Diploma Engineering

Laboratory Manual

Scripting Language - Python (4330701)

Semester-3 of Diploma in Computer Engineering

Enrolment No	
Name	
Branch	
Academic Term	
Institute	

Directorate of Technical Education

Gandhinagar - Gujarat

DTE’s Mission:

- To provide globally competitive technical education;
- Remove geographical imbalances and inconsistencies;
- Develop student friendly resources with a special focus on girls’ education and support to weaker sections;
- Develop programs relevant to industry and create a vibrant pool of technical professionals.

Institute’s Vision:

Institute’s Mission:

Department’s Vision:

Department’s Mission:

Certificate

This is to certify that Mr./Ms. …..…………………………………………………………….

Enrolment No. …………….…………..………………. of Semester ………….

of Diploma in *Computer Engineering* of Institute …………………………………………….…………………

……………………………………………………..….. (GTU code: …………..)

has satisfactorily completed the term work in course *Scripting Language - Python (4330701)* for the academic year: ………...……………….… Term: Odd/Even as prescribed in the GTU curriculum.

Place: ……………………….….

Date: ………………………….

Subject Faculty **Head of the Department**

Preface

The primary aim of any laboratory/Practical/field work is enhancement of required skills as well as creative ability amongst students to solve real time problems by developing relevant competencies in psychomotor domain. Keeping in view, GTU has designed competency focused outcome-based curriculum - 2021 (COGC-2021) for Diploma engineering programmes. In this more time is allotted to practical work than theory. It shows importance of enhancement of skills amongst students and it pays attention to utilize every second of time allotted for practical amongst Students, Instructors and Lecturers to achieve relevant outcomes by performing rather than writing practice in study type. It is essential for effective implementation of competency focused outcome- based Green curriculum-2021. Every practical has been keenly designed to serve as a tool to develop & enhance relevant industry needed competency in each and every student. These psychomotor skills are very difficult to develop through traditional chalk and board content delivery method in the classroom. Accordingly, this lab manual has been designed to focus on the industry defined relevant outcomes, rather than old practice of conducting practical to prove concept and theory.

By using this lab manual, students can read procedure one day in advance to actual performance day of practical experiment which generates interest and also, they can have idea of judgement of magnitude prior to performance. This in turn enhances predetermined outcomes amongst students. Each and every Experiment /Practical in this manual begins by competency, industry relevant skills, course outcomes as well as practical outcomes which serve as a key role for doing the practical. The students will also have a clear idea of safety and necessary precautions to be taken while performing experiment.

This manual also provides guidelines to lecturers to facilitate student-centered lab activities for each practical/experiment by arranging and managing necessary resources in order that the students follow the procedures with required safety and necessary precautions to achieve outcomes. It also gives an idea that how students will be assessed by providing Rubrics.

Python is an open-source, high-level, general-purpose programming language used for software development. It is one of the most popular programming languages in the world today and known for its simplicity as well as rich library. It is widely used programming language in various domains, such as Automation, Server-side Web Development, Tools Development, Game Programming, Blockchain, Data Science, Artificial Intelligence, Machine Learning, Big Data etc. It’s relatively easy to learn to use and incredibly versatile. This lab manual will help students to practice development of solution for different problems using basic building blocks of Python programming language. Students will be able to use their Python programming skills to develop simple applications in various domains mentioned above.

Although we try our level best to design this lab manual, but always there are chances of improvement. We welcome any suggestions for improvement.

Programme Outcomes (POs):

Following programme outcomes are expected to be achieved through the practical of the course:

1. **Basic and Discipline specific knowledge:** Apply knowledge of basic mathematics, science and engineering fundamentals and engineering specialization to solve the engineering problems.
2. **Problem analysis**: Identify and analyse well-defined e*ngineering* problems using codified standard methods.
3. **Design/development of solutions:** Design solutions for e*ngineering* well-defined technical problems and assist with the design of systems components or processes to meet specified needs.
4. **Engineering Tools, Experimentation and Testing:** Apply modern en*gineering* tools and appropriate technique to conduct standard tests and measurements.
5. **Engineering practices for society, sustainability and environment:** Apply appropriate technology in context of society, sustainability, environment and ethical practices.
6. **Project Management:** Use engineering management principles individually, as a team member or a leader to manage projects and effectively communicate about well-defined engineering activities.
7. **Life-long learning:** Ability to analyse individual needs and engage in updating in the context of technological changes *in field of engineering.*

Practical Outcome - Course Outcome matrix

Course Outcomes (COs):

a. *CO1: Develop programs to solve the given simple computational problems.*

b. *CO2: Apply control flow structures to solve the given problems.*

c. *CO3: Implement data structures lists, tuples, sets and dictionaries to solve the given problems.*

d. *CO4: Apply modular programming approach to solve given problems using user-defined functions.*

e. *CO5: Perform string manipulation and file operations to solve a given problem.*

Sr. No.	Experiment/Practical Outcome	CO1	CO2	CO3	CO4	CO5
1	**Environment Setup** i. Install and configure the Python environment. Run basic Python commands to verify the Python environment.	√	-	-	-	-
2	**Input-Output** i. Write a program to read your name, contact number, email, and birthdate and print those details on the screen.	√	-	-	-	-
3	**Variables, Operators and Expressions** i. Write a program to convert temperature from Celsius to Fahrenheit. Equation to convert Celsius to Fahrenheit: $F = (9/5) * C + 32$ ii. Write a program to compute the slope of a line between two points (x1, y1) and (x2, y2). $Slope = \frac{y_2 - y_1}{x_2 - x_1}$ iii. Write a program to calculate simple and compound interest. $Simple\ Interest = \frac{P * R * T}{100}$ $Compound\ Interest = P * \left(1 + \frac{R}{100 * n}\right)^{n * T}$ iv. Write a program to get change values in Quarter, Dime, Nickels and Pennies, and calculate the value of change in Dollars. Consider Quarter = 0.25 $, Dime = 0.10 $, Nickels = 0.05 $ and Penny = 0.01 $. v. Write a program to find a maximum of given three numbers (Use ternary operator).	√	-	-	-	-

	vi. Write a program to calculate area and volume of Sphere. $$Area\ of\ Sphere\ =\ 4\,\pi\,r^2$$ $$Volume\ of\ Sphere\ =\ \frac{4}{3}\,\pi\,r^3$$ vii. Write a program that computes the real roots of a given quadratic equation (Use math library). $$Discriminant\ \Delta\ =\ b^2\ -\ 4\,a\,c$$ $$Real\ Roots\ =\ \frac{-b\pm\sqrt{\Delta}}{2\,a}$$ viii. Write a program to determine the length of ladder required to reach a given height when leaned against the house. The height and the angle of the ladder are given as inputs (Use math Library).					
4	<u>**Decision Making Structures**</u> i. A year is a Leap year if it is divisible by 4, unless it is a century year that is not divisible by 400 (1800 and 1900 are not leap years, 1600 and 2000 are leap years). Write a program that calculates whether a given year is a leap year or not. ii. Many companies pay time-and-a-half for any hours worked above 40 hours in a given week. Write a program to input the number of hours worked and hourly rate and calculate the total wages for the week. iii. The Body Mass Index (BMI) is calculated as a person's weight (in kg), divided by the square of the person's height (in meters). If the BMI is between 19 and 25, the person is healthy. If the BMI is below 19, then the person is underweight. If the BMI is above 25, then the person is overweight. Write a program to get a person's weight (in kgs) and height (in cms) and display a message whether the person is healthy, underweight or overweight. $$BMI\ =\ \frac{Weight\ in\ kg}{(Height\ in\ m)^2}$$ iv. Write a program to read the marks and assign a grade to a student. Grading system: A (>=90), B (80-89), C (70-79), D (60-69), E (50-59), F (<50). (Use if-elif-else)	-	√	-	-	-
5	<u>**Loops**</u> i. Write a program to read n numbers from users and calculate the average of those n numbers.	-	√	-	-	-

	ii. Write a program that prompts the user to enter 10 integers and displays all the combinations of picking two numbers from the 10. iii. Write programs to print below patterns: * * * * * * * * * * * * * * * 1 1 2 1 2 3 1 2 3 4 1 2 3 4 5 iv. Write a program that displays an ASCII character table from ! to ~. Display the ASCII value of a character in decimal and hexadecimal. Display five characters per line. v. Write a program to sum the following series: $\frac{1}{3}+\frac{3}{5}+\frac{5}{7}+\frac{7}{9}+\frac{9}{11}+\frac{11}{13}+\dots+\frac{95}{97}+\frac{97}{99}$ vi. A positive integer is called a perfect number if it is equal to the sum of all of its positive divisors, excluding itself. For example, 6 is the first perfect number, because 6 = 3 + 2 + 1, the next is 28 = 14 + 7 + 4 + 2 + 1. There are four perfect numbers that are less than 10,000. Write a program to find these four numbers.					
6	<u>**List**</u> i. Write a program to perform the below operations on the list: ● Create a list. ● Add/Remove an item to/from a list. ● Get the number of elements in the list. ● Access elements of the list using the index. ● Sort the list. ● Reverse the list. ii. Write a program to read n numbers from a user and print: ● Number of positive numbers. ● Number of negative numbers. ● Number of zeros. ● Number of odd numbers. ● Number of even numbers. ● Average of all numbers. iii. Write a program that counts the occurrences of each digit in a string. The program counts how many times a digit appears in the string. For example, if the input is "12203AB3", then the	-	-	√	-	-

	output should output 0 (1 time), 1 (1 time), 2 (2 times), 3 (2 times). iv. Write a program to eliminate duplicate values in the list. v. Write a program to randomly fill in 0s and 1s into a 4x4 2-dimension list, print the list and find the rows and columns with the most number of 1s.					
7	**<u>Tuples, Sets and Dictionaries</u>** i. Write a program to perform below operations on tuple: ● Create a tuple with different data types. ● Print tuple items. ● Convert tuple into a list. ● Remove data items from a list. ● Convert list into a tuple. ● Print tuple items. ii. Write a program to perform below operations on set: ● Create two different sets with the data. ● Print set items. ● Add/remove items in/from a set. ● Perform operations on sets: union, intersection, difference, symmetric difference, check subset of another set. iii. Write a program to perform below operations on dictionary: ● Create a dictionary. ● Print dictionary items. ● Add/remove key-value pair in/from a dictionary. ● Check whether a key exists in a dictionary. ● Iterate through a dictionary. ● Concatenate multiple dictionaries. iv. Write a program that is given a dictionary containing the average daily temperature for each day of the week, and prints all the days on which the average temperature was between 40 and 50 degrees. v. Write a program to repeatedly prompt the user to enter the capital of a state. Upon receiving the user's input, the program reports whether the answer is correct. Assume the states and their capitals are stored in dictionaries as key-value pairs.	-	-	√	-	-
8	**<u>Functions</u>** i. Write a program that defines a function (shuffle) to scramble a list into a random order, like shuffling a deck of cards.	-	-	-	√	-

	ii. Write a program that defines a function to return a new list by eliminating the duplicate values in the list. iii. Write a program to print Fibonacci sequence up to n numbers using recursion. Fibonacci sequence is defined as below: $Fibonacci\ Sequence = 1\ 1\ 2\ 3\ 5\ 8\ 13\ 21 \ldots$ $where\ n^{th} term\ x_n = x_{n-1} + x_{n-2}$ iv. Write a program that defines a function to determine whether input number n is prime or not. A positive whole number n > 2 is prime, if no number between 2 and $\sqrt{n}$ (inclusive) evenly divides n. If n is not prime, the program should quit as soon as it finds a value that evenly divides n. v. Write a program that defines a function to find the GCD of two numbers using the algorithm below. The greatest common divisor (GCD) of two values can be computed using Euclid's algorithm. Starting with the values m and n, we repeatedly apply the formula: n, m = m, n%m until m is 0. At that point, n is the GCD of the original m and n (Use Recursion). vi. Write a program that lets the user enter the loan amount, number of years, and interest rate, and defines a function to calculate monthly EMI, total payment and display the amortization schedule for the loan.					
9	**Modules** i. Write a program that defines functions (mean and deviation), that computes mean and standard deviation of given numbers. The formula for the mean and standard deviation of n numbers is given as: $mean = \sum_{i=1}^{n} x_i = \frac{x_1 + x_2 + \ldots + x_n}{n}$ $deviation = \sqrt{\frac{\sum_{i=1}^{n} (x_i - mean)^2}{n-1}}$ ii. Write a program that plays the popular scissor-rock-paper game. (A scissor can cut a paper, a rock can knock a scissor, and a paper can wrap a rock.) The program randomly generates a number 0, 1, or 2 representing scissor, rock, and	-	-	-	√	-

	paper. The program prompts the user to enter a number 0, 1, or 2 and displays a message indicating whether the user or the computer wins, loses, or draws. iii. Write a program to print the dates of all the Sundays in a given year. iv. Write a program to display a graph for ReLU (Rectified Linear Unit) function. ReLU function is defined as below: $y = max\,(0,\ x)$ Consider the range of *x* from -5 to 5. v. Write a program to create a list representing the results of 100 students in a test, where each element represents a student's marks (between 0 to 10), and display a histogram for the result. vi. Create a user defined module with simple functions for: addition, subtraction, multiplication, division, modulo, square, factorial. Write a program to import the module and access functions defined in the module.					
10	**<u>String Processing</u>** i. Write a program to check whether a given string is palindrome or not. ii. Write a program to read a string containing letters, each of which may be in either uppercase or lowercase, and return a tuple containing the number of vowels and consonants in the string. iii. Write a program to read a date in the format DD/MM/YYYY and print the same date in MM-DD-YYYY format. iv. Write a program that checks whether two words are anagrams. v. Two words are anagrams if they contain the same letters. For example, *silent* and *listen* are anagrams. vi. Write a program that allows users to enter six-digit RGB color codes and converts them into base 10. In this format, the first two hexadecimal digits represent the amount of red, the second two the amount of green, and the last two the amount of blue. For example: If a user enters FF6347, then the output should be Red (255), Green (99) and Blue (71). vii. Numerologists claim to be able to determine a person's character traits based on the "numeric value" of a name. The value of a name is	-	-	-	-	√

	determined by summing up the values of the letters of the name, where "a" is 1 "b" is 2 "c" is 3 and so on up to "z" being 26. For example, the name "Python" would have the value 16 + 25 + 20 + 8 + 15 + 14 = 98. Write a program that calculates the numeric value of a name provided as input.					
11	**File Handling** i. Write a program to perform the below operations on files: ● Create a text file and write a string to it. ● Read an entire text file. ● Read a text file line by line. ● Write a string to a file. ● Write a list of strings to a file. ● Count the number of lines, words in a file. ii. Write a program that reads a text file and counts the occurrences of each alphabet in the file. The program should prompt the user to enter the filename. iii. Write a program that reads a text file and displays all the numbers found in the file. iv. Write an automated censor program that reads the text from a file and creates a new file where all of the four-letter words have been replaced by “****”. You can ignore punctuation, and you may assume that no words in the file are split across multiple lines. v. Write a program that reads a text file and calculates the average word length and sentence length in that file. vi. Write a program that reads two strings stored in two different text files and prints a string containing the characters of each string interleaved. Remove white spaces from both strings before string interleaving. For example, Two strings “Hello World” and “Sky is the Limit” should generate output “HSeklyliosWtohrelLdimit”.	-	-	-	-	√

Industry Relevant Skills

The following industry relevant skills of the competency "**Develop simple applications using scripting language Python.**" are expected to be developed in the student by undertaking the practical of this laboratory manual.

1. Install and configure software as per requirements.
2. Write code for the given problem.
3. Debug program to fix errors.
4. Follow Coding Guidelines.

Guidelines to Teachers

1. Couse faculty should demonstrate experiment with all necessary implementation strategies described in curriculum.
2. Couse faculty should explain industrial relevance before starting of each experiment.
3. Course faculty should involve & give opportunity to all students for hands on experience.
4. Course faculty should ensure mentioned skills are developed in the students by asking.
5. Utilise 2 hrs of lab hours effectively and ensure completion of write up with quiz also.
6. Encourage peer to peer learning by doing same experiment through fast learners.

Instructions for Students

1. Organize the work individually or in the group and make record of all observations.
2. Students shall develop maintenance skill as expected by industries.
3. Student shall attempt to develop related hand-on skills and build confidence.
4. Student shall develop the habits of evolving more ideas, innovations, skills etc.
5. Student shall refer technical magazines and data books.
6. Student should develop habit to submit the practical on date and time.
7. Student should well prepare while submitting write-up of exercise.

Progressive Assessment Sheet

Sr. No	Experiment/Practical Outcome	Page	Date Perform	Marks (10)	Sign
1	**Environment Setup** i. Install and configure the Python environment. Run basic Python commands to verify the Python environment.				
2	**Input-Output** i. Write a program to read your name, contact number, email, and birthdate and print those details on the screen.				
3	**Variables, Operators and Expressions** i. Write a program to convert temperature from Celsius to Fahrenheit. Equation to convert Celsius to Fahrenheit: $F = (9/5) * C + 32$ ii. Write a program to compute the slope of a line between two points (x1, y1) and (x2, y2). $Slope = \frac{y_2 - y_1}{x_2 - x_1}$ iii. Write a program to calculate simple and compound interest. $Simple\ Interest = \frac{P*R*T}{100}$ $Compound\ Interest = P * \left(1 + \frac{R}{100*n}\right)^{n*T}$ iv. Write a program to get change values in Quarter, Dime, Nickels and Pennies, and calculate the value of change in Dollars. Consider Quarter = 0.25 $, Dime = 0.10 $, Nickels = 0.05 $ and Penny = 0.01 $. v. Write a program to find a maximum of given three numbers (Use ternary operator). vi. Write a program to calculate area and volume of Sphere. $Area\ of\ Sphere = 4\pi r^2$ $Volume\ of\ Sphere = \frac{4}{3}\pi r^3$ vii. Write a program that computes the real roots of a given quadratic equation (Use math library). $Discriminant\ \Delta = b^2 - 4ac$ $Real\ Roots = \frac{-b \pm \sqrt{\Delta}}{2a}$				

	viii. Write a program to determine the length of ladder required to reach a given height when leaned against the house. The height and the angle of the ladder are given as inputs (Use math Library).				
4	<u>**Decision Making Structures**</u> i. A year is a Leap year if it is divisible by 4, unless it is a century year that is not divisible by 400 (1800 and 1900 are not leap years, 1600 and 2000 are leap years). Write a program that calculates whether a given year is a leap year or not. ii. Many companies pay time-and-a-half for any hours worked above 40 hours in a given week. Write a program to input the number of hours worked and hourly rate and calculate the total wages for the week. iii. The Body Mass Index (BMI) is calculated as a person's weight (in kg), divided by the square of the person's height (in meters). If the BMI is between 19 and 25, the person is healthy. If the BMI is below 19, then the person is underweight. If the BMI is above 25, then the person is overweight. Write a program to get a person's weight (in kgs) and height (in cms) and display a message whether the person is healthy, underweight or overweight. $BMI = \frac{Weight\ in\ kg}{(Height\ in\ m)^2}$ iv. Write a program to read the marks and assign a grade to a student. Grading system: A (>=90), B (80-89), C (70-79), D (60-69), E (50-59), F (<50). (Use if-elif-else)				
5	<u>**Loops**</u> i. Write a program to read n numbers from users and calculate the average of those n numbers. ii. Write a program that prompts the user to enter 10 integers and displays all the combinations of picking two numbers from the 10. iii. Write programs to print below patterns: `    *` 1 `   * *` 1 2 `  * * *` 1 2 3 ` * * * *` 1 2 3 4 `* * * * *` 1 2 3 4 5 iv. Write a program that displays an ASCII character table from ! to ~. Display the ASCII value of a				

	character in decimal and hexadecimal. Display five characters per line. v. Write a program to sum the following series: $\frac{1}{3}+\frac{3}{5}+\frac{5}{7}+\frac{7}{9}+\frac{9}{11}+\frac{11}{13}+\ldots+\frac{95}{97}+$ vi. A positive integer is called a perfect number if it is equal to the sum of all of its positive divisors, excluding itself. For example, 6 is the first perfect number, because 6 = 3 + 2 + 1, the next is 28 = 14 + 7 + 4 + 2 + 1. There are four perfect numbers that are less than 10,000. Write a program to find these four numbers.				
6	**<u>List</u>** i. Write a program to perform the below operations on the list: ● Create a list. ● Add/Remove an item to/from a list. ● Get the number of elements in the list. ● Access elements of the list using the index. ● Sort the list. ● Reverse the list. ii. Write a program to read n numbers from a user and print: ● Number of positive numbers. ● Number of negative numbers. ● Number of zeros. ● Number of odd numbers. ● Number of even numbers. ● Average of all numbers. iii. Write a program that counts the occurrences of each digit in a string. The program counts how many times a digit appears in the string. For example, if the input is "12203AB3", then the output should output 0 (1 time), 1 (1 time), 2 (2 times), 3 (2 times). iv. Write a program to eliminate duplicate values in the list. v. Write a program to randomly fill in 0s and 1s into a 4x4 2-dimension list, print the list and find the rows and columns with the most number of 1s.				
7	**<u>Tuples, Sets and Dictionaries</u>** i. Write a program to perform below operations on tuple: ● Create a tuple with different data types. ● Print tuple items. ● Convert tuple into a list. ● Remove data items from a list. ● Convert list into a tuple.				

	• Print tuple items. ii. Write a program to perform below operations on set: • Create two different sets with the data. • Print set items. • Add/remove items in/from a set. • Perform operations on sets: union, intersection, difference, symmetric difference, check subset of another set. iii. Write a program to perform below operations on dictionary: • Create a dictionary. • Print dictionary items. • Add/remove key-value pair in/from a dictionary. • Check whether a key exists in a dictionary. • Iterate through a dictionary. • Concatenate multiple dictionaries. iv. Write a program that is given a dictionary containing the average daily temperature for each day of the week, and prints all the days on which the average temperature was between 40 and 50 degrees. v. Write a program to repeatedly prompt the user to enter the capital of a state. Upon receiving the user's input, the program reports whether the answer is correct. Assume the states and their capitals are stored in dictionaries as key-value pairs.				
8	**<u>Functions</u>** i. Write a program that defines a function (shuffle) to scramble a list into a random order, like shuffling a deck of cards. ii. Write a program that defines a function to return a new list by eliminating the duplicate values in the list. iii. Write a program to print Fibonacci sequence up to n numbers using recursion. Fibonacci sequence is defined as below: $Fibonacci\ Sequence = 1\ 1\ 2\ 3\ 5\ 8\ 13\ 21 \ldots$ $where\ n^{th} term\ x_n = x_{n-1} + x_{n-2}$ iv. Write a program that defines a function to determine whether input number n is prime or not. A positive whole number n > 2 is prime, if no number between 2 and $\sqrt{n}$ (inclusive) evenly divides n. If n is not prime, the program should				

	quit as soon as it finds a value that evenly divides n. v. Write a program that defines a function to find the GCD of two numbers using the algorithm below. The greatest common divisor (GCD) of two values can be computed using Euclid's algorithm. Starting with the values m and n, we repeatedly apply the formula: n, m = m, n%m until m is 0. At that point, n is the GCD of the original m and n (Use Recursion). vi. Write a program that lets the user enter the loan amount, number of years, and interest rate, and defines a function to calculate monthly EMI, total payment and display the amortization schedule for the loan.				
9	<u>**Modules**</u> i. Write a program that defines functions (mean and deviation), that computes mean and standard deviation of given numbers. The formula for the mean and standard deviation of n numbers is given as: $mean = \sum_{i=1}^{n} x_i = \frac{x_1 + x_2 + ... + x_n}{n}$ $deviation = \sqrt{\frac{\sum_{i=1}^{n}(x_i - mean)^2}{n-1}}$ ii. Write a program that plays the popular scissor-rock-paper game. (A scissor can cut a paper, a rock can knock a scissor, and a paper can wrap a rock.) The program randomly generates a number 0, 1, or 2 representing scissor, rock, and paper. The program prompts the user to enter a number 0, 1, or 2 and displays a message indicating whether the user or the computer wins, loses, or draws. iii. Write a program to print the dates of all the Sundays in a given year. iv. Write a program to display a graph for ReLU (Rectified Linear Unit) function. ReLU function is defined as below: $y = max(0, x)$ Consider the range of *x* from -5 to 5. v. Write a program to create a list representing the results of 100 students in a test, where each element represents a student's marks (between 0 to 10), and display a histogram for the result.				

	vi. Create a user defined module with simple functions for: addition, subtraction, multiplication, division, modulo, square, factorial. Write a program to import the module and access functions defined in the module.				
10	**<u>String Processing</u>** i. Write a program to check whether a given string is palindrome or not. ii. Write a program to read a string containing letters, each of which may be in either uppercase or lowercase, and return a tuple containing the number of vowels and consonants in the string. iii. Write a program to read a date in the format DD/MM/YYYY and print the same date in MM-DD-YYYY format. iv. Write a program that checks whether two words are anagrams. v. Two words are anagrams if they contain the same letters. For example, silent and listen are anagrams. vi. Write a program that allows users to enter six-digit RGB color codes and converts them into base 10. In this format, the first two hexadecimal digits represent the amount of red, the second two the amount of green, and the last two the amount of blue. For example: If a user enters FF6347, then the output should be Red (255), Green (99) and Blue (71). vii. Numerologists claim to be able to determine a person's character traits based on the "numeric value" of a name. The value of a name is determined by summing up the values of the letters of the name, where "a" is 1 "b" is 2 "c" is 3 and so on up to "z" being 26. For example, the name "Python" would have the value 16 + 25 + 20 + 8 + 15 + 14 = 98. Write a program that calculates the numeric value of a name provided as input.				
11	**<u>File Handling</u>** i. Write a program to perform the below operations on files: ● Create a text file and write a string to it. ● Read an entire text file. ● Read a text file line by line. ● Write a string to a file. ● Write a list of strings to a file. ● Count the number of lines, words in a file.				

	ii. Write a program that reads a text file and counts the occurrences of each alphabet in the file. The program should prompt the user to enter the filename. iii. Write a program that reads a text file and displays all the numbers found in the file. iv. Write an automated censor program that reads the text from a file and creates a new file where all of the four-letter words have been replaced by "****". You can ignore punctuation, and you may assume that no words in the file are split across multiple lines. v. Write a program that reads a text file and calculates the average word length and sentence length in that file. vi. Write a program that reads two strings stored in two different text files and prints a string containing the characters of each string interleaved. Remove white spaces from both strings before string interleaving. For example, Two strings "Hello World" and "Sky is the Limit" should generate output "HSeklyliosWtohrelLdimit".				

Date: _____________

Practical No. 1: **Environment Setup.**

i. Install and configure the Python environment. Run basic Python commands to verify the Python environment.

A. Objectives:

A development environment is required to write, compile, run, and debug any application. This practical will help student to set up an environment for executing Python scripts using Python interpreter.

B. Relevant Program Outcomes (POs):

i. **Basic and Discipline specific knowledge (PO1):** Apply knowledge of basic mathematics, science and engineering fundamentals and engineering specialization to solve the Computer Engineering problems.

ii. **Engineering Tools, Experimentation and Testing (PO4):** Apply modern Computer Engineering tools and appropriate technique to conduct standard tests and measurements.

iii. Life-long learning (PO7): Ability to analyse individual needs and engage in updating in the context of technological changes in field of engineering.

C. Competency and Practical Skills:

This practical is expected to develop the following skills for the industry-identified competency '**Develop simple applications using scripting language Python.**'

i. Installing and configuring development environment for Python.

ii. Programming skills.

iii. Debugging skills.

D. Relevant Course Outcomes (COs):

i. Develop programs to solve the given simple computational problems.

E. Practical Outcomes:

i. Install and configure Python development environment.

F. Relevant Affective domain Outcomes (ADOs):

i. Maintain tools and equipments.

ii. Follow Coding standards and practices.

iii. Follow ethical practices

G. Prerequisite Theory:

Python is an open-source, high-level, general-purpose programming language used for software development. It is one of the most popular programming languages in the world today. It is a widely used programming language in various domains, such as

- Automation
- Artificial Intelligence, Machine Learning
- Data Science
- Blockchain
- Big Data
- Server-side Web Development
- Tools Development
- Game Programming

History of Python:

- In February 1991, Van Rossum implemented initial version of Python and published the code (version 0.9.0).
- Python 1.0 was released in January 1994. The major new features included in this release were the functional programming tools lambda, map, filter and reduce.
- Python 2.0 added new features such as list comprehensions, garbage collection systems.
- Python 3.0 (also called "Python 3000" or "Py3K") was released on December 3, 2008. It was designed to rectify fundamental design flaws in the language. Python had accumulated new and redundant ways to program the same task, Python 3.0 had an emphasis on removing duplicative constructs and modules, in keeping with the Zen of Python: "There should be one— and preferably only one —obvious way to do it".
- Python is now being developed and maintained by a large team of volunteers and is available for free from the Python Software Foundation.

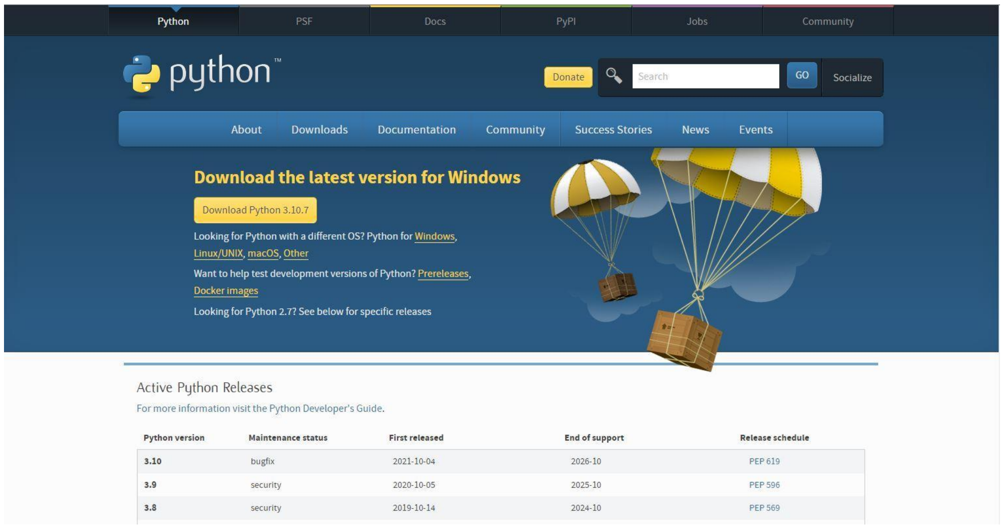
H. Resources Required:

Sr. No	Instrument /Components	Configuration/Specification
1.	Computer System	Processor: RAM: Operating System:
2.	Python Interpreter	Python Version:
3.	Text Editor	Editor:
4.	Internet Connection	Yes/No

I. Procedure:

Below steps describes installation of Python Interpreter on Windows operating system.

i. Visit https://www.python.org/downloads and download the latest release of Python. If you want to install specific version of Python then previous versions can be downloaded from links available in the page.



Download the latest version for Windows

Looking for Python with a different OS? Python for Windows, Linux/UNIX, macOS, Other

Want to help test development versions of Python? Prereleases, Docker images

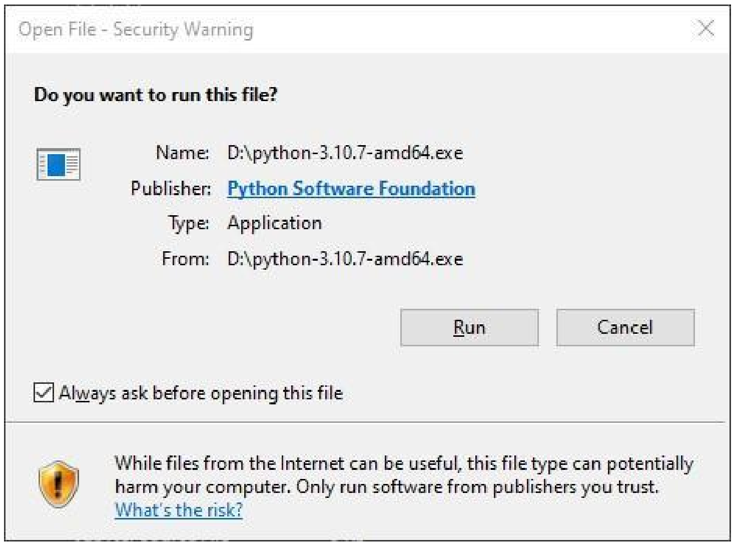
Looking for Python 2.7? See below for specific releases

Active Python Releases

For more information visit the Python Developer's Guide.

Python version	Maintenance status	First released	End of support	Release schedule
3.10	bugfix	2021-10-04	2026-10	PEP 619
3.9	security	2020-10-05	2025-10	PEP 596
3.8	security	2019-10-14	2024-10	PEP 569

ii. Double-click the executable file, which is downloaded. It will open **Open File - Security Warning** dialog box. Click on Run.



Do you want to run this file?

Name: D:\python-3.10.7-amd64.exe

Publisher: Python Software Foundation

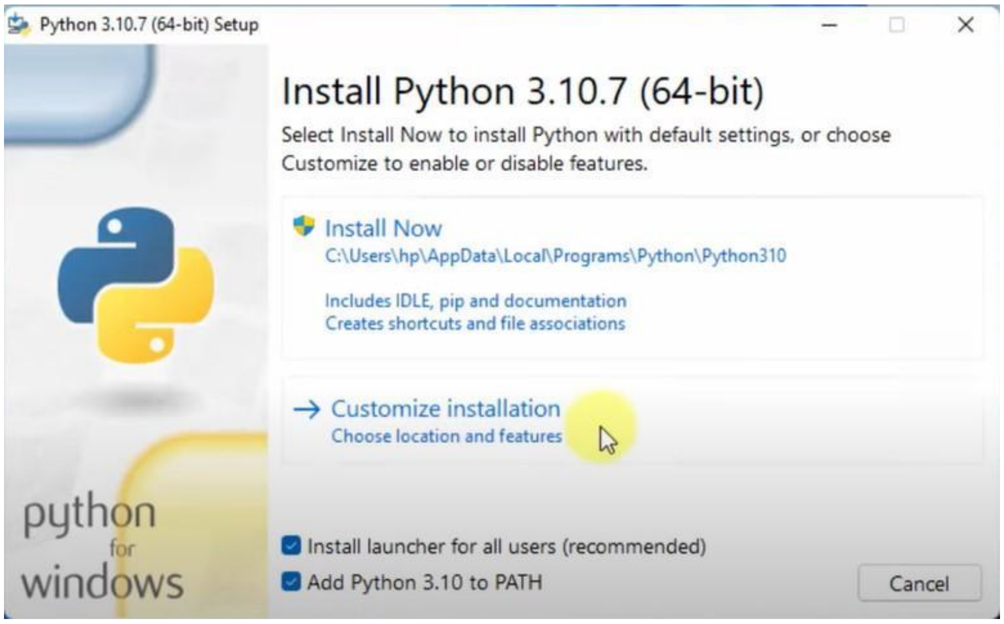
Type: Application

From: D:\python-3.10.7-amd64.exe

☑ Always ask before opening this file

While files from the Internet can be useful, this file type can potentially harm your computer. Only run software from publishers you trust. What's the risk?

iii. In the next **Python 3.10.7 (64-bit) Setup** dialog box, click on *Customize installation*.


Python 3.10.7 (64-bit) Setup

Install Python 3.10.7 (64-bit)

Select Install Now to install Python with default settings, or choose Customize to enable or disable features.

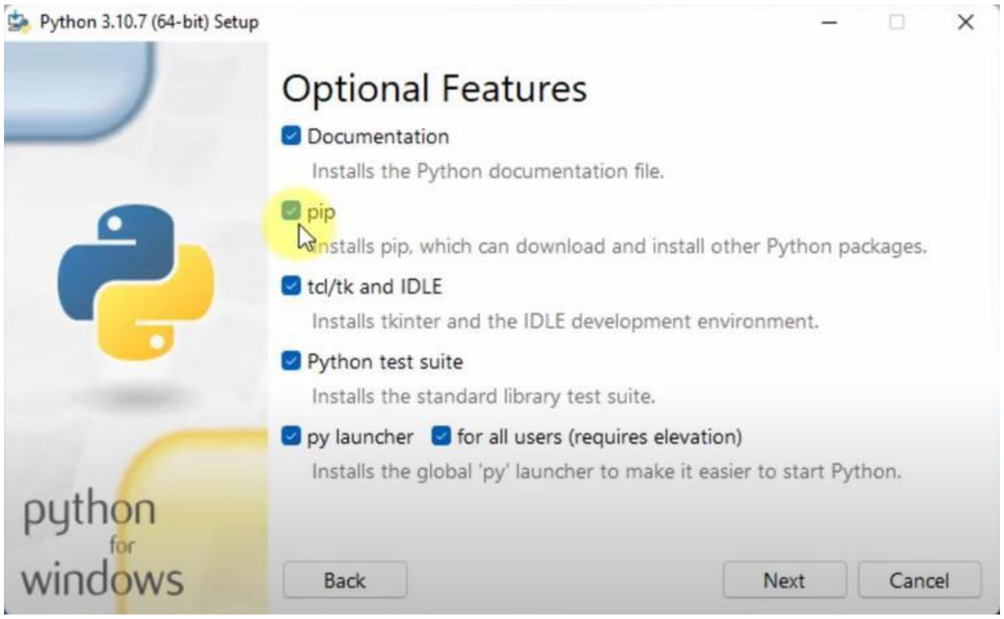
Install Now
C:\Users\hp\AppData\Local\Programs\Python\Python310

Includes IDLE, pip and documentation
Creates shortcuts and file associations

→ Customize installation
Choose location and features

☑ Install launcher for all users (recommended)
☑ Add Python 3.10 to PATH

iv. In the next dialog box, select *pip* and other optional features to be installed and click on *Next*.


Python 3.10.7 (64-bit) Setup

Optional Features

☑ Documentation
Installs the Python documentation file.

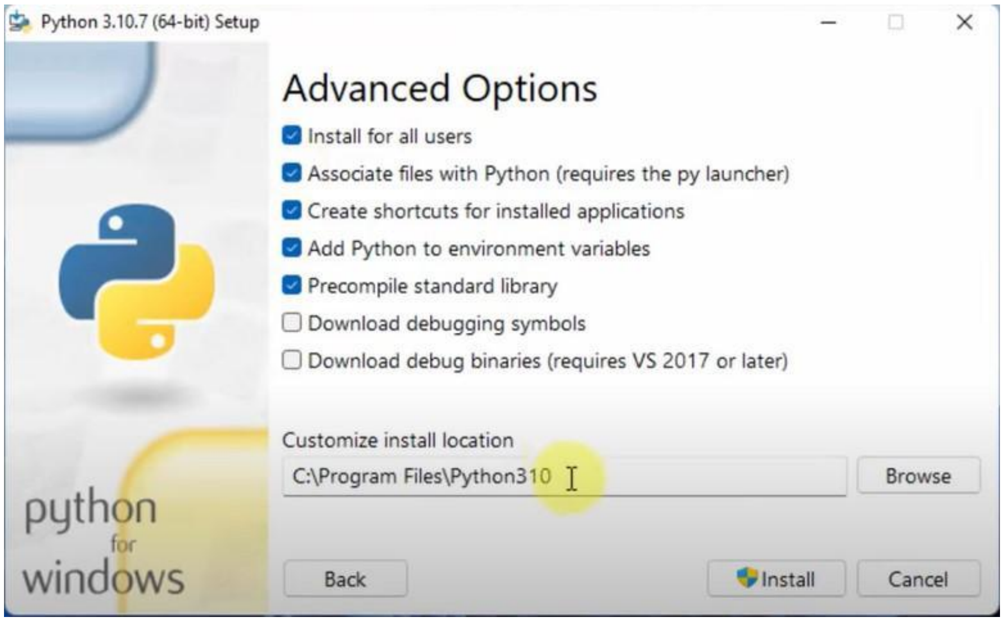
☑ pip
Installs pip, which can download and install other Python packages.

☑ tcl/tk and IDLE
Installs tkinter and the IDLE development environment.

☑ Python test suite
Installs the standard library test suite.

☑ py launcher ☑ for all users (requires elevation)
Installs the global 'py' launcher to make it easier to start Python.

v. In the next dialog box, select advanced options to be installed. If you wish, you can change installation path by changing Customize install location. Click on Install. Select Add Python to environment variables option to run Python from command prompt.


Python 3.10.7 (64-bit) Setup

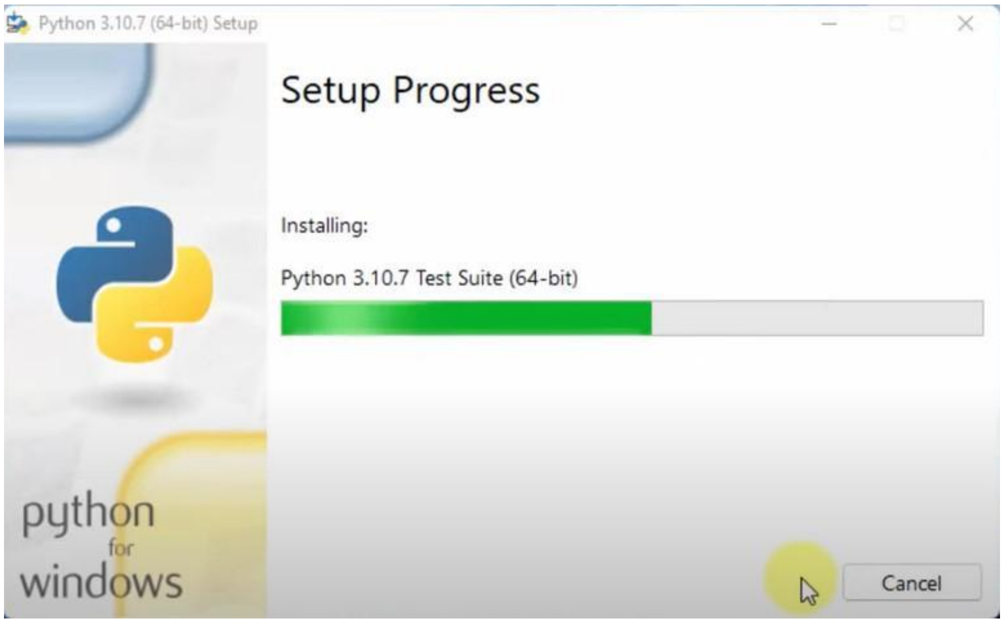
Advanced Options

- [✓] Install for all users
- [✓] Associate files with Python (requires the py launcher)
- [✓] Create shortcuts for installed applications
- [✓] Add Python to environment variables
- [✓] Precompile standard library
- [ ] Download debugging symbols
- [ ] Download debug binaries (requires VS 2017 or later)

Customize install location

C:\Program Files\Python310

vi. Wait for the installation to complete.

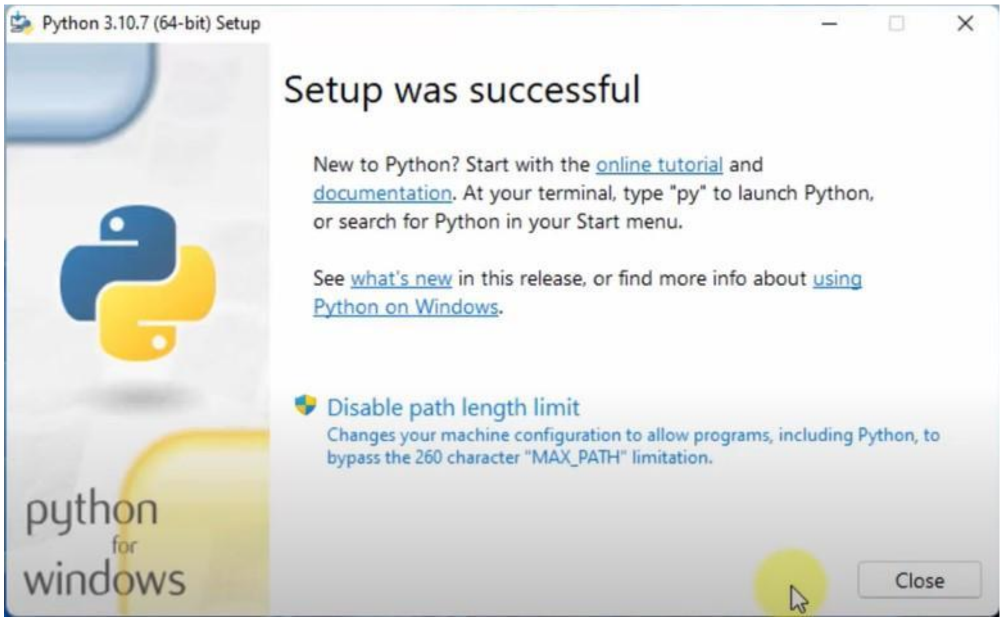

Python 3.10.7 (64-bit) Setup

Setup Progress

Installing:

Python 3.10.7 Test Suite (64-bit)

vii. After completion, you will get *Setup was successful*. Cilck on *Close*.



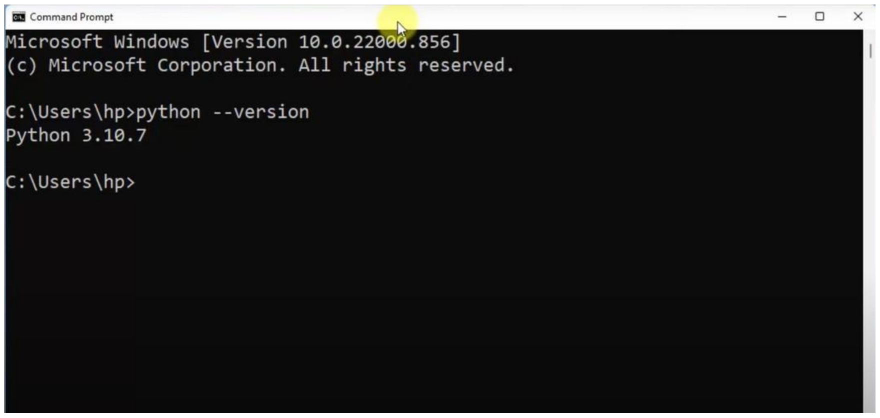
Setup was successful

New to Python? Start with the online tutorial and documentation. At your terminal, type "py" to launch Python, or search for Python in your Start menu.

See what's new in this release, or find more info about using Python on Windows.

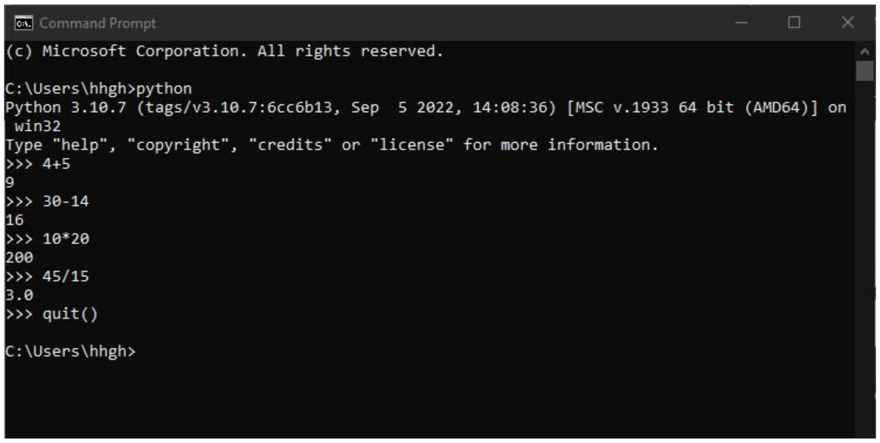
Disable path length limit
Changes your machine configuration to allow programs, including Python, to bypass the 260 character "MAX_PATH" limitation.

viii. To verify installation, open command prompt and run *python --version* command. It will show current version of Python interpreter.



```
Microsoft Windows [Version 10.0.22000.856]
(c) Microsoft Corporation. All rights reserved.

C:\Users\hp>python --version
Python 3.10.7

C:\Users\hp>
```

ix. In the command prompt, run python command. Now you can use prompt as interpreter, where you can run python commands. Run simple mathemetical operations on prompt as shown in below figure. Run quit() to exit from command prompt for python.

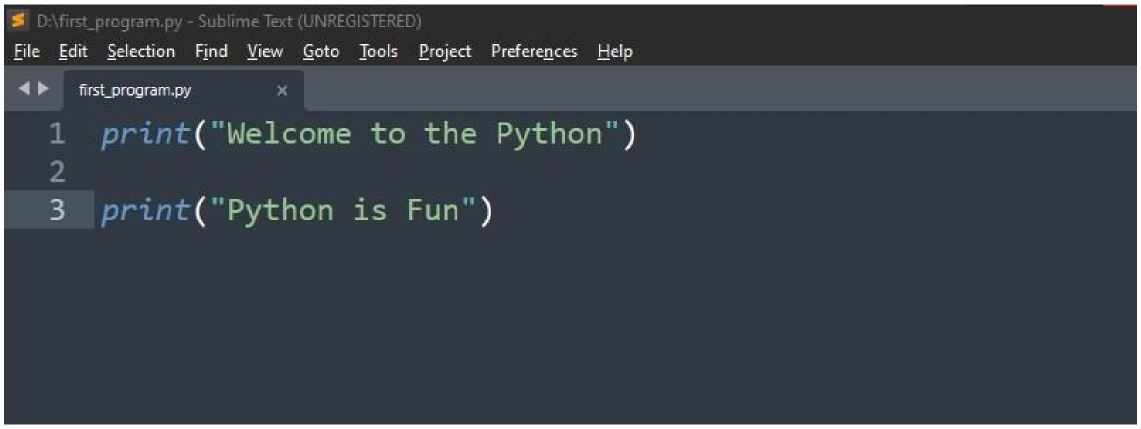


```
(c) Microsoft Corporation. All rights reserved.

C:\Users\hhgh>python
Python 3.10.7 (tags/v3.10.7:6cc6b13, Sep  5 2022, 14:08:36) [MSC v.1933 64 bit (AMD64)] on
 win32
Type "help", "copyright", "credits" or "license" for more information.
>>> 4+5
9
>>> 30-14
16
>>> 10*20
200
>>> 45/15
3.0
>>> quit()

C:\Users\hhgh>
```

To install Python Interpreter on Linux, follow Guide for Installation of Python Interpreter on Linux (https://www.geeksforgeeks.org/how-to-install-python-on-linux).

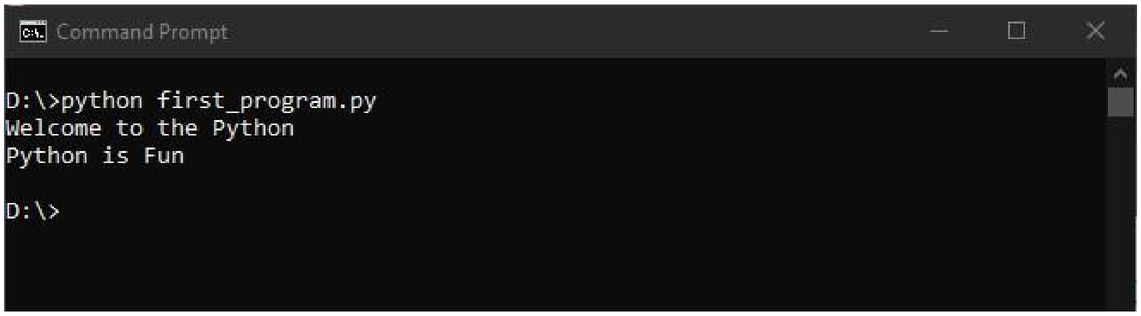
Writing First Program:

Write simple print program and store it as *first_program.py*. Run the Python script from command prompt.


D:\first_program.py - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help
first_program.py

```python
print("Welcome to the Python")

print("Python is Fun")
```

Output:



```
D:\>python first_program.py
Welcome to the Python
Python is Fun

D:\>
```

J. Practical related Quiz:

i. Which website is the official source for downloading Python?

A) python.com

B) python.org

C) pythonlanguage.org

D) pythonsource.com

ii. Which of the following is not an installation option during Python installation?

A) Add Python to PATH

B) Customize installation location

C) Install additional libraries

D) Configure the Python version

iii. What is the role of the PATH variable during Python installation?

A) It specifies the location of the Python interpreter

B) It adds Python to the computer's environment variables

C) It configures the version of Python to be installed

D) It determines the installation location of Python

iv. Which operating systems are supported for Python installation?

A) Windows only

B) Mac OS only

C) Linux only

D) Windows, Mac OS, and Linux

v. What is the recommended version of Python for most users?

A) Python 2.x

B) Python 3.x

vi. What is the purpose of the virtual environment in Python installation?

A) To isolate Python environments for different projects

B) To run Python code without installation

C) To provide additional security during Python execution

D) To improve Python's performance

vii. Which of the following tools is commonly used to manage Python installations and packages?

A) PyCharm

B) Visual Studio Code

C) Anaconda

D) Git

K. References:

i. https://www.python.org/
ii. https://wiki.python.org/moin/BeginnersGuide/Overview
iii. https://docs.python.org/3/using/windows.html
iv. https://www.geeksforgeeks.org/how-to-install-python-on-linux
v. https://opensource.com/article/20/4/install-python-linux
vi. https://www.dataquest.io/blog/installing-python-on-mac
vii.

L. Assessment Rubrics:

Criteria	% of point	Rubrics	Marks
C1: Program Completeness/ Correctness	50 %	Excellent (10-8 marks): Completed programs/scripts correctly as per the requirements.	
		Adequate (7-6 marks): Completed programs/scripts correctly with approx. 70% requirements.	
		Poor (5-4 marks): Completed programs/scripts correctly with 70% - 50% requirements.	
		Unsatisfactory (0-3 marks): Completed programs/ scripts correctly with less than 50% requirements.	
C2: Readability	25 %	Excellent (10-8 marks): The code is clean, well-organized and very easy to understand.	
		Adequate (7-6 marks): The code is fairly easy to read and understand.	
		Poor (5-4 marks): The code is readable only by someone who knows what it is supposed to be doing.	
		Unsatisfactory (0-3 marks): The code is poorly organized and very difficult to understand.	
C3: Coding Standards/ Documentation	25 %	Excellent (10-8 marks): ● Coding standards are followed in complete code while naming variables/functions/ classes, explaining the purpose of variables/ functions/classes. ● The Complete code is well-documented with comments explaining the code.	
		Adequate (7-6 marks): ● Coding standards are followed in most of the code while naming variables/functions/ classes, explaining the purpose of variables/ functions/classes. ● Most of the code is documented with comments explaining the code.	
		Poor (5-4 marks): ● Coding standards are followed in very little code while naming variables/functions/ classes, explaining the purpose of variables/ functions/classes. ● Very little code is documented with comments explaining the code.	
		Unsatisfactory (0-3 marks): ● Coding standards are not followed properly. ● Proper comments are not written explaining the code.	
Total Marks for Practical = 0.5 * (Marks of C1) + 0.25 * (Marks of C2) + 0.25 * (Marks of C3)			

Signature with Date:

Date: _____________

Practical No. 2: Input-Output

i. Write a program to read your name, contact number, email, and birthdate and print those details on the screen.

A. Objectives:

In many application program needs some inputs from user as well as display some information on output. This practical will help students to write Python scripts that takes inputs from user as well display on output.

B. Relevant Program Outcomes (POs):

i. **Basic and Discipline specific knowledge (PO1):** Apply knowledge of basic mathematics, science and engineering fundamentals and engineering specialization to solve the Computer Engineering problems.

ii. **Problem analysis (PO2):** Identify and analyse well-defined Computer Engineering problems using codified standard methods.

iii. **Design/development of solutions (PO3):** Design solutions for Computer Engineering well-defined technical problems and assist with the design of systems components or processes to meet specified needs.

iv. **Engineering Tools, Experimentation and Testing (PO4):** Apply modern Computer Engineering tools and appropriate technique to conduct standard tests and measurements.

C. Competency and Practical Skills:

This practical is expected to develop the following skills for the industry-identified competency '**Develop simple applications using scripting language Python.**'

i. Programming skills.

ii. Debugging skills.

D. Relevant Course Outcomes (COs):

i. Develop programs to solve the given simple computational problems.

E. Practical Outcomes:

i. Write Python script to read data from user and display data to user.

F. Relevant Affective domain Outcomes (ADOs):

i. Maintain tools and equipments.

ii. Follow Coding standards and practices.

iii. Follow ethical practices

G. Prerequisite Theory:

- ***Input Function*** allows to read input from the user on the console. You can use the input() function to ask the user to input a value. When the input function is called it stops the program and waits for the user's input. When the user presses enter, the program resumes and returns what the user typed.

  input(prompt=None)

  when, the prompt is passed as a string value, it will be displayed as a prompt to the user to enter a value. Default value of prompt is None.

- ***Outuput Function*** allows user to display results on console. You can use print() function to display results/message.

  print(value, ..., sep = ' ', end = '\n' , file = sys.stdout, flush = False)

  value - value to be printed

  ... - it means we can pass multiple values

  sep - separator used to separate multiple values

  end - value of end parameter is printed at the last

  file - file where output is printed

For example:

```
variable = input("Enter a value: ")
print("variable = ", variable)
print("Type of variable ", type(variable))
```

Output of the above code:

```
Enter a value: 5
variable = 5
Type of variable <class 'str'>
```

Note:

- This function first takes the input from the user and converts it into a string. The type of the returned object always will be <type 'str'>.
- If your input datatype is not string then, you need to explicitly convert input value to other data type using typecasting.

```
variable = int(input("Enter a value: "))
print("variable = ", variable)
print("Type of variable ", type(variable))
```

Output of the above code:

```
Enter a value: 7
variable = 7
Type of variable <class 'int'>
```

H. Resources Required:

Sr. No	Instrument /Components	Configuration/Specification
1.	Computer System	Processor: RAM: Operating System:
2.	Python Interpreter	Python Version:
3.	Text Editor	Editor:

I. Source code and Output:

i. Write a program to read your name, contact number, email, and birthdate and print those details on the screen.

Source Code:

Output:

J. Practical related Quiz:

i. Which function is used to display output in Python?

A) input()

B) print()

C) read()

D) display()

ii. Which function is used to read input from the user in Python?

A) input()

B) print()

C) read()

D) display()

iii. Which of the following is a correct way to prompt the user for input in Python?

A) input("Enter a value: ")

B) "Enter a value: " = input()

C) input = "Enter a value: "

D) print("Enter a value: ")

iv. Which of the following is true regarding the input() function in Python?

A) It always returns an integer value.

B) It can only accept numeric input.

C) It returns a string value.

D) It reads input from a file.

v. What will be the output of the following code?

```
name = input("Enter your name: ")
print("Hello, " + name + "!")
```

A) Enter your name: (user input)

B) Hello, (user input)!

C) Enter your name: Hello, (user input)!

D) Error: Invalid syntax.

vi. Which of the following is the correct way to print the value of a variable x using the print() function?

A) print(x)

B) print 'x'

C) print "x"

D) print(x())

vii. What is the purpose of the sep parameter in the print() function?

A) It specifies the character used to separate multiple printed values.

B) It changes the standard output stream.

C) It adds a newline character at the end of the printed output.

D) It specifies the number of characters to print.

viii. What will be the output of the following code?

```
print("Hello", "World", sep="-")
```

A) Hello World

B) HelloWorld

C) Hello-World

D) Error: Invalid syntax.

K. References:

i. https://docs.python.org/3/library/functions.html

ii. https://docs.python.org/3/tutorial/inputoutput.html

iii. https://www.geeksforgeeks.org/input-and-output-in-python/

iv. https://www.programiz.com/python-programming/input-output-import

v. https://www.scaler.com/topics/python/python-input-and-output/

L. Assessment Rubrics:

Criteria	% of point	Rubrics	Marks
C1: Program Completeness/ Correctness	50 %	Excellent (10-8 marks): Completed programs/scripts correctly as per the requirements.	
		Adequate (7-6 marks): Completed programs/scripts correctly with approx. 70% requirements.	
		Poor (5-4 marks): Completed programs/scripts correctly with 70% - 50% requirements.	
		Unsatisfactory (0-3 marks): Completed programs/ scripts correctly with less than 50% requirements.	
C2: Readability	25 %	Excellent (10-8 marks): The code is clean, well-organized and very easy to understand.	
		Adequate (7-6 marks): The code is fairly easy to read and understand.	
		Poor (5-4 marks): The code is readable only by someone who knows what it is supposed to be doing.	
		Unsatisfactory (0-3 marks): The code is poorly organized and very difficult to understand.	
C3: Coding Standards/ Documentation	25 %	Excellent (10-8 marks): ● Coding standards are followed in complete code while naming variables/functions/ classes, explaining the purpose of variables/ functions/classes. ● The Complete code is well-documented with comments explaining the code.	
		Adequate (7-6 marks): ● Coding standards are followed in most of the code while naming variables/functions/ classes, explaining the purpose of variables/ functions/classes. ● Most of the code is documented with comments explaining the code.	
		Poor (5-4 marks): ● Coding standards are followed in very little code while naming variables/functions/ classes, explaining the purpose of variables/ functions/classes. ● Very little code is documented with comments explaining the code.	
		Unsatisfactory (0-3 marks): ● Coding standards are not followed properly. ● Proper comments are not written explaining the code.	
Total Marks for Practical = 0.5 * (Marks of C1) + 0.25 * (Marks of C2) + 0.25 * (Marks of C3)			

Signature with Date:

Practical No. 3: **Variables, Operators and Expressions**

i. Write a program to convert temperature from Celsius to Fahrenheit. Equation to convert Celsius to Fahrenheit:

$F = (9/5) * C + 32.$

ii. Write a program to compute the slope of a line between two points (x1, y1) and (x2, y2).

$$Slope = \frac{y_2 - y_1}{x_2 - x_1}$$

iii. Write a program to calculate simple and compound interest.

$$Simple\ Interest = \frac{P * R * T}{100}$$

$$Compound\ Interest = P * \left(1 + \frac{R}{100 * n}\right)^{n * T}$$

iv. Write a program to get change values in Quarter, Dime, Nickels and Pennies, and calculate the value of change in Dollars. Consider Quarter = 0.25 $, Dime = 0.10 $, Nickels = 0.05 $ and Penny = 0.01 $.

v. Write a program to find a maximum of given three numbers (Use ternary operator).

vi. Write a program to calculate area and volume of Sphere.

$$Area\ of\ Sphere = 4 \pi r^2$$

$$Volume\ of\ Sphere = \frac{4}{3} \pi r^3$$

vii. Write a program that computes the real roots of a given quadratic equation (Use math library).

$$Discriminant\ \Delta = b^2 - 4\,a\,c$$

$$Real\ Roots = \frac{-b \pm \sqrt{\Delta}}{2\,a}$$

viii. Write a program to determine the length of ladder required to reach a given height when leaned against the house. The height and the angle of the ladder are given as inputs (Use math Library).

A. Objectives:

Variables, Operators and Expressions are core part of any programming language.

i. Variables are used to store data.

ii. Operators are used to perform various types of operations on data.

iii. Anything that you write in Python script is an expression.

iv. This practical will allow students to practise writing Python scripts that use variables, operators, and expressions to solve simple problems.

B. Relevant Program Outcomes (POs):

i. **Basic and Discipline specific knowledge (PO1):** Apply knowledge of basic mathematics, science and engineering fundamentals and engineering specialization to solve the Computer Engineering problems.

ii. **Problem analysis (PO2):** Identify and analyse well-defined Computer Engineering problems using codified standard methods.

iii. **Design/development of solutions (PO3):** Design solutions for Computer Engineering well-defined technical problems and assist with the design of systems components or processes to meet specified needs.

iv. **Engineering Tools, Experimentation and Testing (PO4):** Apply modern Computer Engineering tools and appropriate technique to conduct standard tests and measurements.

C. Competency and Practical Skills:

This practical is expected to develop the following skills for the industry-identified competency '**Develop simple applications using scripting language Python.**'

i. Problem analysis skills.

ii. Programming skills.

iii. Debugging skills.

D. Relevant Course Outcomes (COs):

i. Develop programs to solve the given simple computational problems.

E. Practical Outcomes:

i. Write Python scripts to solve given problem using variables, operators and expressions.

F. Relevant Affective domain Outcomes (ADOs):

i. Maintain tools and equipments.

ii. Follow Coding standards and practices.

iii. Follow ethical practices.

G. Prerequisite Theory:

Data types:

A Data type is a set of values, and a set of operators that may be applied to those values. For example, the integer data type consists of the set of integers, and operators for addition, subtraction, multiplication, and division, among others. Python has the following data types built-in by default:

- None Type: None
- Boolean Type: bool
- Numeric Types: int, float, complex

- Text Type: str
- Sequence Types: list, tuple, range
- Mapping Type: dict
- Set Types: set, frozenset
- Binary Types: bytes, bytearray, memoryview

Variables and Expressions:

Variables are identifiers that are used to reference values stored in the memory. They are called "variables", because they may reference different values.

The statement for assigning a value to a variable is called an assignment statement. In Python, the equal sign (=) is used as the assignment operator. The syntax for assignment statements is as follows:

variable = expression

An **expression** represents a computation involving values, variables, and operators that, taken together, evaluate to a value. A variable can also be used in both sides of the = operator. For example, consider the following code:

```
y = 5 * (6 / 2) + 3 * 2
z = 10
z = z + 5
print("Y =", y)
print("z =", z)

radius = 10
area = radius * radius * 3.14
print("Area =", area)
```

Output of the above code is as below:

```
Y = 21.0
z = 15
Area = 314.0
```

Operators:

An operator is a symbol that represents an operation that may be performed on one or more operands (Variables, constants etc.). Python has following types of operators:

- Arithmetic operators
- Bitwise operators
- Logical operators
- Assignment operators
- Comparison operators
- Identity operators
- Membership operators

Arithmetic Operators:

Operator	Name	Example	Description
+	Addition	a + b	Sum of two operands
-	Subtraction	a - b	Difference of two operands
*	Multiplication	a * b	Multiply tow operands
/	Division (float)	a / b	Quotient of operands
//	Division (int/floor)	a // b	Integer Quotient of operands
%	Modulo	a % b	Reminder of operands
**	Power	a ** b	first operand raised to power second operand

Logical Operators:

Operator	Name	Example	Description
and	Logical AND	a and b	Logical AND operation of a and b
Or	Logical OR	a or b	Logical OR operation of a and b
not	Logical not	not a	Logical NOT operation on a

Bitwise Operators:

Operator	Name	Example	Description
&	Bitwise AND	a & b	Bitwise AND operation between a and b
|	Bitwise OR	a | b	Bitwise OR operation between a and b
^	Bitwise XOR	a ^ b	Bitwise XOR operation between a and b
~	Bitwise NOT	~ a	Bitwise NOT operation on a
<<	Left shift	a << b	Left shift bits of a by b steps
>>	Right shift	a >> b	Right shift bits of a by b steps

Assignment Operators:

Operator	Name	Example	Description
=	Assign	a = b	Value of right operand is assigned to left operand
+=	Add then assign	a += b	Same as a = a + b
-=	Subtract then assign	a -= b	Same as a = a - b
*=	Multiply then assign	a *= b	Same as a = a * b
/=	Divide then assign (Quotient)	a /= b	Same as a = a / b
%=	Divide then assign (Reminder)	a %= b	Same as a = a % b
//=	Divide then assign (Int Quotient)	a //= b	Same as a = a // b
&=	Bitwise AND then assign	a &= b	Same as a = a & b
|=	Bitwise OR then assign	a |= b	Same as a = a | b
^=	Bitwise NOT then assign	a ^= b	Same as a = a ^ b
>>=	Bitwise Right shift then assign	a >>= b	Same as a = a >> b
<<=	Bitwise Left shift then assign	a <<= b	Same as a = a << b

Comparison Operators:

Operator	Name	Example	Description
==	Equal	a == b	Returns True if a is equal to b
!=	Not equal	a != b	Returns True if a is not equal to b
<	Less than	a < b	Returns True if a is less than b

>	Greater than	a > b	Returns True if a is greater than b
<=	Less than or equal to	a <= b	Returns True if a is less than or equal to b
>=	Greater than or equal to	a >= b	Returns True if a is greater than or equal to b

Identity Operators:

Operator	Name	Example	Description
is	Identical	a is b	Returns True is a and b are identical
is not	not identical	a is not b	Returns True is a and b are not identical

Ternery Operator:

Ternary operators allows us to write conditional expressions in a single line without using if-else condition. It evaluate something based on a condition being true or false.

value_if_true if condition else value_if_false

```
a, b = 6, 9
max = a if (a > b) else b
print("Max: ", max)
```

H. Resources Required:

Sr. No	Instrument /Components	Configuration/Specification
1.	Computer System	Processor: RAM: Operating System:
2.	Python Interpreter	Python Version:
3.	Text Editor	Editor:

I. Source code and Output:

i. Write a program to convert temperature from Celsius to Fahrenheit. Equation to convert Celsius to Fahrenheit:

$$F = (9/5) * C + 32.$$

Source Code:

Output:

ii. Write a program to compute the slope of a line between two points (x1, y1) and (x2, y2).

$$Slope = \frac{y_2 - y_1}{x_2 - x_1}$$

Source Code:

Output:

iii. Write a program to calculate simple and compound interest.

$$Simple\ Interest\ =\ \frac{P * R * T}{100}$$

$$Compound\ Interest\ =\ P\ *\ \left(1\ +\ \frac{R}{100 * n}\right)^{n * T}$$

Source Code:

Output:

iv. Write a program to get change values in Quarter, Dime, Nickels and Pennies, and calculate the value of change in Dollars. Consider Quarter = 0.25 $, Dime = 0.10 $, Nickels = 0.05 $ and Penny = 0.01 $.

Source Code:

Output:

v. Write a program to find a maximum of given three numbers (Use ternary operator).

Source Code:

Output:

vi. Write a program to calculate area and volume of Sphere.

$$Area\ of\ Sphere\ =\ 4\,\pi\,r^2$$

$$Volume\ of\ Sphere\ =\ \frac{4}{3}\,\pi\,r^3$$

Source Code:

Output:

vii. Write a program that computes the real roots of a given quadratic equation (Use math library).

$$Discriminant\ \Delta = b^2 - 4ac$$

$$Real\ Roots = \frac{-b \pm \sqrt{\Delta}}{2a}$$

Source Code:

Output:

viii. Write a program to determine the length of ladder required to reach a given height when leaned against the house. The height and the angle of the ladder are given as inputs (Use math Library).

Source Code:
Output:

J. Practical related Quiz:

i. Which of the following is a valid variable name in Python?

A) 1variable

B) my-variable

C) _variable

D) variable!

ii. What is the output of the following code in Python?

```
x = 10
y = 3
result = x / y
print(result)
```

A) 3.3333333333333335

B) 3.33

C) 3

D) 10 / 3

iii. What is the result of the following expression in Python: True and False?

A) True

B) False

C) None

D) Error

iv. What is the value of the variable ‘y’ after executing the following code in Python?

```
x = 5
y = x // 2
```

A) 2.5

B) 2

C) 2.0

D) 3

v. What is the result of the following expression in Python: (True or False) and False?

A) True

B) False

C) None

D) Error

vi. What is the result of the following expression in Python: 8 << 2?

A) 4

B) 8

C) 16

D) 32

vii. Which of the following is the correct way to compare if two variables are equal in Python?

A) x = y

B) x == y

C) x = = y

D) x === y

viii. What is the result of the following expression in Python: 10 > 5 or 3 < 2?

A) True

B) False

C) None

D) Error

ix. Which of the following is the correct way to compare if two variables are not equal in Python?

A) x != y

B) x <> y

C) x = y

D) x !== y

x. What is the result of the following expression in Python: ~10?

A) -10

B) -11

C) 10

D) 11

xi. What is the result of the following expression in Python: 8 ^ 3?

A) 0

B) 4

C) 3

D) 11

K. References:

i. https://www.w3schools.com/python/python_variables.asp

ii. https://www.w3schools.com/python/python_operators.asp

iii. https://www.programiz.com/python-programming/operators

iv. https://www.scaler.com/topics/python/variables-in-python

v. https://www.scaler.com/topics/python/operators-in-python

L. Assessment Rubrics:

Criteria	% of point	Rubrics	Marks
C1: Program Completeness/ Correctness	50 %	Excellent (10-8 marks): Completed programs/scripts correctly as per the requirements.	
		Adequate (7-6 marks): Completed programs/scripts correctly with approx. 70% requirements.	
		Poor (5-4 marks): Completed programs/scripts correctly with 70% - 50% requirements.	
		Unsatisfactory (0-3 marks): Completed programs/ scripts correctly with less than 50% requirements.	
C2: Readability	25 %	Excellent (10-8 marks): The code is clean, well-organized and very easy to understand.	
		Adequate (7-6 marks): The code is fairly easy to read and understand.	
		Poor (5-4 marks): The code is readable only by someone who knows what it is supposed to be doing.	
		Unsatisfactory (0-3 marks): The code is poorly organized and very difficult to understand.	
C3: Coding Standards/ Documentation	25 %	Excellent (10-8 marks): ● Coding standards are followed in complete code while naming variables/functions/ classes, explaining the purpose of variables/ functions/classes. ● The Complete code is well-documented with comments explaining the code.	
		Adequate (7-6 marks): ● Coding standards are followed in most of the code while naming variables/functions/ classes, explaining the purpose of variables/ functions/classes. ● Most of the code is documented with comments explaining the code.	
		Poor (5-4 marks): ● Coding standards are followed in very little code while naming variables/functions/ classes, explaining the purpose of variables/ functions/classes. ● Very little code is documented with comments explaining the code.	
		Unsatisfactory (0-3 marks): ● Coding standards are not followed properly. ● Proper comments are not written explaining the code.	
Total Marks for Practical = 0.5 * (Marks of C1) + 0.25 * (Marks of C2) + 0.25 * (Marks of C3)			

Signature with Date:

Practical No. 4: **Decision-Making Structures**

i. A year is a Leap year if it is divisible by 4, unless it is a century year that is not divisible by 400 (1800 and 1900 are not leap years, 1600 and 2000 are leap years). Write a program that calculates whether a given year is a leap year or not.

ii. Many companies pay time-and-a-half for any hours worked above 40 hours in a given week. Write a program to input the number of hours worked and hourly rate and calculate the total wages for the week.

iii. The Body Mass Index (BMI) is calculated as a person's weight (in kg), divided by the square of the person's height (in meters). If the BMI is between 19 and 25, the person is healthy. If the BMI is below 19, then the person is underweight. If the BMI is above 25, then the person is overweight. Write a program to get a person's weight (in kgs) and height (in cms) and display a message whether the person is healthy, underweight or overweight.

$$BMI = \frac{Weight\ in\ kg}{(Height\ in\ m)^2}$$

iv. Write a program to read the marks and assign a grade to a student. Grading system: A (>=90), B (80-89), C (70-79), D (60-69), E (50-59), F (<50). (Use if-elif-else)

A. Objectives:

Conditional statements are used to perform different actions based on different conditions. This practical will help student to practice writing Python scripts using Decision making structures.

B. Relevant Program Outcomes (POs):

i. **Basic and Discipline specific knowledge (PO1):** Apply knowledge of basic mathematics, science and engineering fundamentals and engineering specialization to solve the Computer Engineering problems.

ii. Problem analysis (PO2): Identify and analyse well-defined Computer Engineering problems using codified standard methods.

iii. Design/development of solutions (PO3): Design solutions for Computer Engineering well-defined technical problems and assist with the design of systems components or processes to meet specified needs.

iv. **Engineering Tools, Experimentation and Testing (PO4):** Apply modern Computer Engineering tools and appropriate technique to conduct standard tests and measurements.

v. **Project Management (PO6):** Use engineering management principles individually, as a team member or a leader to manage projects and effectively communicate about well-defined engineering activities.

vi. **Life-long learning (PO7):** Ability to analyse individual needs and engage in updating in the context of technological changes in field of engineering.

C. Competency and Practical Skills:

This practical is expected to develop the following skills for the industry-identified competency '**Develop simple applications using scripting language Python.**'

i. Problem analysis skills.

ii. Programming skills.

iii. Debugging skills.

D. Relevant Course Outcomes (COs):

i. Apply control flow structures to solve the given problems.

E. Practical Outcomes:

i. Write Python scripts using decision making statements.

F. Relevant Affective domain Outcomes (ADOs):

i. Maintain tools and equipments.

ii. Follow Coding standards and practices.

iii. Follow ethical practices.

G. Prerequisite Theory:

Decision Making Statements:

Controls statements are used to control are used to control the flow of execution of program based on certain conditions. In Python, there are following decision making statements:

- `if` statement
- `if...else` statement
- nested `if...else` statement
- if...elif...else statement

`if` statement:

`if` statement allow us to run a block of code if certain condition is true. If condition is false it will not execute block of code.

```
if condition:
      Block of code
```

`if...else` statement:

`if...else` executes a block of code if certain condition is true and another block of code if condition is false.

```
if condition:
      Block of code
else:
      Block of code
```

<u>Nested `if...else` statement:</u>

There are often times when selection among more than two sets of statements (suites) is needed. For such situations, if statements can be nested, resulting in multi-way selection.

```
if condition:
      if condition:
            Block of code
      else:
            Block of code
else:
      if condition:
            Block of code
      else:
            Block of code
```

<u>`if...elif...else` statement:</u>

It is similar to multiple `if...else` statements. It executes different blocks of code based on different conditions.

```
if condition:
      Block of code
elif condition:
      Block of code
elif condition:
      Block of code
else:
      Block of code
```

H. Resources Required:

Sr. No	Instrument /Components	Configuration/Specification
1.	Computer System	Processor: RAM: Operating System:
2.	Python Interpreter	Python Version:

3.	Text Editor	Editor:

I. Source code and Output:

i. A year is a Leap year if it is divisible by 4, unless it is a century year that is not divisible by 400 (1800 and 1900 are not leap years, 1600 and 2000 are leap years). Write a program that calculates whether a given year is a leap year or not.

Source Code:
Output:

ii. Many companies pay time-and-a-half for any hours worked above 40 hours in a given week. Write a program to input the number of hours worked and hourly rate and calculate the total wages for the week.

Source Code:

Output:

iii. The Body Mass Index (BMI) is calculated as a person's weight (in kg), divided by the square of the person's height (in meters). If the BMI is between 19 and 25, the person is healthy. If the BMI is below 19, then the person is underweight. If the BMI is above 25, then the person is overweight. Write a program to get a person's weight (in kgs) and height (in cms) and display a message whether the person is healthy, underweight or overweight.

$$BMI = \frac{Weight\ in\ kg}{(Height\ in\ m)^2}$$

Source Code:

Output:

iv. Write a program to read the marks and assign a grade to a student. Grading system: A (>=90), B (80-89), C (70-79), D (60-69), E (50-59), F (<50). (Use if-elif-else)

Source Code:

Output:

J. Practical related Quiz:

i. Which keyword is used to check multiple conditions in Python?

A) if

B) else

C) elif

D) and

ii. What is the output of the following code in Python?

```
x = 5
if x > 10:
    print("Greater than 10")
elif x > 5:
    print("Greater than 5")
else:
    print("Less than or equal to 5")
```

A) Greater than 10

B) Greater than 5

C) Less than or equal to 5

D) No output

iii. Which operator is used to check if two values are equal in Python?

A) ==

B) =

C) !=

D) >

iv. Which keyword is used to execute a block of code when all conditions in an if statement are false?

A) else

B) if

C) elif

D) not

v. What is the output of the following code in Python?

```
x = 7
if x % 2 == 0:
    print("Even")
else:
    if x % 3 == 0:
        print("Divisible by 3")
    else:
        print("Not even or divisible by 3")
```

A) Even

B) Divisible by 3

C) Not even or divisible by 3

D) No output

vi. Which keyword is used to check if a value is present in a list or sequence in Python?

A) if

B) else

C) elif

D) in

vii. Which keyword is used to check if a value is not equal to another value in Python?

A) ==

B) =

C) !=

D) <>

K. References:

i. https://docs.python.org/3/tutorial/controlflow.html#if-statements

ii. https://www.w3schools.com/python/python_conditions.asp

iii. https://www.geeksforgeeks.org/python-if-else/
iv. https://www.scaler.com/topics/python/control-flow-statements-in-python/
v. https://www.programiz.com/python-programming/if-elif-else

L. Assessment Rubrics:

Criteria	% of point	Rubrics	Marks
C1: Program Completeness/ Correctness	50 %	Excellent (10-8 marks): Completed programs/scripts correctly as per the requirements.	
		Adequate (7-6 marks): Completed programs/scripts correctly with approx. 70% requirements.	
		Poor (5-4 marks): Completed programs/scripts correctly with 70% - 50% requirements.	
		Unsatisfactory (0-3 marks): Completed programs/ scripts correctly with less than 50% requirements.	
C2: Readability	25 %	Excellent (10-8 marks): The code is clean, well-organized and very easy to understand.	
		Adequate (7-6 marks): The code is fairly easy to read and understand.	
		Poor (5-4 marks): The code is readable only by someone who knows what it is supposed to be doing.	
		Unsatisfactory (0-3 marks): The code is poorly organized and very difficult to understand.	
C3: Coding Standards/ Documentation	25 %	Excellent (10-8 marks): ● Coding standards are followed in complete code while naming variables/functions/ classes, explaining the purpose of variables/ functions/classes. ● The Complete code is well-documented with comments explaining the code.	
		Adequate (7-6 marks): ● Coding standards are followed in most of the code while naming variables/functions/ classes, explaining the purpose of variables/ functions/classes. ● Most of the code is documented with comments explaining the code.	
		Poor (5-4 marks): ● Coding standards are followed in very little code while naming variables/functions/ classes, explaining the purpose of variables/ functions/classes. ● Very little code is documented with comments explaining the code.	
		Unsatisfactory (0-3 marks): ● Coding standards are not followed properly. ● Proper comments are not written explaining the code.	
Total Marks for Practical = 0.5 * (Marks of C1) + 0.25 * (Marks of C2) + 0.25 * (Marks of C3)			

Signature with Date:

Practical No. 5: Loops

i. Write a program to read *n* numbers from users and calculate the average of those *n* numbers.

ii. Write a program that prompts the user to enter 10 integers and displays all the combinations of picking two numbers from the 10.

iii. Write programs to print below patterns:

```
    *
   * *
  * * *
 * * * *
* * * * *
```
	1
	1 2
	1 2 3
	1 2 3 4
	1 2 3 4 5

iv. Write a program that displays an ASCII character table from '!' to '~'. Display the ASCII value of a character in decimal and hexadecimal. Display five characters per line.

v. Write a program to sum the following series:

$$\frac{1}{3}+\frac{3}{5}+\frac{5}{7}+\frac{7}{9}+\frac{9}{11}+\frac{11}{13}+\ldots+\frac{95}{97}+\frac{97}{99}$$

vi. A positive integer is called a perfect number if it is equal to the sum of all of its positive divisors, excluding itself. For example, 6 is the first perfect number, because 6 = 3 + 2 + 1, the next is 28 = 14 + 7 + 4 + 2 + 1. There are four perfect numbers that are less than 10,000. Write a program to find these four numbers.

A. Objectives:

Loops are used to run same block of code again and again certain number of times. This practical will help student to practice writing Python scripts using Loops.

B. Relevant Program Outcomes (POs):

i. **Basic and Discipline specific knowledge (PO1):** Apply knowledge of basic mathematics, science and engineering fundamentals and engineering specialization to solve the Computer Engineering problems.

ii. Problem analysis (PO2): Identify and analyse well-defined Computer Engineering problems using codified standard methods.

iii. Design/development of solutions (PO3): Design solutions for Computer Engineering well-defined technical problems and assist with the design of systems components or processes to meet specified needs.

iv. **Engineering Tools, Experimentation and Testing (PO4):** Apply modern Computer Engineering tools and appropriate technique to conduct standard tests and measurements.

v. **Project Management (PO6):** Use engineering management principles individually, as a team member or a leader to manage projects and effectively communicate about well-defined engineering activities.

vi. **Life-long learning (PO7):** Ability to analyse individual needs and engage in updating in the context of technological changes in field of engineering.

C. Competency and Practical Skills:

This practical is expected to develop the following skills for the industry-identified competency '**Develop simple applications using scripting language Python.**'

i. Problem analysis skills.

ii. Programming skills.

iii. Debugging skills.

D. Relevant Course Outcomes (COs):

i. Apply control flow structures to solve the given problems.

E. Practical Outcomes:

i. Write Python scripts using loops control structures.

F. Relevant Affective domain Outcomes (ADOs):

i. Maintain tools and equipments.

ii. Follow Coding standards and practices.

iii. Follow ethical practices.

G. Prerequisite Theory:

Loops in Python are used to execute the same block of code a specified number of times. Python supports below loop statements:

- `while` loop
- `for` loop

`while` loop:

while statement is an iterative control statement that repeatedly executes a set of statements based on a provided Boolean expression (condition). All iterative control needed in a program can be achieved by use of the while statement

```
while condition:
      Block of code          # statements to execute
                             # till condition is true
```

- As long as the condition of a while statement is true, the statements within the loop are (re)executed.
- Once the condition becomes false, the iteration terminates and control continues with the first statement after the while loop.
- Note that it is possible that the fi rst time a loop is reached, the condition may be false, and therefore the loop would never be executed.

`for` loop:

In Python for loop is used to iterate through each value in sequence. For loops are used for constructing definite loop.

A for statement is an iterative control statement that iterates once for each element in a specified sequence of elements.

```
For k in sequence:
      Block of code           # statements to execute for each item
                              # in the sequence
```

Variable k is referred to as a loop variable.

The Built-in range Function

Python provides a built-in range function that can be used for generating a sequence of integers that a for loop can iterate over.

The values in the generated sequence include the starting value, up to but not including the ending value. For example, range(1, 11) generates the sequence [1, 2, 3, 4, 5, 6, 7, 8, 9].

```
sum = 0
for k in range(1,11):
      sum = sum + k
print("Sum:",sum)
```

break statement:

- break statement in a loop is used to immediately terminate a loop. Control of the program flows to the statement immediately after the body of the loop.
- If the break statement is inside a nested loop (loop inside another loop), the break statement will terminate the inner most loop.
- Break statement makes the program simpler and easier to read.

```
n = eval(input("Enter an integer >= 2: "))
factor=2
while factor <= n:
      if n % factor == 0:
            break
      factor += 1
print("The smallest factor other than 1 for",n,"is:",factor)
```

continue statement:

- continue statement in a loop is used to skip the rest of the code inside a loop for the current iteration only.
- When the continue statement is executed in the loop, the code inside the loop following the continue statement will be skipped and the next iteration of the loop will begin.

- Continue statements breaks out of an iteration (but not the loop), while the break statement breaks out of the loop.

```
For valin"Hello World":
      if val=="L":
            continue
      print(val)
```

H. Resources Required:

Sr. No	Instrument /Components	Configuration/Specification
1.	Computer System	Processor: RAM: Operating System:
2.	Python Interpreter	Python Version:
3.	Text Editor	Editor:

I. Source code and Output:

i. Write a program to read n numbers from users and calculate the average of those n numbers.

Source Code:
Output:

ii. Write a program that prompts the user to enter 10 integers and displays all the combinations of picking two numbers from the 10.

Source Code:

Output:

iii. Write programs to print below patterns:

```
    *
   * *
  * * *
 * * * *
* * * * *
```

```
1
1 2
1 2 3
1 2 3 4
1 2 3 4 5
```

Source Code:

Output:

iv. Write a program that displays an ASCII character table from ! to ~. Display the ASCII value of a character in decimal and hexadecimal. Display five characters per line.

Source Code:

Output:

v. Write a program to sum the following series:

$$\frac{1}{3} + \frac{3}{5} + \frac{5}{7} + \frac{7}{9} + \frac{9}{11} + \frac{11}{13} + \dots + \frac{95}{97} + \frac{97}{99}$$

Source Code:

Output:

vi. A positive integer is called a perfect number if it is equal to the sum of all of its positive divisors, excluding itself. For example, 6 is the first perfect number, because 6 = 3 + 2 + 1, the next is 28 = 14 + 7 + 4 + 2 + 1. There are four perfect numbers that are less than 10,000. Write a program to find these four numbers.

Source Code:

Output:

J. Practical related Quiz

i. Which loop statement is used to iterate over a sequence of elements in Python?

A) for

B) while

C) do-while

D) switch

ii. Which loop statement is used when the number of iterations is not known in advance?

A) for

B) while

C) do-while

D) switch

iii. What is the output of the following code in Python?

```
for i in range(1, 10, 2):
    print(i)
```

A) 1 2 3 4 5 6 7 8 9

B) 1 3 5 7 9

C) 2 4 6 8 10

D) 0 2 4 6 8

iv. What is the output of the following code in Python?

```
for i in range(3):
    for j in range(2):
        print(i, j)
```

A) 0 0 0 1 1 0 1 1 2 0 2 1

B) 0 0 1 1 2 0 2 1

C) 0 1 2 0 1 2

D) 1 0 1 1 2 0 2 1

v. What is the output of the following code in Python?

```
x = 0
while x < 3:
    x += 1
    if x == 2:
        continue
    print(x)
else:
    print("Loop finished")
```

A) 1 2 3
   Loop finished

B) 1 3
   Loop finished

C) 2
   Loop finished

D) 1
   Loop finished

vi. Which loop statement is used to iterate over the elements of an iterable in reverse order?

A) for

B) while

C) do-while

D) switch

K. References:

i. https://docs.python.org/3/tutorial/controlflow.html

ii. https://www.scaler.com/topics/python/control-flow-statements-in-python

iii. https://www.w3schools.com/python/python_while_loops.asp

iv. https://www.geeksforgeeks.org/python-for-loops

v. https://www.programiz.com/python-programming/for-loop

L. Assessment Rubrics:

Criteria	% of point	Rubrics	Marks
C1: Program Completeness/ Correctness	50 %	<u>Excellent (10-8 marks):</u> Completed programs/scripts correctly as per the requirements.	
		<u>Adequate (7-6 marks):</u> Completed programs/scripts correctly with approx. 70% requirements.	
		<u>Poor (5-4 marks):</u> Completed programs/scripts correctly with 70% - 50% requirements.	
		<u>Unsatisfactory (0-3 marks):</u> Completed programs/ scripts correctly with less than 50% requirements.	
C2: Readability	25 %	<u>Excellent (10-8 marks):</u> The code is clean, well-organized and very easy to understand.	
		<u>Adequate (7-6 marks):</u> The code is fairly easy to read and understand.	
		<u>Poor (5-4 marks):</u> The code is readable only by someone who knows what it is supposed to be doing.	
		<u>Unsatisfactory (0-3 marks):</u> The code is poorly organized and very difficult to understand.	
C3: Coding Standards/ Documentation	25 %	<u>Excellent (10-8 marks):</u> ● Coding standards are followed in complete code while naming variables/functions/ classes, explaining the purpose of variables/ functions/classes. ● The Complete code is well-documented with comments explaining the code.	
		<u>Adequate (7-6 marks):</u> ● Coding standards are followed in most of the code while naming variables/functions/ classes, explaining the purpose of variables/ functions/classes. ● Most of the code is documented with comments explaining the code.	
		<u>Poor (5-4 marks):</u> ● Coding standards are followed in very little code while naming variables/functions/ classes, explaining the purpose of variables/ functions/classes. ● Very little code is documented with comments explaining the code.	
		<u>Unsatisfactory (0-3 marks):</u> ● Coding standards are not followed properly. ● Proper comments are not written explaining the code.	
Total Marks for Practical = 0.5 * (Marks of C1) + 0.25 * (Marks of C2) + 0.25 * (Marks of C3)			

Signature with Date:

Practical No. 6: Lists

i. Write a program to perform the below operations on the list:
   - Create a list.
   - Add/Remove an item to/from a list.
   - Get the number of elements in the list.
   - Access elements of the list using the index.
   - Sort the list.
   - Reverse the list.

ii. Write a program to read n numbers from a user and print:
   - Number of positive numbers.
   - Number of negative numbers.
   - Number of zeros.
   - Number of odd numbers.
   - Number of even numbers.
   - Average of all numbers.

iii. Write a program that counts the occurrences of each digit in a string. The program counts how many times a digit appears in the string. For example, if the input is "12203AB3", then the output should output 0 (1 time), 1 (1 time), 2 (2 times), 3 (2 times).

iv. Write a program to eliminate duplicate values in the list.

v. Write a program to randomly fill in 0s and 1s into a 4x4 2-dimension list, print the list and find the rows and columns with the most number of 1s.

A. Objectives:

In Python, list data structure allows user to store multiple elements of similar data type under a single variable. List provide below advantages:

- No need to use multiple variables to store different data.
- Easy to traverse data in list using loops.
- Easy to sort data stored in list

This practical will help student to practice writing Python scripts using list.

B. Relevant Program Outcomes (POs):

i. **Basic and Discipline specific knowledge (PO1):** Apply knowledge of basic mathematics, science and engineering fundamentals and engineering specialization to solve the Computer Engineering problems.

ii. **Problem analysis (PO2):** Identify and analyse well-defined Computer Engineering problems using codified standard methods.

iii. **Design/development of solutions (PO3):** Design solutions for Computer Engineering well-defined technical problems and assist with the design of systems components or processes to meet specified needs.

iv. **Engineering Tools, Experimentation and Testing (PO4):** Apply modern Computer Engineering tools and appropriate technique to conduct standard tests and measurements.

v. **Project Management (PO6):** Use engineering management principles individually, as a team member or a leader to manage projects and effectively communicate about well-defined engineering activities.

vi. **Life-long learning (PO7):** Ability to analyse individual needs and engage in updating in the context of technological changes in field of engineering.

C. Competency and Practical Skills:

This practical is expected to develop the following skills for the industry-identified competency '**Develop simple applications using scripting language Python.**'

i. Problem analysis skills.

ii. Programming skills.

iii. Debugging skills.

D. Relevant Course Outcomes (COs):

i. Implement data structures lists, tuples, sets and dictionaries to solve the given problems.

E. Practical Outcomes:

i. Write Python scripts to store data in list to solve given problem.

F. Relevant Affective domain Outcomes (ADOs):

i. Maintain tools and equipments.

ii. Follow Coding standards and practices.

iii. Follow ethical practices.

G. Prerequisite Theory:

- A list is a linear data structure, meaning that its elements have a linear ordering. That is, there is a first element, a second element, and so on.
- List in Python is mutable. It means we can add items to list, remove items from list or modify items in list at any point of time.
- List can store elements of same type as well as of different types.
- Each item in the list is identified by its index value. In Python, list index starts from 0.
- Number of elements in the list is called Length of list. As we add/remove elements from the list, length of the list changes. If there are n elements in the list then index in the list starts from 0 and ends at n-1.

<u>Creating List:</u>

```
# Create an empty list l1
l1 = []
print("L1:", l1)

# Create a list l2 with 10 integers
l2 = [55, 89, 98, 68, 21, 7, 132, 76, 49, 36]
print("L2:", l2)

# Create a list l3 containing 7 strings as its items
l3 = ['orange', 'apple', 'pear', 'banana', 'kiwi', 'apple',
'banana']
print("L3:", l3)

# Create a list l4 containg mix data types as its items
l4 = ['orange', 99, 34.50, 'kiwi', 10, 'apple']
print("L4:" l4)
```

Length of a List:

len() function returns the number of elements in the list.

```
l2 = [55, 89, 98, 68, 21, 7, 132, 76, 49, 36]
print(len(l2))
```

Accessing list using Index:

An element in a list can be accessed through the index operator ([]) and its index value.

List Slicing:

The index operator allows you to select an element at the specified index. The slicing operator returns a slice of the list using the syntax list[start : end : stepsize].

```
l = [55, 89, 98, 68, 21, 7, 132, 76, 49, 36]
print("l[2:4]:", l[2:4])      # items from index 2 to 4-1=3
print("l[:5]:", l[:5])        # items from beginning to index 5-1=4
print("l[7:]:", l[7:])        # items from index 7 to last item
print("l[:]:", l[:])          # all the items in the list
print("l[4:-3]:", l[4:-3])    # items from 4 to 10-3-1=6
print("l[-4:-2]:", l[-4:-2])  # items from 10-4=6 to 10-2-1=7
```

Accessing list using while loop:

Using for loop:

```
mylist = [5.6, 4.5, 3.3, 13.2, 4.0, 34.33, 34.0, 99.993, 11123]

print("\nAccessing List using for loop:")
for item in mylist:
      print(item)
```

Using while loop:

```
mylist = [5.6, 4.5, 3.3, 13.2, 4.0, 34.33, 34.0, 99.993, 11123]

print("Accessing List using while loop:")
i = 0
while(i < len(mylist)):
      print(i, ":", mylist[i])
      i = i + 1
```

H. Resources Required:

Sr. No	Instrument /Components	Configuration/Specification
1.	Computer System	Processor: RAM: Operating System:
2.	Python Interpreter	Python Version:
3.	Text Editor	Editor:

I. Source code and Output:

i. Write a program to perform the below operations on the list:
- Create a list.
- Add/Remove an item to/from a list.
- Get the number of elements in the list.

- Access elements of the list using the index.
- Sort the list.
- Reverse the list.

Source Code:

Output:

ii. Write a program to read n numbers from a user and print:

- Number of positive numbers.
- Number of negative numbers.
- Number of zeros.
- Number of odd numbers.
- Number of even numbers.
- Average of all numbers.

Source Code:

Output:

iii. Write a program that counts the occurrences of each digit in a string. The program counts how many times a digit appears in the string. For example, if the input is "12203AB3", then the output should output 0 (1 time), 1 (1 time), 2 (2 times), 3 (2 times).

Source Code:
Output:

iv. Write a program to eliminate duplicate values in the list.

Source Code:

Output:

v. Write a program to randomly fill in 0s and 1s into a 4x4 2-dimension list, print the list and find the rows and columns with the most number of 1s.

Source Code:
Output:

J. Practical related Quiz:

i. How do you access an element from a list in Python?

A) Using parentheses ()

B) Using square brackets []

C) Using curly braces {}

D) Using angle brackets <>

ii. What is the index of the first element in a list?

A) 0

B) 1

C) -1

D) None of the above

iii. What does the append() method do in Python?

A) Adds an element to the beginning of the list

B) Adds an element to the end of the list

C) Removes an element from the list

D) Reverses the order of the elements in the list

iv. How do you find the length of a list in Python?

A) length(list)

B) size(list)

C) len(list)

D) list.length()

v. Which of the following methods can be used to remove an element from a list in Python?

A) delete()

B) remove()

C) pop()

D) All of the above

vi. How do you check if an element exists in a list in Python?

A) using the exists() method

B) using the in keyword

C) using the has_element() method

D) using the contains() keyword

vii. What will be the output of the following code snippet?

```
my_list = [1, 2, 3, 4, 5]
print(my_list[2:4])
```

A) [2, 3, 4]

B) [3, 4]

C) [2, 3]

D) [2, 3, 4, 5]

viii. Which method is used to sort a list in ascending order in Python?

A) sort()

B) sort(ascending=True)

C) sorted()

D) order()

ix. How do you reverse the order of elements in a list in Python?

A) list.reverse()

B) reverse(list)

C) list.reversed()

D) reversed(list)

x. What will be the output of the following code snippet?

```
my_list = [1, 2, 3, 4, 5]
print(my_list[-3])
```

A) 3

B) 2

C) 4

D) 5

xi. How do you concatenate two lists in Python?

A) list.concat(list2)

B) list.append(list2)

C) list.extend(list2)

D) list.add(list2)

xii. What will be the output of the following code snippet?

```
my_list = [1, 2, 3]
print(my_list.pop())
```

a) 1

b) 2

c) 3

d) None

K. References:

i. https://docs.python.org/3/tutorial/datastructures.html
ii. https://www.w3schools.com/python/python_lists.asp
iii. https://www.geeksforgeeks.org/python-lists/
iv. https://www.programiz.com/python-programming/list
v. https://www.simplilearn.com/tutorials/python-tutorial/python-list

L. Assessment Rubrics:

Criteria	% of point	Rubrics	Marks
C1: Program Completeness/ Correctness	50 %	Excellent (10-8 marks): Completed programs/scripts correctly as per the requirements.	
		Adequate (7-6 marks): Completed programs/scripts correctly with approx. 70% requirements.	
		Poor (5-4 marks): Completed programs/scripts correctly with 70% - 50% requirements.	
		Unsatisfactory (0-3 marks): Completed programs/ scripts correctly with less than 50% requirements.	
C2: Readability	25 %	Excellent (10-8 marks): The code is clean, well-organized and very easy to understand.	
		Adequate (7-6 marks): The code is fairly easy to read and understand.	
		Poor (5-4 marks): The code is readable only by someone who knows what it is supposed to be doing.	
		Unsatisfactory (0-3 marks): The code is poorly organized and very difficult to understand.	
C3: Coding Standards/ Documentation	25 %	Excellent (10-8 marks): ● Coding standards are followed in complete code while naming variables/functions/ classes, explaining the purpose of variables/ functions/classes. ● The Complete code is well-documented with comments explaining the code.	
		Adequate (7-6 marks): ● Coding standards are followed in most of the code while naming variables/functions/ classes, explaining the purpose of variables/ functions/classes. ● Most of the code is documented with comments explaining the code.	
		Poor (5-4 marks): ● Coding standards are followed in very little code while naming variables/functions/ classes, explaining the purpose of variables/ functions/classes. ● Very little code is documented with comments explaining the code.	
		Unsatisfactory (0-3 marks): ● Coding standards are not followed properly. ● Proper comments are not written explaining the code.	
Total Marks for Practical = 0.5 * (Marks of C1) + 0.25 * (Marks of C2) + 0.25 * (Marks of C3)			

Signature with Date:

Practical No. 7: **Tuples, Sets and Dictionaries**

i. Write a program to perform below operations on tuple:
   - Create a tuple with different data types.
   - Print tuple items.
   - Convert tuple into a list.
   - Remove data items from a list.
   - Convert list into a tuple.
   - Print tuple items.

ii. Write a program to perform below operations on set:
   - Create two different sets with the data.
   - Print set items.
   - Add/remove items in/from a set.
   - Perform operations on sets: union, intersection, difference, symmetric difference, check subset of another set.

iii. Write a program to perform below operations on dictionary:
   - Create a dictionary.
   - Print dictionary items.
   - Add/remove key-value pair in/from a dictionary.
   - Check whether a key exists in a dictionary.
   - Iterate through a dictionary.
   - Concatenate multiple dictionaries.

iv. Write a program that is given a dictionary containing the average daily temperature for each day of the week, and prints all the days on which the average temperature was between 40 and 50 degrees.

v. Write a program to repeatedly prompt the user to enter the capital of a state. Upon receiving the user's input, the program reports whether the answer is correct. Assume the states and their capitals are stored in dictionaries as key-value pairs

A. Objectives:

Other than List Python provides three inbuilt data types: Tuple, Set and Dictionary. This practical will help student to practice writing Python scripts using these data structures.

B. Relevant Program Outcomes (POs):

i. **Basic and Discipline specific knowledge (PO1):** Apply knowledge of basic mathematics, science and engineering fundamentals and engineering specialization to solve the Computer Engineering problems.

ii. **Problem analysis (PO2):** Identify and analyse well-defined Computer Engineering problems using codified standard methods.

iii. **Design/development of solutions (PO3):** Design solutions for Computer Engineering well-defined technical problems and assist with the design of systems components or processes to meet specified needs.

iv. **Engineering Tools, Experimentation and Testing (PO4):** Apply modern Computer Engineering tools and appropriate technique to conduct standard tests and measurements.

v. **Project Management (PO6):** Use engineering management principles individually, as a team member or a leader to manage projects and effectively communicate about well-defined engineering activities.

vi. **Life-long learning (PO7):** Ability to analyse individual needs and engage in updating in the context of technological changes in field of engineering.

C. Competency and Practical Skills:

This practical is expected to develop the following skills for the industry-identified competency '**Develop simple applications using scripting language Python.**'

i. Problem analysis skills.

ii. Programming skills.

iii. Debugging skills.

D. Relevant Course Outcomes (COs):

i. Implement data structures lists, tuples, sets and dictionaries to solve the given problems.

E. Practical Outcomes:

i. Write Python scripts to perform operation on Tuples, Sets and Dictionaries.

F. Relevant Affective domain Outcomes (ADOs):

i. Maintain tools and equipments.

ii. Follow Coding standards and practices.

iii. Follow ethical practices.

G. Prerequisite Theory:

Tuples:

- Tuples are like lists, but their elements are fixed; that is, once a tuple is created, you cannot add new elements, delete elements, replace elements, or reorder the elements in the tuple. Hence, Tuples are Immutable objects.

- If the contents of a list in your application shouldn't change, you can use a tuple to prevent elements from being added, deleted, or replaced accidentally.
- Furthermore, tuples are more efficient than lists due to Python's implementations.

```
# create an empty tuple
t1 = ()
print("t1:", t1)

# create a tuple with 4 elements
t2 = (1, 3, 5, 7)
print("t2:", t2)

# create a tuple from list
t3 = tuple([1, 2, 3, 4,5])
print("t3:", t3)

# create a tuple from list variable
l = [2, 4, 6, 8, 10]
t4 = tuple(l)
print("t4:", t4)
```

Sets:

- Sets are like lists in that you use them for storing a collection of elements.
- Unlike lists, however, the elements in a set are non-duplicates and are not placed in any particular order.
- If your application does not care about the order of the elements, using a set to store elements is more efficient than using lists due to Python's implementations.

```
# create an empty set
s1 = set()
print("s1:", s1)

# create a set with three elements
s2 = {1, 3, 5}
print("s2:", s2)

# create a set from list, removes duplicate items
s3 = set([1, 3, 5, 7, 9, 7, 1])
print("s3:", s3)
```

Dictionaries:

- A dictionary is a container object that stores a collection of key/value pairs.
- It enables fast retrieval, deletion, and updating of the value by using the key.
- A dictionary is a collection that stores the values along with the keys. The keys are like an index operator.
- In a list, the indexes are integers. In a dictionary, the key must be a hash table object.
- A dictionary cannot contain duplicate keys. Each key maps to one value.

```
# Create an empty dictionary
dict1 = {}
print("Dict1:", dict1)

# create a dictionary with 2 key:value pairs
dict2 = {"111-34-3434":"John", "132-56-6290":"Peter"}
print("Dict2:", dict2)
```

Add, modify, retrieve and delete value from dictionary.

```
# define customer dictionary with 3 key:value pairs
customer = { "name": "John", "custid": "1234", "country":
"USA"}
print("customer dictionary:", customer)

# add new key:value pair to customer dictionary
customer["contactno"] = "123456789"
print("customer dictionary after adding contactno:", customer)

# Modify value for the key: country
customer["country"] = "UK"
print("customer dictionary after changing country:", customer)

print("Name of the customer:", customer["name"], ", Customer
id:", customer["custid"])

del customer['country']
print("customer dictionary after deleting country:", customer)
```

Looping through items in dictionary

```
customer = {'name': 'John', 'custid': '1234', 'country': 'UK',
'contactno': '123456789'}

for key in customer:
     print(key, ":", customer[key])
```

H. Resources Required:

Sr. No	Instrument /Components	Configuration/Specification
1.	Computer System	Processor: RAM: Operating System:
2.	Python Interpreter	Python Version:
3.	Text Editor	Editor:

I. Source code:

i. Write a program to perform below operations on tuple:

- Create a tuple with different data types.
- Print tuple items.
- Convert tuple into a list.
- Remove data items from a list.
- Convert list into a tuple.
- Print tuple items.

Source Code:

Output:

ii. Write a program to perform below operations on set:
   - Create two different sets with the data.
   - Print set items.
   - Add/remove items in/from a set.

- Perform operations on sets: union, intersection, difference, symmetric difference, check subset of another set.

Source Code:

Output:

iii. Write a program to perform below operations on dictionary:

- Create a dictionary.
- Print dictionary items.
- Add/remove key-value pair in/from a dictionary.
- Check whether a key exists in a dictionary.
- Iterate through a dictionary.
- Concatenate multiple dictionaries.

Source Code:

Output:

iv. Write a program that is given a dictionary containing the average daily temperature for each day of the week, and prints all the days on which the average temperature was between 40 and 50 degrees.

Source Code:

Output:

v. Write a program to repeatedly prompt the user to enter the capital of a state. Upon receiving the user's input, the program reports whether the answer is correct. Assume the states and their capitals are stored in dictionaries as key-value pairs

Source Code:

Output:

J. Practical related Quiz:

i. Which data structure in Python is immutable and ordered?

   A) Tuple

   B) Set

   C) Dictionary

   D) List

ii. How do you create an empty tuple in Python?

   A) ()

   B) []

   C) {}

   D) None

iii. Which method is used to count the occurrences of an element in a tuple?

   A) count()

   B) index()

   C) len()

   D) size()

iv. What is the key difference between a tuple and a list in Python?

   A) Tuples are immutable, while lists are mutable.

   B) Tuples can store elements of different data types, while lists can only store elements of the same data type.

   C) Tuples are unordered, while lists are ordered.

D) Tuples can be resized, while lists have a fixed size.

v. Which of the following is true about Python tuples?

A) Tuples are mutable

B) Tuples are ordered

C) Tuples can contain duplicate elements

D) Tuples can be resized

vi. Which data structure in Python stores unique and unordered elements?

A) Tuple

B) Set

C) Dictionary

D) List

vii. Which method is used to add elements to a set in Python?

A) add()

B) insert()

C) append()

D) extend()

viii. Which method is used to remove an element from a set in Python?

A) remove()

B) delete()

C) discard()

D) pop()

ix. How do you create an empty set in Python?

A) ()

B) []

C) {}

D) set()

x. What is the key property of sets that makes them useful for eliminating duplicates?

A) Sets are ordered.

B) Sets are immutable.

C) Sets are mutable.

D) Sets only store unique elements.

xi. How do you create an empty dictionary in Python?

A) ()

B) []

C) {}

D) dict()

xii. How do you add a new key-value pair to a dictionary in Python?

A) dictionary.add(key, value)

B) dictionary[key] = value

C) dictionary.insert(key, value)

D) dictionary.append(key, value)

xiii. What happens if you try to access a key that doesn't exist in a dictionary?

A) A KeyError is raised.

B) The program throws an error.

C) The value None is returned.

D) The dictionary is resized to include the new key.

xiv. How do you retrieve all the keys from a dictionary in Python?

A) dictionary.get_keys()

B) dictionary.keys()

C) dictionary.retrieve_keys()

D) dictionary.all_keys()

xv. How do you retrieve both keys and values from a dictionary in Python?

A) dictionary.items()

B) dictionary.get_items()

C) dictionary.all_items()

D) dictionary.retrieve_items()

xvi. Which method is used to clear all the elements from a dictionary in Python?

A) clear()

B) delete_all()

C) remove_all()

D) discard_all()

K. References:

i. https://www.w3schools.com/python/python_tuples.asp
ii. https://www.geeksforgeeks.org/tuples-in-python/
iii. https://www.programiz.com/python-programming/tuple
iv. https://www.w3schools.com/python/python_sets.asp
v. https://www.geeksforgeeks.org/sets-in-python/
vi. https://www.programiz.com/python-programming/set
vii. https://www.w3schools.com/python/python_dictionaries.asp
viii. https://www.geeksforgeeks.org/python-dictionary/
ix. https://www.programiz.com/python-programming/dictionary

L. Assessment Rubrics:

Criteria	% of point	Rubrics	Marks
C1: Program Completeness/ Correctness	50 %	Excellent (10-8 marks): Completed programs/scripts correctly as per the requirements.	
		Adequate (7-6 marks): Completed programs/scripts correctly with approx. 70% requirements.	
		Poor (5-4 marks): Completed programs/scripts correctly with 70% - 50% requirements.	
		Unsatisfactory (0-3 marks): Completed programs/ scripts correctly with less than 50% requirements.	
C2: Readability	25 %	Excellent (10-8 marks): The code is clean, well-organized and very easy to understand.	
		Adequate (7-6 marks): The code is fairly easy to read and understand.	
		Poor (5-4 marks): The code is readable only by someone who knows what it is supposed to be doing.	
		Unsatisfactory (0-3 marks): The code is poorly organized and very difficult to understand.	
C3: Coding Standards/ Documentation	25 %	Excellent (10-8 marks): ● Coding standards are followed in complete code while naming variables/functions/ classes, explaining the purpose of variables/ functions/classes. ● The Complete code is well-documented with comments explaining the code.	
		Adequate (7-6 marks): ● Coding standards are followed in most of the code while naming variables/functions/ classes, explaining the purpose of variables/ functions/classes. ● Most of the code is documented with comments explaining the code.	
		Poor (5-4 marks): ● Coding standards are followed in very little code while naming variables/functions/ classes, explaining the purpose of variables/ functions/classes. ● Very little code is documented with comments explaining the code.	
		Unsatisfactory (0-3 marks): ● Coding standards are not followed properly. ● Proper comments are not written explaining the code.	
Total Marks for Practical = 0.5 * (Marks of C1) + 0.25 * (Marks of C2) + 0.25 * (Marks of C3)			

Signature with Date:

Practical No. 8: Function

i. Write a program that defines a function (shuffle) to scramble a list into a random order, like shuffling a deck of cards.

ii. Write a program that defines a function to return a new list by eliminating the duplicate values in the list.

iii. Write a program to print Fibonacci sequence up to n numbers using recursion. Fibonacci sequence is defined as below:

$Fibonacci\ Sequence$= 1 1 2 3 5 8 13 21...

$$where\ n^{th} term\ x_n = x_{n-1} + x_{n-2}$$

iv. Write a program that defines a function to determine whether input number *n* is prime or not. A positive whole number *n* > 2 is prime, if no number between 2 and $\sqrt{n}$ (inclusive) evenly divides *n*. If *n* is not prime, the program should quit as soon as it finds a value that evenly divides *n*.

v. Write a program that defines a function to find the GCD of two numbers using the algorithm below. The greatest common divisor (GCD) of two values can be computed using Euclid's algorithm. Starting with the values m and n, we repeatedly apply the formula: n, m = m, n%m until m is 0. At that point, n is the GCD of the original m and n (Use Recursion).

vi. Write a program that lets the user enter the loan amount, number of years, and interest rate, and defines a function to calculate monthly EMI, total payment and display the amortization schedule for the loan.

A. Objectives:

A function is a block of reusable code that is used to perform a specific action. Functions provide below advantages:

- Reduce duplication of the code.
- Modularisation of the code.
- Improve clarity of the code.
- Information hiding.

This practical will help student to practice writing Python scripts using user defined functions and in-built functions.

B. Relevant Program Outcomes (POs):

i. **Basic and Discipline specific knowledge (PO1):** Apply knowledge of basic mathematics, science and engineering fundamentals and engineering specialization to solve the Computer Engineering problems.

ii. Problem analysis (PO2): Identify and analyse well-defined Computer Engineering problems using codified standard methods.

iii. **Design/development of solutions (PO3):** Design solutions for Computer Engineering well-defined technical problems and assist with the design of systems components or processes to meet specified needs.

iv. **Engineering Tools, Experimentation and Testing (PO4):** Apply modern Computer Engineering tools and appropriate technique to conduct standard tests and measurements.

v. **Project Management (PO6):** Use engineering management principles individually, as a team member or a leader to manage projects and effectively communicate about well-defined engineering activities.

vi. **Life-long learning (PO7):** Ability to analyse individual needs and engage in updating in the context of technological changes in field of engineering.

C. Competency and Practical Skills:

This practical is expected to develop the following skills for the industry-identified competency '**Develop simple applications using scripting language Python.**'

i. Problem analysis skills.

ii. Programming skills.

iii. Debugging skills.

D. Relevant Course Outcomes (COs):

i. Apply modular programming approach to solve the given problems using user-defined functions.

E. Practical Outcomes:

i. Write Python scripts to develop user-defined functions to solve given problem.

F. Relevant Affective domain Outcomes (ADOs):

i. Maintain tools and equipments.

ii. Follow Coding standards and practices.

iii. Follow ethical practices.

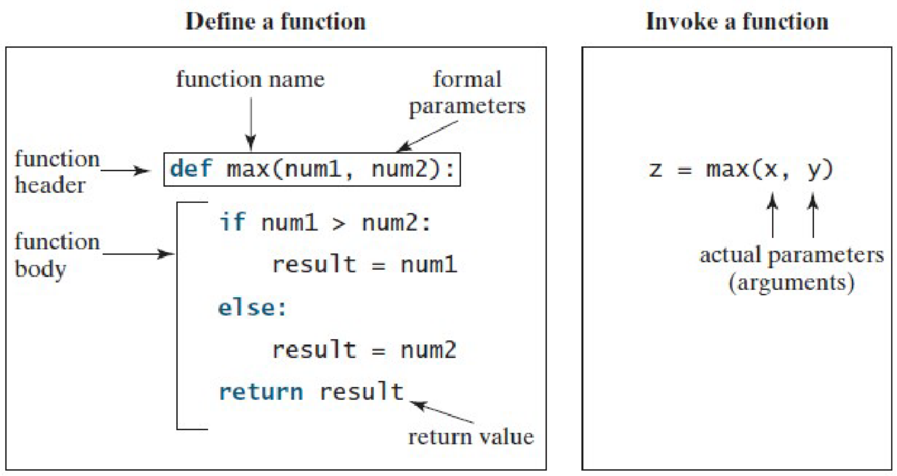
G. Prerequisite Theory:

- Functions can be used to define reusable code and organize and simplify code.
- A Function is a named group of instructions performing some task. A Function can be invoked (called) as many times as needed in a given program.

Defining a Function

```
def functionName(List of parameters):
      # Function body
```

A function definition consists of the function's header and body:



```
def max(num1, num2):
    if num1 > num2:
        result = num1
    else:
        result = num2
    return result
```

```
z = max(x, y)
```

- The function header begins with the def keyword, followed by the function's name and parameters (surrounded by parentheses), and ends with a colon.
- The variables in the function header are known as formal parameters or simply parameters.
- A parameter is like a placeholder: When a function is invoked, you pass a value to the parameter. This value is referred to as an actual parameter or argument.
- Parameters are optional: a function may or may not have any parameters.
  - For example, the random.random() function has no parameters.
  - In above example max(num1, num2) function has 2 parameters: num1 and num2.
- The function body contains a collection of statements that define what the function does.
- Some functions return a value, while other functions perform desired operations without returning a value. If a function returns a value, it is called a value-returning function.
- For example, the function body of the max function uses anif statement to determine which number is larger and return the value of that number.
- A return statement using the keyword return is required for a value-returning function to return a result.
- The function terminates when a return statement is executed.

Calling a Function

Calling a function executes the code in the function.

```
larger = max(3, 4)
```

calls max(3, 4) and assigns the result of the function to the variable larger.

Passing Parameter to a function

When you invoke a function with arguments, each argument's reference is passed by value to the parameter in the function.

```
def main():
      x = 1
      print("Before the call x:",x)
      increment(x)
      print("After the call x:",x)

def increment(n):
      print("\tInside the function before increment n:",n)
      n = n + 1
      print("\tInside the function after the increment n:",n)

main()
# Call the main function
```

H. Resources Required:

Sr. No	Instrument /Components	Configuration/Specification
1.	Computer System	Processor: RAM: Operating System:
2.	Python Interpreter	Python Version:
3.	Text Editor	Editor:

I. Source code and Output:

i. Write a program that defines a function (shuffle) to scramble a list into a random order, like shuffling a deck of cards.

Source Code:

Output:

ii. Write a program that defines a function to return a new list by eliminating the duplicate values in the list.

Source Code:

Output:

iii. Write a program to print Fibonacci sequence up to n numbers using recursion. Fibonacci sequence is defined as below:

$Fibonacci\ Sequence$= 1 1 2 3 5 8 13 21…

$$where\ n^{th} term\ x_n = x_{n-1} + x_{n-2}$$

Source Code:

Output:

iv. Write a program that defines a function to determine whether input number *n* is prime or not. A positive whole number *n* > 2 is prime, if no number between 2 and $\sqrt{n}$ (inclusive) evenly divides *n*. If *n* is not prime, the program should quit as soon as it finds a value that evenly divides *n*.

Source Code:

Output:

v. Write a program that defines a function to find the GCD of two numbers using the algorithm below. The greatest common divisor (GCD) of two values can be computed using Euclid's algorithm. Starting with the values m and n, we repeatedly apply the formula: n, m = m, n%m until m is 0. At that point, n is the GCD of the original m and n (Use Recursion).

Source Code:

Output:

vi. Write a program that lets the user enter the loan amount, number of years, and interest rate, and defines a function to calculate monthly EMI, total payment and display the amortization schedule for the loan.

Source Code:

Output:

J. Practical related Quiz:

i. How do you define a function in Python?

A) def my_function()

B) function my_function()

C) define my_function()

D) def my_function():

ii. What is the purpose of the return statement in a function?

A) It terminates the function and returns a value to the caller.

B) It restarts the function from the beginning.

C) It prints the output of the function.

D) It defines the function's parameters.

iii. What is the difference between parameters and arguments in a function?

A) Parameters are specified when defining a function, while arguments are passed when calling a function.

B) Parameters are passed when calling a function, while arguments are specified when defining a function.

C) Parameters and arguments are the same and can be used interchangeably.

D) Parameters and arguments are not used in Python functions.

iv. How do you pass parameters to a function in Python?

A) By using the keyword "param"

B) By using square brackets []

C) By specifying the parameters in the function call

D) Parameters are not allowed in Python functions

v. How do you call a function with named arguments in Python?

A) my_function(value1, value2)

B) my_function(arg1=value1, arg2=value2)

C) my_function(arg1, arg2)

D) my_function(arg1=value1, value2)

vi. What is a recursive function in Python?

A) A function that calls itself during its execution.

B) A function that accepts keyword arguments.

C) A function that returns multiple values.

D) A function that does not have a return statement.

vii. What is the purpose of the *args parameter in a function definition?

A) It allows the function to accept an arbitrary number of arguments.

B) It specifies default values for the function's arguments.

C) It defines keyword arguments for the function.

D) It is not a valid parameter in Python.

viii. What is a default parameter in a function?

A) A parameter that is automatically assigned the value "default"

B) A parameter that must always be passed by the caller

C) A parameter with a pre-defined value that can be overridden by the caller

D) A parameter that is not allowed in Python functions

K. References:

i. https://www.learnpython.org/en/Functions

ii. https://www.w3schools.com/python/python_functions.asp
iii. https://www.geeksforgeeks.org/python-functions/
iv. https://www.programiz.com/python-programming/function
v. https://www.tutorialspoint.com/python/python_functions.htm
vi. https://docs.python.org/2/library/functions.html
vii.

L. Assessment Rubrics:

Criteria	% of point	Rubrics	Marks
C1: Program Completeness/ Correctness	50 %	Excellent (10-8 marks): Completed programs/scripts correctly as per the requirements.	
		Adequate (7-6 marks): Completed programs/scripts correctly with approx. 70% requirements.	
		Poor (5-4 marks): Completed programs/scripts correctly with 70% - 50% requirements.	
		Unsatisfactory (0-3 marks): Completed programs/ scripts correctly with less than 50% requirements.	
C2: Readability	25 %	Excellent (10-8 marks): The code is clean, well-organized and very easy to understand.	
		Adequate (7-6 marks): The code is fairly easy to read and understand.	
		Poor (5-4 marks): The code is readable only by someone who knows what it is supposed to be doing.	
		Unsatisfactory (0-3 marks): The code is poorly organized and very difficult to understand.	
C3: Coding Standards/ Documentation	25 %	Excellent (10-8 marks): ● Coding standards are followed in complete code while naming variables/functions/ classes, explaining the purpose of variables/ functions/classes. ● The Complete code is well-documented with comments explaining the code.	
		Adequate (7-6 marks): ● Coding standards are followed in most of the code while naming variables/functions/ classes, explaining the purpose of variables/ functions/classes. ● Most of the code is documented with comments explaining the code.	
		Poor (5-4 marks): ● Coding standards are followed in very little code while naming variables/functions/ classes, explaining the purpose of variables/ functions/classes. ● Very little code is documented with comments explaining the code.	
		Unsatisfactory (0-3 marks): ● Coding standards are not followed properly. ● Proper comments are not written explaining the code.	
Total Marks for Practical = 0.5 * (Marks of C1) + 0.25 * (Marks of C2) + 0.25 * (Marks of C3)			

Signature with Date:

Practical No. 9: Modules

i. Write a program that defines functions (mean and deviation), that computes mean and standard deviation of given numbers. The formula for the mean and standard deviation of n numbers is given as:

$$mean = \sum_{i=1}^{n} x_i = \frac{x_1 + x_2 + \ldots + x_n}{n}$$

$$deviation = \sqrt{\frac{\sum_{i=1}^{n} (x_i - mean)^2}{n-1}}$$

ii. Write a program that plays the popular scissor-rock-paper game. (A scissor can cut a paper, a rock can knock a scissor, and a paper can wrap a rock.) The program randomly generates a number 0, 1, or 2 representing scissor, rock, and paper. The program prompts the user to enter a number 0, 1, or 2 and displays a message indicating whether the user or the computer wins, loses, or draws.

iii. Write a program to print the dates of all the Sundays in a given year.

iv. Write a program to display a graph for ReLU (Rectified Linear Unit) function. ReLU function is defined as below:

$$y = max\,(0,\ x)$$

Consider the range of *x* from -5 to 5.

v. Write a program to create a list representing the results of 100 students in a test, where each element represents a student's marks (between 0 to 10), and display a histogram for the result.

vi. Create a user defined module with simple functions for: addition, subtraction, multiplication, division, modulo, square, factorial. Write a program to import the module and access functions defined in the module.

A. Objectives:

Python provides various in-built modules to be used in program. Use can also create user-defined modules as per the application requirement. This practical will help students to practice use of in-built modules and use-defined modules.

B. Relevant Program Outcomes (POs):

i. **Basic and Discipline specific knowledge (PO1):** Apply knowledge of basic mathematics, science and engineering fundamentals and engineering specialization to solve the Computer Engineering problems.

ii. Problem analysis (PO2): Identify and analyse well-defined Computer Engineering problems using codified standard methods.

iii. Design/development of solutions (PO3): Design solutions for Computer Engineering well-defined technical problems and assist with the design of systems components or processes to meet specified needs.

iv. **Engineering Tools, Experimentation and Testing (PO4):** Apply modern Computer Engineering tools and appropriate technique to conduct standard tests and measurements.

v. **Project Management (PO6):** Use engineering management principles individually, as a team member or a leader to manage projects and effectively communicate about well-defined engineering activities.

vi. **Life-long learning (PO7):** Ability to analyse individual needs and engage in updating in the context of technological changes in field of engineering.

C. Competency and Practical Skills:

This practical is expected to develop the following skills for the industry-identified competency '**Develop simple applications using scripting language Python.**'

i. Problem analysis skills.

ii. Programming skills.

iii. Debugging skills.

D. Relevant Course Outcomes (COs):

i. Apply modular programming approach to solve the given problems using user-defined functions.

E. Practical Outcomes:

i. Write Python scripts to use in-built modules to solve given problem.

F. Relevant Affective domain Outcomes (ADOs):

i. Maintain tools and equipments.

ii. Follow Coding standards and practices.

iii. Follow ethical practices.

G. Prerequisite Theory:

Python module is a file containing a set of functions that you can include in your application. A module can define functions, classes, and variables. Grouping related code into a module makes the code easier to understand and use. It also makes the code logically organized.

User can define their own modules and import them in the source code or they can import already available modules.

Create a module:

Create a simple calc.py in which we define two functions, one add and another subtract.

```
# A simple module, calc.py
def add(x, y):
    return (x+y)

def subtract(x, y):
    return (x-y)
```

Import a module:

We can import the functions, and classes defined in a module to another module using the import statement in some other Python source file.

```
# importing module calc.py
import calc

print(calc.add(10, 2))
```

Python’s from statement lets you import specific attributes from a module without importing the module as a whole.

```
# importing sqrt() and factorial from the module math
from math import sqrt, factorial

# if we simply do "import math", then math.sqrt(16) and
# math.factorial() are required.
print(sqrt(16))
print(factorial(6))
```

Some of the Python modules:

- **random:** *random* module is an in-built module of Python that is used to generate random numbers
- **math:** *math* module provides set of methods and constants for mathematical tasks.
- **datetime:** *datetime* module provides classes and functions to work with date and time.
- **matplotlib:** *matplotlib* is a comprehensive library for creating static, animated, and interactive visualizations in Python.

H. Resources Required:

Sr. No	Instrument /Components	Configuration/Specification
1.	Computer System	Processor: RAM: Operating System:

2.	Python Interpreter	Python Version:
3.	Text Editor	Editor:

I. Source code and Output:

i. Write a program that defines functions (mean and deviation), that computes mean and standard deviation of given numbers. The formula for the mean and standard deviation of n numbers is given as:

$$mean = \sum_{i=1}^{n} x_i = \frac{x_1 + x_2 + ... + x_n}{n}$$

$$deviation = \sqrt{\frac{\sum_{i=1}^{n} (x_i - mean)^2}{n-1}}$$

Source Code:
Output:

ii. Write a program that plays the popular scissor-rock-paper game. (A scissor can cut a paper, a rock can knock a scissor, and a paper can wrap a rock.) The program randomly generates a number 0, 1, or 2 representing scissor, rock, and paper. The program prompts the user to enter a number 0, 1, or 2 and displays a message indicating whether the user or the computer wins, loses, or draws.

Source Code:

Output:

iii. Write a program to print the dates of all the Sundays in a given year.

Source Code:

Output:

iv. Write a program to display a graph for ReLU (Rectified Linear Unit) function. ReLU function is defined as below:

$$y = max\,(0,\ x)$$

Consider the range of *x* from -5 to 5.

Source Code:

Output:

v. Write a program to create a list representing the results of 100 students in a test, where each element represents a student's marks (between 0 to 10), and display a histogram for the result.

Source Code:

Output:

vi. Create a user defined module with simple functions for: addition, subtraction, multiplication, division, modulo, square, factorial. Write a program to import the module and access functions defined in the module.

Source Code:

Output:

J. Practical related Quiz:

i. What is a Python module?

   A) A built-in function

   B) A file containing Python code

   C) A data type

   D) A software development framework

ii. Which keyword is used to import a module in Python?

   A) use

   B) load

   C) import

   D) include

iii. What is the purpose of the math module in Python?

   A) To perform mathematical operations

   B) To handle dates and times

   C) To manipulate strings

   D) To interact with the operating system

iv. Which module is used for working with dates and times in Python?

   a) datetime

   b) math

   c) time

   d) calendar

v. How can you access a function defined in a module named my_module?

   A) my_module.function_name()

B) function_name.my_module()

C) my_module::function_name()

D) function_name.my_module

vi. What is the purpose of the random module in Python?

A) To generate random numbers

B) To sort lists

C) To handle network connections

D) To parse XML data

vii. What is the purpose of the os module in Python?

A) To interact with the operating system

B) To handle exceptions

C) To manipulate strings

D) To perform mathematical operations

viii. Which module is used for working with databases in Python?

A) db

B) sqlite

C) sql

D) database

ix. How can you create an alias for a module during import?

A) import module_name as alias

B) import module_name alias

C) import module_name = alias

D) module_name = alias

x. Which module is commonly used for sending HTTP requests in Python?

A) requests

B) urllib

C) http

D) http.client

xi. To handle command-line arguments, which module is used in Python?

A) argparse

B) cmdargs

C) args

D) sysargs

K. References:

i. https://docs.python.org/3/tutorial/modules.html
ii. https://www.w3schools.com/python/python_modules.asp
iii. https://www.geeksforgeeks.org/python-modules/
iv. https://www.w3schools.com/python/module_random.asp
v. https://www.w3schools.com/python/module_math.asp
vi. https://www.geeksforgeeks.org/python-datetime-module/
vii. https://www.geeksforgeeks.org/python-introduction-matplotlib/

L. Assessment Rubrics:

Criteria	% of point	Rubrics	Marks
C1: Program Completeness/ Correctness	50 %	Excellent (10-8 marks): Completed programs/scripts correctly as per the requirements.	
		Adequate (7-6 marks): Completed programs/scripts correctly with approx. 70% requirements.	
		Poor (5-4 marks): Completed programs/scripts correctly with 70% - 50% requirements.	
		Unsatisfactory (0-3 marks): Completed programs/ scripts correctly with less than 50% requirements.	
C2: Readability	25 %	Excellent (10-8 marks): The code is clean, well-organized and very easy to understand.	
		Adequate (7-6 marks): The code is fairly easy to read and understand.	
		Poor (5-4 marks): The code is readable only by someone who knows what it is supposed to be doing.	
		Unsatisfactory (0-3 marks): The code is poorly organized and very difficult to understand.	
C3: Coding Standards/ Documentation	25 %	Excellent (10-8 marks): ● Coding standards are followed in complete code while naming variables/functions/ classes, explaining the purpose of variables/ functions/classes. ● The Complete code is well-documented with comments explaining the code.	
		Adequate (7-6 marks): ● Coding standards are followed in most of the code while naming variables/functions/ classes, explaining the purpose of variables/ functions/classes. ● Most of the code is documented with comments explaining the code.	
		Poor (5-4 marks): ● Coding standards are followed in very little code while naming variables/functions/ classes, explaining the purpose of variables/ functions/classes. ● Very little code is documented with comments explaining the code.	
		Unsatisfactory (0-3 marks): ● Coding standards are not followed properly. ● Proper comments are not written explaining the code.	
Total Marks for Practical = 0.5 * (Marks of C1) + 0.25 * (Marks of C2) + 0.25 * (Marks of C3)			
Signature with Date:			

Practical No. 10: **String Processing**

i. Write a program to check whether a given string is palindrome or not.

ii. Write a program to read a string containing letters, each of which may be in either uppercase or lowercase, and return a tuple containing the number of vowels and consonants in the string.

iii. Write a program to read a date in the format DD/MM/YYYY and print the same date in MM-DD-YYYY format.

iv. Write a program that checks whether two words are anagrams.

v. Two words are anagrams if they contain the same letters. For example, silent and listen are anagrams.

vi. Write a program that allows users to enter six-digit RGB color codes and converts them into base 10. In this format, the first two hexadecimal digits represent the amount of red, the second two the amount of green, and the last two the amount of blue. For example: If a user enters FF6347, then the output should be Red (255), Green (99) and Blue (71).

vii. Numerologists claim to be able to determine a person's character traits based on the "numeric value" of a name. The value of a name is determined by summing up the values of the letters of the name, where "a" is 1 "b" is 2 "c" is 3 and so on up to "z" being 26. For example, the name "Python" would have the value 16 + 25 + 20 + 8 + 15 + 14 = 98. Write a program that calculates the numeric value of a name provided as input.

A. Objectives:

This practical will help students to practise sting processing using various string processing functions.

B. Relevant Program Outcomes (POs):

i. **Basic and Discipline specific knowledge (PO1):** Apply knowledge of basic mathematics, science and engineering fundamentals and engineering specialization to solve the Computer Engineering problems.

ii. **Problem analysis (PO2):** Identify and analyse well-defined Computer Engineering problems using codified standard methods.

iii. **Design/development of solutions (PO3):** Design solutions for Computer Engineering well-defined technical problems and assist with the design of systems components or processes to meet specified needs.

iv. **Engineering Tools, Experimentation and Testing (PO4):** Apply modern Computer Engineering tools and appropriate technique to conduct standard tests and measurements.

v. **Project Management (PO6):** Use engineering management principles individually, as a team member or a leader to manage projects and effectively communicate about well-defined engineering activities.

vi. **Life-long learning (PO7):** Ability to analyse individual needs and engage in updating in the context of technological changes in field of engineering.

C. Competency and Practical Skills:

This practical is expected to develop the following skills for the industry-identified competency '**Develop simple applications using scripting language Python.**'

i. Problem analysis skills.

ii. Programming skills.

iii. Debugging skills.

D. Relevant Course Outcomes (COs):

i. Perform string manipulation and file operations to solve the given problems.

E. Practical Outcomes:

i. Write Python scripts using string processing functions.

F. Relevant Affective domain Outcomes (ADOs):

i. Maintain tools and equipments.

ii. Follow Coding standards and practices.

iii. Follow ethical practices.

G. Prerequisite Theory:

Python has a set of built-in methods that you can use on strings.

- string_name.capitalize()
    - Converts the first character of the string to a capital (uppercase) letter, while making all other characters in the string lowercase letters.
- string_name.count(substring, start=…, end=…)
    - Returns the number of occurrences of a substring in the given string.
- string_name.find(sub, start, end)
    - Returns the lowest index or first occurrence of the substring if it is found in a given string. If it is not found, then it returns -1.
- string_name.isalnum()
    - Checks whether all the characters in a given string are either alphabet or numeric (alphanumeric) characters.
- string_name.isalpha()
    - Check whether all characters in the String are an alphabet.
- string_name.isdecimal()
    - Returns true if all characters in a string are decimal, else it returns False.

- string_name.isdigit()
    - Returns “True” if all characters in the string are digits, Otherwise, It returns “False”.
- string_name.isidentifier()
    - Checks whether a string is a valid identifier or not. The method returns True if the string is a valid identifier, else returns False.
- string_name.islower()
    - Returns True if all alphabets in a string are lowercase alphabets. If the string contains at least one uppercase alphabet, it returns False.
- string_name.isupper()
    - Returns True if all alphabets in a string are uppercase alphabets. If the string contains at least one lowercase alphabet, it returns False.
- string_name.isspace()
    - Returns “True” if all characters in the string are whitespace characters, Otherwise, It returns “False”.
- string_name.isnumeric()
    - Returns “True” if all characters in the string are numeric characters, otherwise returns “False”.
- string_name.join(iterable)
    - Joins elements of the sequence separated by a string separator. This function joins elements of a sequence and makes it a string.
- string_name.lower()
    - Converts all uppercase characters in a string into lowercase characters and returns it.
- String_name.upper()
    - Converts all lowercase characters in a string into uppercase characters and returns it.
- string_name.swapcase()
    - converts all uppercase characters to lowercase and vice versa of the given string and returns it.
- string_name.rstrip([chars])
    - Returns a copy of the string with trailing characters removed (based on the string argument passed). If no argument is passed, it removes trailing spaces.

H. Resources Required:

Sr. No	Instrument /Components	Configuration/Specification
1.	Computer System	Processor: RAM: Operating System:
2.	Python Interpreter	Python Version:
3.	Text Editor	Editor:

I. Source code and Output:

i. Write a program to check whether a given string is palindrome or not.

Source Code:
Output:

ii. Write a program to read a string containing letters, each of which may be in either uppercase or lowercase, and return a tuple containing the number of vowels and consonants in the string.

Source Code:

Output:

iii. Write a program to read a date in the format DD/MM/YYYY and print the same date in MM-DD-YYYY format.

Source Code:

Output:

iv. Write a program that checks whether two words are anagrams.

Source Code:

Output:

v. Two words are anagrams if they contain the same letters. For example, silent and listen are anagrams.

Source Code:

Output:

vi. Write a program that allows users to enter six-digit RGB color codes and converts them into base 10. In this format, the first two hexadecimal digits represent the amount of red, the second two the amount of green, and the last two the amount of blue. For example: If a user enters FF6347, then the output should be Red (255), Green (99) and Blue (71).

Source Code:

Output:

vii. Numerologists claim to be able to determine a person's character traits based on the "numeric value" of a name. The value of a name is determined by summing up the values of the letters of the name, where "a" is 1 "b" is 2 "c" is 3 and so on up to "z" being 26. For example, the name "Python" would have the value 16 + 25 + 20 + 8 + 15 + 14 = 98. Write a program that calculates the numeric value of a name provided as input.

Source Code:

Output:

J. Practical related Quiz:

i. Which method is used to convert a string to lowercase in Python?

A) str.lower()

B) str.upper()

C) str.capitalize()

D) str.swapcase()

ii. Which method is used to split a string into a list of substrings based on a specified delimiter?

A) str.split()

B) str.join()

C) str.replace()

D) str.partition()

iii. Which method is used to find the index of the first occurrence of a substring within a string?

A) str.find()

B) str.index()

C) str.count()

D) str.startswith()

iv. Which method is used to check if a string starts with a specified substring?

A) str.startswith()

B) str.endswith()

C) str.islower()

D) str.isupper()

v. Which method is used to remove leading and trailing whitespace characters from a string?

A) str.strip()

B) str.lstrip()

C) str.rstrip()

D) str.replace()

vi. What is the output of the following code snippet?

```
text = "Python"
print(text.islower())
```

A) True

B) False

C) Error: unsupported method

D) Error: missing closing parenthesis

vii. Which method is used to check if a string contains only numeric characters?

A) str.isalpha()

B) str.isdigit()

C) str.islower()

D) str.isupper()

viii. What is the output of the following code snippet?

```
text = "python is awesome"
print(text.capitalize())
```

a) "Python is awesome"

b) "python is awesome"

c) "Python Is Awesome"

d) "PYTHON IS AWESOME"

ix. What is the output of the following code snippet?

```
text = "Hello, World!"
print(len(text))
```

a) 13

b) 12

c) 11

d) 10

K. References:

i. https://www.w3schools.com/python/python_ref_string.asp

ii. https://www.geeksforgeeks.org/python-string-methods/
iii. https://www.javatpoint.com/python-strings
iv. https://www.programiz.com/python-programming/methods/string

L. Assessment Rubrics:

Criteria	% of point	Rubrics	Marks
C1: Program Completeness/ Correctness	50 %	<u>Excellent (10-8 marks):</u> Completed programs/scripts correctly as per the requirements.	
		<u>Adequate (7-6 marks):</u> Completed programs/scripts correctly with approx. 70% requirements.	
		<u>Poor (5-4 marks):</u> Completed programs/scripts correctly with 70% - 50% requirements.	
		<u>Unsatisfactory (0-3 marks):</u> Completed programs/ scripts correctly with less than 50% requirements.	
C2: Readability	25 %	<u>Excellent (10-8 marks):</u> The code is clean, well-organized and very easy to understand.	
		<u>Adequate (7-6 marks):</u> The code is fairly easy to read and understand.	
		<u>Poor (5-4 marks):</u> The code is readable only by someone who knows what it is supposed to be doing.	
		<u>Unsatisfactory (0-3 marks):</u> The code is poorly organized and very difficult to understand.	
C3: Coding Standards/ Documentation	25 %	<u>Excellent (10-8 marks):</u> ● Coding standards are followed in complete code while naming variables/functions/ classes, explaining the purpose of variables/ functions/classes. ● The Complete code is well-documented with comments explaining the code.	
		<u>Adequate (7-6 marks):</u> ● Coding standards are followed in most of the code while naming variables/functions/ classes, explaining the purpose of variables/ functions/classes. ● Most of the code is documented with comments explaining the code.	
		<u>Poor (5-4 marks):</u> ● Coding standards are followed in very little code while naming variables/functions/ classes, explaining the purpose of variables/ functions/classes. ● Very little code is documented with comments explaining the code.	
		<u>Unsatisfactory (0-3 marks):</u> ● Coding standards are not followed properly. ● Proper comments are not written explaining the code.	
Total Marks for Practical = 0.5 * (Marks of C1) + 0.25 * (Marks of C2) + 0.25 * (Marks of C3)			

Signature with Date:

Practical No. 11: File Handling

i. Write a program to perform the below operations on files:
   - Create a text file and write a string to it.
   - Read an entire text file.
   - Read a text file line by line.
   - Write a string to a file.
   - Write a list of strings to a file.
   - Count the number of lines, words in a file.

ii. Write a program that reads a text file and counts the occurrences of each alphabet in the file. The program should prompt the user to enter the filename.

iii. Write a program that reads a text file and displays all the numbers found in the file.

iv. Write an automated censor program that reads the text from a file and creates a new file where all of the four-letter words have been replaced by “****”. You can ignore punctuation, and you may assume that no words in the file are split across multiple lines.

v. Write a program that reads a text file and calculates the average word length and sentence length in that file.

vi. Write a program that reads two strings stored in two different text files and prints a string containing the characters of each string interleaved. Remove white spaces from both strings before string interleaving. For example, two strings “Hello World” and “Sky is the Limit” should generate output “HSeklyliosWtohrelLdimit”

A. Objectives:

This practical will help students to perform file operations using file handling functions.

B. Relevant Program Outcomes (POs):

i. **Basic and Discipline specific knowledge (PO1):** Apply knowledge of basic mathematics, science and engineering fundamentals and engineering specialization to solve the Computer Engineering problems.

ii. **Problem analysis (PO2):** Identify and analyse well-defined Computer Engineering problems using codified standard methods.

iii. **Design/development of solutions (PO3):** Design solutions for Computer Engineering well-defined technical problems and assist with the design of systems components or processes to meet specified needs.

iv. **Engineering Tools, Experimentation and Testing (PO4):** Apply modern Computer Engineering tools and appropriate technique to conduct standard tests and measurements.

v. **Project Management (PO6):** Use engineering management principles individually, as a team member or a leader to manage projects and effectively communicate about well-defined engineering activities.

vi. **Life-long learning (PO7):** Ability to analyse individual needs and engage in updating in the context of technological changes in field of engineering.

C. Competency and Practical Skills:

This practical is expected to develop the following skills for the industry-identified competency '**Develop simple applications using scripting language Python.**'

i. Problem analysis skills.

ii. Programming skills.

iii. Debugging skills.

D. Relevant Course Outcomes (COs):

i. Perform string manipulation and file operations to solve the given problems.

E. Practical Outcomes:

i. Write Python scripts using file processing functions.

F. Relevant Affective domain Outcomes (ADOs):

i. Maintain tools and equipments.

ii. Follow Coding standards and practices.

iii. Follow ethical practices.

G. Prerequisite Theory:

Python has several functions for creating, reading, updating, and deleting files.

Opening a file:

The open() function takes two parameters; filename, and mode. There are four different methods (modes) for opening a file:

- "r" - Read - Default value. Opens a file for reading, error if the file does not exist
- "a" - Append - Opens a file for appending, creates the file if it does not exist
- "w" - Write - Opens a file for writing, creates the file if it does not exist
- "x" - Create - Creates the specified file, returns an error if the file exists

In addition you can specify if the file should be handled as binary or text mode:

- "t" - Text - Default value. Text mode
- "b" - Binary - Binary mode (e.g. images)

To open a file:

```
f = open("demofile.txt", "rt")
```

Read a file:

By default the read() method returns the whole text, but you can also specify how many characters you want to return:

```
f = open("demofile.txt", "r")
print(f.read(5))
```

You can return one line by using the readline() method:

```
f = open("demofile.txt", "r")
print(f.readline())
```

Write to a file:

You can use write() method to write to a file:

```
f = open("demofile2.txt", "a")
f.write("Now the file has more content!")
f.close()
```

Close Files

It is a good practice to always close the file when you are done with it.

```
f = open("demofile.txt", "r")
print(f.readline())
f.close()
```

H. Resources Required:

Sr. No	Instrument /Components	Configuration/Specification
1.	Computer System	Processor: RAM: Operating System:
2.	Python Interpreter	Python Version:
3.	Text Editor	Editor:

I. Source code and Output:

i. Write a program to perform the below operations on files:

- Create a text file and write a string to it.
- Read an entire text file.
- Read a text file line by line.
- Write a string to a file.

- Write a list of strings to a file.
- Count the number of lines, words in a file.

Source Code:
Output:

ii. Write a program that reads a text file and counts the occurrences of each alphabet in the file. The program should prompt the user to enter the filename.

Source Code:
Output:

iii. Write a program that reads a text file and displays all the numbers found in the file.

Source Code:

Output:

iv. Write an automated censor program that reads the text from a file and creates a new file where all of the four-letter words have been replaced by "****". You can ignore punctuation, and you may assume that no words in the file are split across multiple lines.

Source Code:

Output:

v. Write a program that reads a text file and calculates the average word length and sentence length in that file.

Source Code:

Output:

vi. Write a program that reads two strings stored in two different text files and prints a string containing the characters of each string interleaved. Remove white spaces from both strings before string interleaving. For example, two strings “Hello World” and “Sky is the Limit” should generate output “HSeklyliosWtohrelLdimit”

Source Code:

Output:

J. Practical related Quiz:

i. What is the primary purpose of file handling in Python?

A) To perform mathematical operations on files

B) To create graphical user interfaces

C) To read, write, and manipulate files

D) To work with networking protocols

ii. What is the default mode used by the open() function if no mode is specified?

A) "r" (read mode)

B) "w" (write mode)

C) "a" (append mode)

D) "x" (exclusive creation mode)

iii. How can you read the contents of a file in Python?

A) Using the read() method

B) Using the write() method

C) Using the readline() method

D) Using the writelines() method

iv. How can you write data to a file in Python?

a) Using the read() method

b) Using the write() method

c) Using the readline() method

d) Using the writelines() method

v. What is the purpose of the seek() method in file handling?

A) To set the file pointer at a specific position

B) To rename the file

C) To delete the file

D) To check if the file exists

vi. Which mode is used to open a file for both reading and writing in Python?

A) "r+" (read and write mode)

B) "w+" (write and read mode)

C) "a+" (append and read mode)

D) "x+" (exclusive creation and read mode)

vii. What is the purpose of the "with" statement in file handling?

A) To close the file automatically after usage

B) To open multiple files simultaneously

C) To read files in reverse order

D) To create a backup of the file

viii. How can you check if a file exists in Python?

A) Using the exists() function from the os module

B) Using the exists() function from the sys module

C) Using the file_exists() function from the fileio module

D) Using the file_exists() function from the io module

ix. What is the purpose of the isfile() function in file handling?

A) To check if a file is a regular file

B) To check if a file is a directory

C) To check if a file is readable

D) To check if a file is writable

x. Which method is used to check if the file pointer is at the end of the file?

A) end()

B) is_end()

C) eof()

D) at_end()

xi. What is the purpose of the flush() method in file handling?

A) To close the file

B) To write the file contents to disk

C) To read the file contents into memory

D) To clear the file contents

K. References:

i. https://www.w3schools.com/python/python_file_handling.asp

ii. https://www.geeksforgeeks.org/file-handling-python/

iii. https://www.javatpoint.com/python-files-io

iv. https://www.programiz.com/python-programming/file-operation

v. https://www.freecodecamp.org/news/file-handling-in-python/

vi. https://www.tutorialspoint.com/python/python_files_io.htm

L. Assessment Rubrics:

Criteria	% of point	Rubrics	Marks
C1: Program Completeness/ Correctness	50 %	Excellent (10-8 marks): Completed programs/scripts correctly as per the requirements.	
		Adequate (7-6 marks): Completed programs/scripts correctly with approx. 70% requirements.	
		Poor (5-4 marks): Completed programs/scripts correctly with 70% - 50% requirements.	
		Unsatisfactory (0-3 marks): Completed programs/ scripts correctly with less than 50% requirements.	
C2: Readability	25 %	Excellent (10-8 marks): The code is clean, well-organized and very easy to understand.	
		Adequate (7-6 marks): The code is fairly easy to read and understand.	
		Poor (5-4 marks): The code is readable only by someone who knows what it is supposed to be doing.	
		Unsatisfactory (0-3 marks): The code is poorly organized and very difficult to understand.	
C3: Coding Standards/ Documentation	25 %	Excellent (10-8 marks): ● Coding standards are followed in complete code while naming variables/functions/ classes, explaining the purpose of variables/ functions/classes. ● The Complete code is well-documented with comments explaining the code.	
		Adequate (7-6 marks): ● Coding standards are followed in most of the code while naming variables/functions/ classes, explaining the purpose of variables/ functions/classes. ● Most of the code is documented with comments explaining the code.	
		Poor (5-4 marks): ● Coding standards are followed in very little code while naming variables/functions/ classes, explaining the purpose of variables/ functions/classes. ● Very little code is documented with comments explaining the code.	
		Unsatisfactory (0-3 marks): ● Coding standards are not followed properly. ● Proper comments are not written explaining the code.	
Total Marks for Practical = 0.5 * (Marks of C1) + 0.25 * (Marks of C2) + 0.25 * (Marks of C3)			
Signature with Date:			

SCRIPTING LANGUAGE - PYTHON

4330701

Lab manual is prepared by

Shri. Kartik J. Detroja

Lecturer, Computer Engineering

Government Polytechnic - Porbandar

Smt. Kinjal K. Patel

Lecturer, Computer Engineering

Government Polytechnic - Ahmedabad

<u>Branch Coordinator</u>

Shri Kantevala Bipinkumar Haribhai

HoD, Computer Engineering

Government Polytechnic - Ahmedabad

<u>Committee Chairman</u>

Shri R. D. Raghani

(HOD-EC)

Principal (I/C)

Government Polytechnic, Gandhinagar