

Lab	Type	Practical
Python Setup and First Program with Jupyter notebook		
1	A	<ol style="list-style-type: none"> 1. Install latest version of Python 2. Install Anaconda to use Jupyter Notebook 3. Run a basic Python program that prints “Hello World” with all different possible ways: <ul style="list-style-type: none"> • Python IDLE (Interactive Mode + Script Mode) • Jupyter Notebook
Perform basic mathematical formula-based programs.		
2	A	<ol style="list-style-type: none"> 1. WAP to print “Hello World.!” 2. WAP to print addition of two numbers with and without using input(). 3. WAP to check the type of the variable. 4. WAP to calculate simple interest. 5. WAP to calculate area and perimeter of a circle. 6. WAP to calculate area of a triangle. 7. WAP to compute quotient and remainder. 8. WAP to convert degree into Fahrenheit and vice versa. 9. WAP to find the distance between two points in 2-D space. 10. WAP to print sum of n natural numbers. 11. WAP to print sum of square of n natural numbers. 12. WAP to concate the first and last name of the student. 13. WAP to swap two numbers. 14. WAP to get the distance from user into kilometer, and convert it into meter, feet, inches and centimeter. 15. WAP to get day, month and year from the user and print the date in the given format: 20-10-2024.
Perform programs based on conditional statements.		
3	A	<ol style="list-style-type: none"> 1. WAP to check whether the given number is positive or negative. 2. WAP to check whether the given number is odd or even. 3. WAP to find out largest number from given two numbers using simple if and ternary operator. 4. WAP to find out largest number from given three numbers. 5. WAP to check whether the given year is leap year or not. 6. WAP to display the name of the day according to the number given by the user.

	A	7. WAP to implement simple calculator which performs (addition, subtraction, multiplication and division) of two numbers based on user input.
	B	8. WAP to read marks of five subjects. Calculate percentage and print class accordingly. Fail below 35, Pass Class between 36 to 45, Second Class between 46 to 60, First Class between 61 to 70, Distinction if more than 70.
	B	9. Three sides of a triangle are entered through the keyboard, WAP to check whether the triangle is isosceles, equilateral, scalene or right-angled triangle.
	C	10. WAP to find the second largest number among three user input numbers.
	C	11. WAP to calculate electricity bill based on following criteria. Which takes the unit from the user. i. First 1 to 50 units – Rs. 2.60/unit ii. Next 50 to 100 units – Rs. 3.25/unit iii. Next 100 to 200 units – Rs. 5.26/unita iv. bove 200 units – Rs. 8.45/unit

Perform programs based on looping statements.

4	A	1. WAP to print 1 to 10.
	A	2. WAP to print 1 to n.
	A	3. WAP to print odd numbers between 1 to n.
	A	4. WAP to print numbers between two given numbers which is divisible by 2 but not divisible by 3.
	A	5. WAP to print sum of 1 to n numbers.
	A	6. WAP to print sum of series $1 + 4 + 9 + 16 + 25 + 36 + \dots + n$.
	A	7. WAP to print multiplication table of given number.
	A	8. WAP to print sum of series $1 - 2 + 3 - 4 + 5 - 6 + 7 \dots + n$.
	A	9. WAP to find factorial of the given number.
	B	10. WAP to find whether the given number is prime or not.
	B	11. WAP to find factors of a given number.
	B	12. WAP to print sum of digits of given number.
	C	13. WAP to find GCD of the given two numbers.
	C	14. WAP to check whether the given number is palindrome or not.

Perform programs to manipulate string in Python.

5	A	1. WAP to check given string is palindrome or not.
	A	2. WAP to reverse the words in given string.
	A	3. WAP to remove i^{th} character from given string.
	A	4. WAP to find length of String without using len() function.
	B	5. WAP to print even length word in string.
	B	6. WAP to count numbers of vowels in given string.
	C	7. WAP to capitalize the first and last character of each word in a string.
	C	8. WAP to convert given array to string.

Perform various operation on Python List.		
6	A	1. WAP to find sum of all the elements in a list. 2. WAP to find the largest element in a list. 3. WAP to find the length of a list. 4. WAP to interchange first and last elements in a list. 5. WAP to split the list into two parts and append the first part to the end. 6. WAP to interchange the elements on two positions entered by a user. 7. WAP to reverse a list given by user. 8. WAP to print even numbers in a list. 9. WAP to count unique items in a list. 10. WAP to copy a list. 11. WAP to print all odd numbers in a given range. 12. WAP to count occurrences of an element in a list. 13. WAP to find second largest number in a list. 14. WAP to extract elements with frequency greater than K.
Perform various operation on Python Tuple.		
Perform various operation on Python Set & Dictionary.		
7	A	1. WAP to find sum of tuple elements. 2. WAP to find Maximum and Minimum K elements in a given tuple. 3. WAP to find tuples which have all elements divisible by K from a list of tuples. 4. WAP to create a list of tuples from given list having number and its cube in each tuple. 5. WAP to find tuples with all positive elements from the given list of tuples. 6. WAP to add tuple to list and vice – versa. 7. WAP to remove tuples of length K. 8. WAP to remove duplicates from tuple. 9. WAP to multiply adjacent elements of a tuple and print that resultant tuple. 10. WAP to test if the given tuple is distinct or not.
8	A	1. WAP to iterate over a set. 2. WAP to convert set into list, string and tuple. 3. WAP to find Maximum and Minimum from a set. 4. WAP to perform union of two sets. 5. WAP to check if two lists have at-least one element common. 6. WAP to sort dictionary by key or value. 7. WAP to find the sum of all items in a dictionary given by user. 8. WAP to check if a given string is binary string or not. 9. WAP to find common elements in three lists using set.

	B	10. WAP to merge two dictionaries given by user.
	C	11. WAP to count number of vowels in given string using set.
	C	12. WAP to handle missing keys in dictionaries.

Build Python Functions to perform given functionalities.

9	A	1. WAP to count simple interest using function.
	A	2. WAP that defines a function to add first n numbers.
	A	3. WAP to find maximum number from given two numbers using function.
	A	4. WAP that defines a function which returns 1 if the number is prime otherwise return 0.
	B	5. WAP to generate Fibonacci series of N given number using function name fibbo. (e.g. 0 1 1 2 3 5 8...)
	B	6. WAP to find the factorial of a given number using recursion.
	C	7. WAP to implement simple calculator using lambda function.

Perform various operation on File.

10	A	1. WAP to read entire file named abc.txt.
	A	2. WAP to print program itself on console.
	A	3. WAP to read first 5 lines from the file named abc.txt.
	B	4. WAP to find the longest word in a file named abc.txt.
	B	5. WAP to find the size of the file named abc.txt.
	C	6. WAP to implement search function to search specific occurrence of word in a given text file.

Perform programs to understand Exception Handling.

11	A	1. WAP to handle Divide By Zero Exception.
	A	2. WAP to handle File Not Found Exception.
	A	3. WAP to handle Type Exception.
	B	4. WAP to demonstrate ValueError and IndexError with example.
	C	5. WAP to raise your custom Exception.

Perform programs on Python Data Structures and File Handling.

12	A	1. WAP to store student names in a list. Use list methods to add names, insert a new name at a specific position, remove a name, sort the list, reverse it, and display the final result.
	A	2. WAP to merge two lists and sort the combined list and then find how many times a particular value appears.
	A	3. WAP to insert a given value at every even index of a list and display the final list.

12	B	<p>4. Unique Word Analyzer</p> <p>Write a Python program that reads a text paragraph from a file named article.txt.</p> <ul style="list-style-type: none"> • Extract all words and store them in a list. • Convert the list to a set to determine the unique words. • Create a dictionary where each key = word, and value = frequency. • Store the frequency dictionary in a new file wordcount.txt.
	C	<p>5. Customer Purchase Summary</p> <p>Write a Python program to read customer purchase details from a file named purchases.txt.</p> <p>Each line of the file will have this information:</p> <p>CustomerID, CustomerName, Amount1, Amount2, Amount3, Amount4</p> <ul style="list-style-type: none"> • Read each line from the file and store the information in a list of tuples. • Take the four purchase amounts and put them into a set, so duplicate amounts are removed. • Find the average of these unique amounts. • Create a dictionary where: <ul style="list-style-type: none"> ◦ The key is the CustomerID ◦ The value is the average amount • Save this dictionary into a new file called summary.txt. • Use exception handling wherever it is required.

Perform programs to use Python Modules.

13	A	1. WAP to pick a random character from a given string.
	A	2. WAP to pick a random element from a given list.
	A	3. WAP to demonstrate the use of the math module.
	B	4. WAP to demonstrate the use of date time module.
	C	5. WAP to create Calculator module which defines functions like add, sub, mul and div. Create another file that uses the Calculator module.

Perform programs of generating different Graphs.

14	A	1. WAP to demonstrate the use of Pie chart.
	A	2. WAP to plot List random of X, Y Coordinates in Matplotlib.
	A	3. WAP to demonstrate the use of Bar chart.
	A	4. WAP to demonstrate the use of Histogram.
	B	5. WAP to display the value of each bar in a bar chart using Matplotlib.
	B	6. WAP create a Scatter Plot with several colours in Matplotlib.
	C	7. WAP to demonstrate the use of Box Plot.

Perform Programs of Object-Oriented Programming concept.		
15	A	1. Write a Python program to create a class named Students that initializes attributes like name, age, and grade using a constructor, and displays the details using a method. 2. Create a class Bank_Account with data members Account_No, User_Name, Email, Account_Type, and Account_Balance. Include methods GetAccountDetails() and DisplayAccountDetails(). 3. Create a class Circle with methods to calculate area and perimeter. 4. Define a class Time with hour and minute as data members and define a method to add two-time objects. 5. Demonstrate single inheritance using base class Person and derived class Student. 6. Write a Python program to demonstrate Multiple Inheritance where a class inherits properties and methods from more than one parent class. 7. Write a Python program to implement Multilevel Inheritance where a class is derived from another derived class. 8. Write a Python program to demonstrate Constructor Overloading using Default Arguments. 9. Write a Python program to implement Method Overloading using variable-length arguments. 10. Write a Python program to demonstrate Method Overriding in inheritance. 11. Write a Python program to demonstrate Operator Overloading using special methods. 12. Write a Python program to demonstrate Encapsulation using private data members. 13. Write a Python program to demonstrate Abstraction using the abc (Abstract Base Class) module. 14. Write a Python program to illustrate Polymorphism through method overriding and dynamic method dispatch. 15. Write a Python program to demonstrate the use of Static Methods and Class Methods.
	A	
	A	
	A	
	A	
	A	
	B	
	B	
	C	
	B	
	B	
	B	
	B	
	C	
	C	