.	Write a Python program to print the element(s) that has a specified id of a given web page.	CO 5
189.	Write a Python program to create a Beautiful Soup parse tree into a nicely formatted Unicode string, with a separate line for each HTML/XML tag and string.	CO 5
190.	Write a Python program to find the first tag with a given attribute value in an html document.	CO 5
191.	Write a Python program to find tag(s) beneath other tag(s) in a given html document.	CO 5
192.	Write a Python program to find tag(s) directly beneath other tag(s) in a given html document.	CO 5
193.	Write a Python program to find the siblings of tags in a given html document.	CO 5
194.	Write a Python program to find tags by CSS class in a given html document.	CO 5
195.	Write a Python program to change the tag's contents and replace with the given string.	CO 5
196.	Write a Python program to add to a tag's contents in a given html document.	CO 5
197.	Write a Python program to insert a new text within a url in a specified position.	CO 5

198.	Write a Python program to insert tags or strings immediately before specified tags or strings.	CO 5
199.	Write a Python program to insert tags or strings immediately after specified tags or strings.	CO 5
200.	Write a Python program to remove the contents of a tag in a given html document.	CO 5
201.	Write a Python program to extract a tag or string from a given tree of html document.	CO 5
202.	Write a Python program to remove a tag from a given tree of html document and destroy it and its contents.	CO 5
203.	Write a Python program to remove a tag or string from a given tree of html document and replace it with the given tag or string.	CO 5
204.	Write a Python program to wrap an element in the specified tag and create the new wrapper.	CO 5
205.	Write a Python program to replace a given tag with whatever's inside a given tag.	CO 5
206.	As a data analyst working for a retail company. Your task is to gather product information from various e-commerce websites to analyze market trends and competitor pricing.	CO5
207.	Write a program illustrating class definition and accessing class members.	CO 1
208.	Write a program to implement default constructor, parameterized constructor, and destructor.	CO 1

Required Software and Tools

1. Anaconda (Jupiter Notebook)
2. Python Compiler (Open Source)
3. Google Co-Lab

Textbooks	
Sr No	**Book Details**
1	Magnus Lie Hetland, "Beginning Python-From Novice to Professional"—Third Edition, Apress
2	Peter Morgan, Data Analysis from Scratch with Python, AI Sciences
3	Allen B. Downey, “Think Python: How to Think Like a Computer Scientist,” 2nd edition, Updated for Python 3,Shroff/O‘Reilly Publishers, 2016
4	Miguel Grinberg, Developing Web applications with Python, OREILLY
Reference Books	
Sr No	**Book Details**
1	Dusty Phillips, Python 3 Object-oriented Programming - Second Edition, O’Reilly
2	Burkhard Meier, Python GUI Programming Cookbook - Third ,Packt
3	DOUG HELLMANN, THE PYTHON 3 STANDARD LIBRARY BY EXAMPLE, : Pyth 3 Stan Libr Exam _2 (Developer's Library),1st Edition, Kindle Edition.
4	Kenneth A. Lambert, ―Fundamentals of Python: First Programs,‖ CENGAGE Learning, 2012.

Links (Only Verified links should be pasted here)
UNIT 1: https://nptel.ac.in/courses/106/106/106106145/ https://www.youtube.com/watch?v=_vr5faCXFo8 **UNIT 2:** https://realpython.com/python-gui-tkinter/ https://realpython.com/courses/functional-programming-python/ **UNIT 3:** https://www.youtube.com/watch?v=5rNu16O3YNE https://www.youtube.com/watch?v=8Y0qQEh7dJg **UNIT 4:** https://www.youtube.com/watch?v=OZOOLe2imFo https://www.youtube.com/watch?v=6GUZXDef2U0 **UNIT 5:** https://www.youtube.com/watch?v=8dTpNajxaH0 https://www.youtube.com/watch?v=4tAp9Lu0eDI