جب کوئی قوم فن اور علم سے عاری ہو جاتی ہے تو وہ غربت کو دعوت دیتی ہے اور جب غربت آتی ہے تو وہ ہزاروں جرائم کو جنم دیتی ہے۔

"When a nation becomes devoid of art and learning, it invites poverty and when poverty comes it brings in its wake thousands of crimes."
-Sir Syed Ahmad Khan

B.Sc. (CA) VI Semester Lab

Laboratory Course-VI

Course Code- CABSMJ6P06

DEPARTMENT OF COMPUTER SCIENCE

ALIGARH MUSLIM UNIVERSITY, ALIGARH

2025-2026

Credits

The following lab manual up-gradation committee updated the lab manual:

- ❒ *Prof. Arman Rasool Faridi (Chairperson)*
- ❒ *Prof. Mohammad UbaidUllah Bokhari*
- ❒ *Prof. Aasim Zafar*
- ❒ *Prof. Suhel Mustajab*
- ❒ *Dr. Faisal Anwar*
- ❒ *Dr. Mohammad Sajid*
- ❒ *Dr. Mohammad Nadeem*
- ❒ *Dr. Faraz Masood*

The following committee member originally designed the lab manual:

- ❒ *Prof. Mohammad Ubaidullah Bokhari*
- ❒ *Dr. Arman Rasool Faridi*
- ❒ *Dr. Faisal Anwer*
- ❒ *Prof. Aasim Zafar (Convener)*

Design & Compilation:

- ❒ *Dr Faisal Anwer, Dr. Faraz Masood*

Revised Edition: Dec, 2026

Department of Computer Science, A.M.U.,

Aligarh (U.P.) India

COURSE TITLE: Laboratory Course-VI **COURSE CODE:** CABSMJ6P06
CREDIT: 04 **PERIODS PER WEEK:** 06
CONTINUOUS ASSESSMENT: 60 **EXAMS:** 40

COURSE DESCRIPTION

This manual is intended for the second-year students in the subject of Programming with Python. This manual typically contains Practical/Lab Sessions related to Programming with Python covering various aspects in order to enhance subject understanding.

Python is a general purpose, high-level programming language; other high-level languages you might have heard of C++, PHP, and Java. Virtually all modern programming languages make us of an Integrated Development Environment (IDE), which allows the creation, editing, testing, and saving of programs and modules. In Python, the IDE is called IDLE (like many items in the language, this is a reference to the British comedy group Monty Python, and in this case, one of its members, Eric Idle).

Many modern languages use both processes. They are first compiled into a lower-level language, called byte code, and then interpreted by a program called a virtual machine. Python uses both processes, but because of the way programmers interact with it, it is usually considered an interpreted language.

There are two ways to use the Python interpreter: shell mode and script mode. In shell mode, you type Python statements into the Python shell and the interpreter immediately prints the result. In script mode, you type statements into the editor and save it in a file known as script. The interpreter executes the code of the script.

Students are advised to thoroughly go through this manual rather than only topic mentioned in the syllabus as practical aspects are the key to understanding and conceptual visualization of theoretical aspects covered in the books.

COURSE CONTENT

This course is designed to provide the students the opportunity of learning programming with Python. This course is indented to develop a deep understanding of various operations on data structure such as searching, sorting, insertion, deletion and traversing and provides students the idea of new programming language. Apart from basic concepts, it helps to understand the concept of lists, tuples, dictionaries and file handling.

OBJECTIVES

This course is designed to help students:

- ❒ Introduce the basic concepts of Python programming language.
- ❒ Make students familiar with the algorithmic approach to problem-solving in Python language.

OUTCOMES

After completing this course, the students would be able to:

- ❒ Develop, debug and document programs in Python.
- ❒ Define/use Python functions, modules and packages.
- ❒ Use Python data structures –- lists, tuples, dictionaries.
- ❒ Do input/output with files in Python.
- ❒ Explain and write programs in Python language to solve common problems using concept of Data types such as lists, tuples and dictionaries.
- ❒ Analyze, design and demonstrate various algorithms for data analysis.

HOW TO DO WELL IN THIS COURSE

- ❒ The students are advised to attend all their theory classes and respective labs regularly as both are integrated to each other. If any student will miss the theory lecture, he/she may not able to do well in lab related to that topic.

- ❒ The students are advised to submit the assignments given in theory and lab classes timely to their respective Teachers/Instructors.
- ❒ The students should demonstrate disciplined and well-behaved demeanor in the Department.
- ❒ Each student shall be assigned a system in their introductory lab. They are advised to do their work on that system only for the whole semester. Students should store all their lab activities regularly.
- ❒ All students are advised to understand course objectives and outcomes and achieve both during their lab work.
- ❒ The students are advised to follow books/eBooks/online tutorial/other online study material links given in lecture/lab manual/ syllabus references. These study materials are very helpful in terms of skills, knowledge and placement.
- ❒ This Lab course is very important in terms of placement. Therefore, students are advised to implement all the problems by her /him given in the individual week.
- ❒ All students are advised to solve old placement papers for campus selection.

❖ Following links may be useful for the preparation of your campus placements.

- https://www.indiabix.com/placement-papers/companies/
- https://www.offcampusjobs4u.com/download-tcs-placement-test-question-papers-with-solutions/
- https://www.indiabix.com/placement-papers/tcs/
- https://www.firstnaukri.com/career-guidance/infosys-placement-papers-with-solutions-2019-firstnaukri-prep
- https://prepinsta.com/ibm/
- https://www.faceprep.in/infosys/infosys-aptitude-questions/

- https://alpingi.com/infosys-placement-papers-solution-pdf-download/
- http://placement.freshersworld.com/

❒ Students are advised to follow mentioned tutorials links:

- https://www.w3schools.com/python/
- https://www.tutorialspoint.com/python/index.htm
- https://www.learnpython.org/
- https://www.javatpoint.com/python-tutorial

❒ Students are advised to follow mentioned links to download software:

- https://www.python.org/downloads/
- https://jupyter.org/install

RULES AND REGULATIONS

Students are required to strictly adhere to the following rules.

❒ The students must complete the weekly activities/assignments well in time (i.e., within the same week) that need to be checked and signed by the concerned teachers in the lab in the immediate succeeding week. Failing which activities/assignments for that week will be treated as incomplete.

❒ The students must maintain the Lab File of their completed activities/assignments in the prescribed format (**Appendix 1**).

❒ At least **TWELVE (12)** such timely completed and duly signed weekly activities/assignments are compulsory, failing which students will not be allowed to appear in the final Lab Examination.

❒ Late submissions would not be accepted after the due date.

❒ The Continuous Lab assessment will be based on two sessionals, each carrying 30 marks.

- Marks distribution for each sessional:
  - 20 Marks: For a duly signed lab report by the respective lab teacher.
  - 5 Marks: For solving a lab question/program on the day of the sessional.
  - 5 Marks: For the viva conducted during the sessional.
- This distribution ensures a balanced evaluation of students' practical work, problem-solving skills, and conceptual understanding.
- Cooperate, collaborate, and explore for the best individual learning outcomes, but copying is strictly prohibited.

APPENDIX-1

Template for the Index of Lab File

WEEK NO.	PROBLEMS WITH DESCRIPTION		PAGE NO.	SIGNATURE OF THE TEACHER WITH DATE
1	1#			
	2#			
	3#			
2	1#			
	2#			
	3#			
3	1#			
	2#			
	3#			

Note: *The students should use a Header and Footer mentioning their roll no. & name in footer and page no in header.*

WEEK #1

OBJECTIVES

- ❒ To learn the basic features of the Python programming language

OUTCOMES

After completing this, the students would be able to:

- ❒ To explain the basic features of the Python programming language.

PROBLEMS

1# Discuss the process by which Python executes a program, focusing on the stages involved from source code to execution.

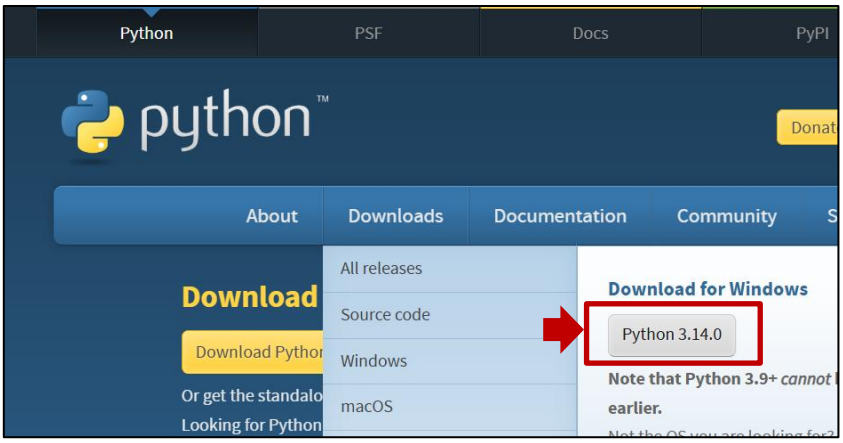
2# Discuss the fundamental features of the Python programming language that make it widely used for modern software development.

3# Install IDLE interpreter for Python programming by following the steps mentioned below.

STEPS TO INSTALL PYTHON 3.14 FOR WINDOWS

STEP 1: DOWNLOAD PYTHON 3.14

To start, go to python.org/downloads Click on Downloads tab and click on the button to download the latest version of Python:

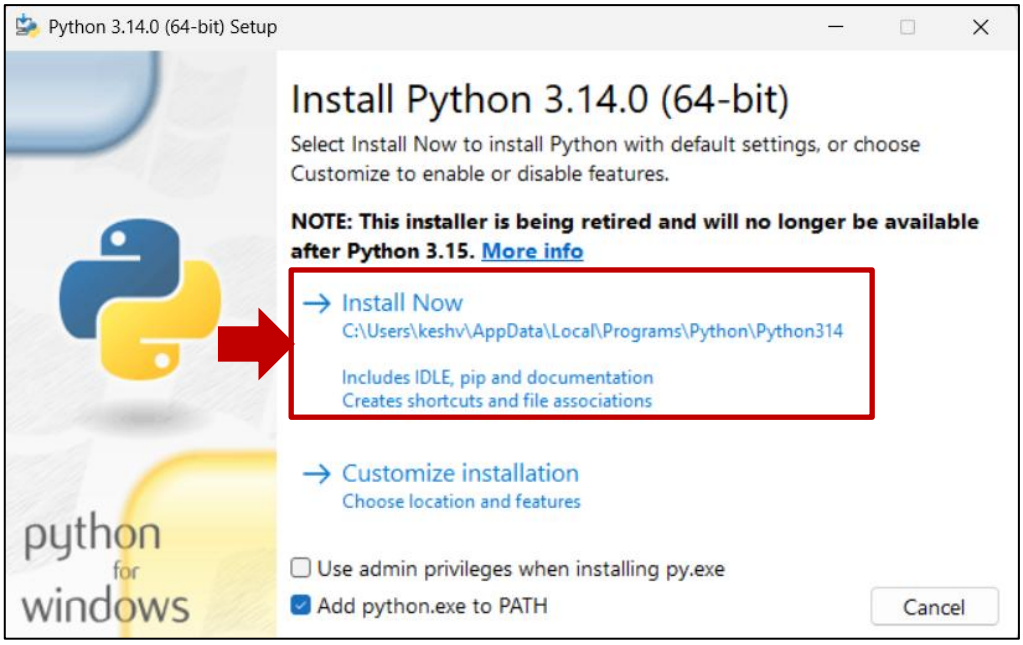


STEP 2: RUN THE .EXE FILE

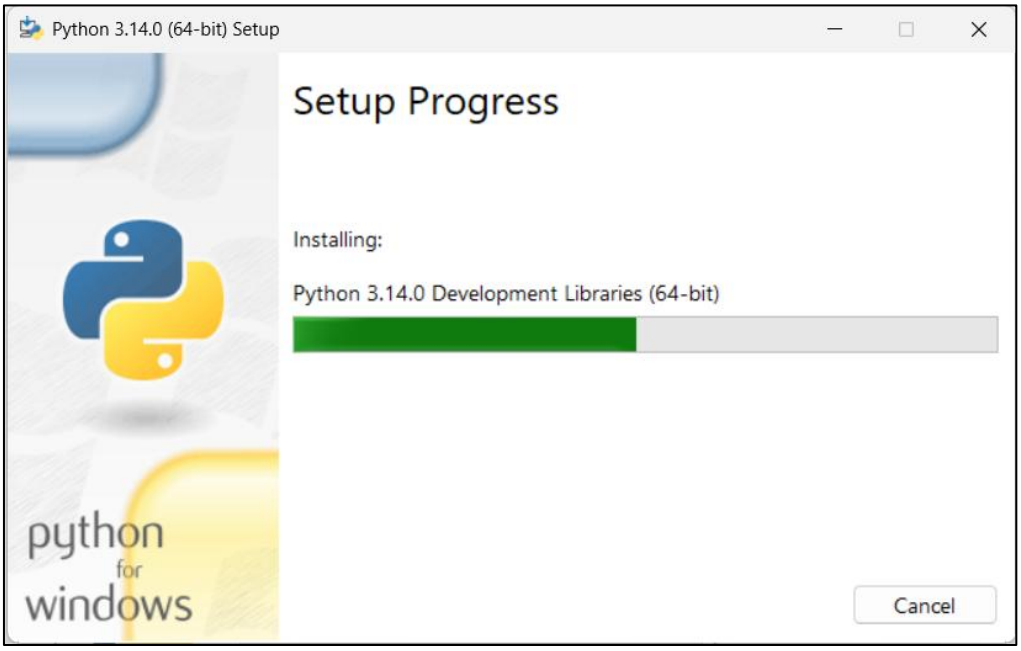
Next, run the .exe file that you just downloaded.

STEP 3: INSTALL PYTHON 3.14

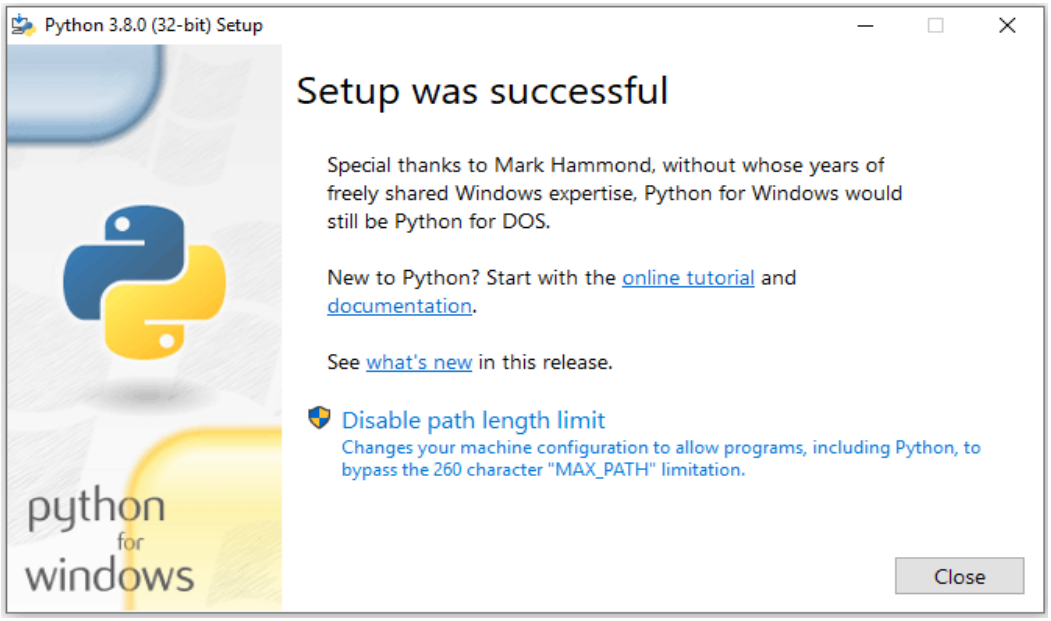
You can now start the installation of Python by clicking on Install Now. (Note that you can enable the **Add Python to PATH** option if required.)



Your installation should now begin:



After a short period of time, your setup will be completed:

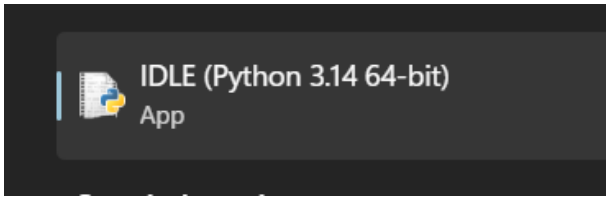


Congrats, you just installed Python for Windows!

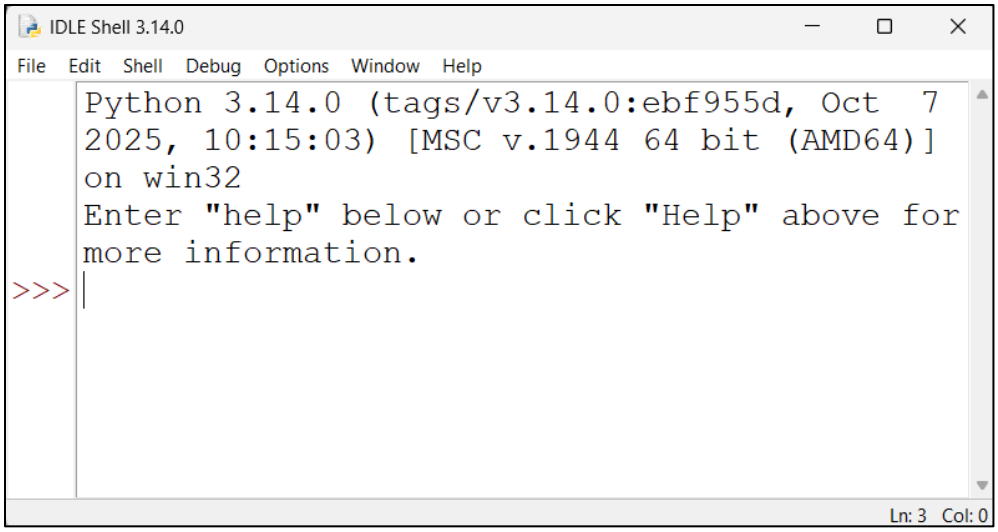
Let's now see how to run a simple code in Python.

STEP 4: OPEN IDLE INTEGRATED DEVELOPMENT ENVIRONMENT FOR PYTHON

A quick way to find your Python IDLE on Windows is by clicking on the Start menu. Search for 'IDLE', and click to open.



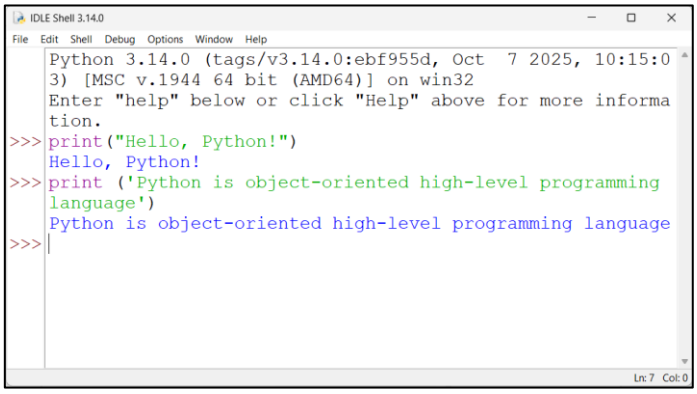
Once you click on the Python IDLE, you will then see the Shell screen:



STEP-5: RUN FOLLOWING COMMAND IN IDLE INTERPRETER

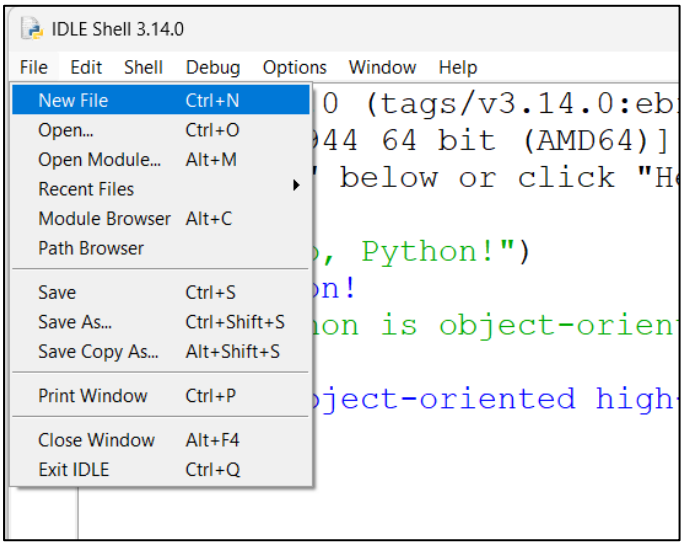
print ('Hello, world')

print ('Python is object-oriented high-level programming language')

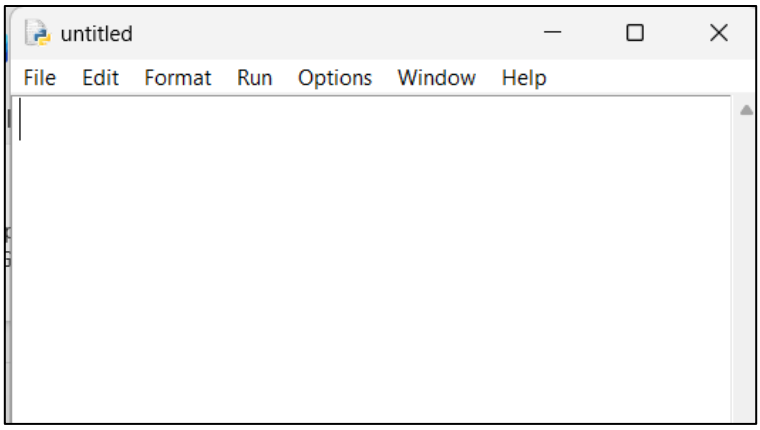


STEP-6: CREATE A PYTHON SCRIPT

Click on File and then select New File (alternatively, you may use the keyboard shortcut of CTRL):

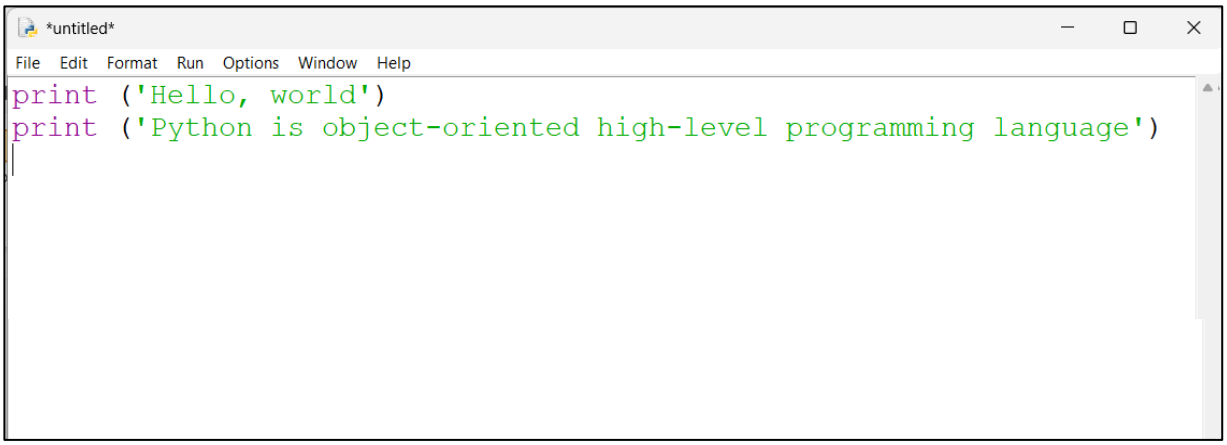


You will now see the "untitled" box, where you can type your Python code:



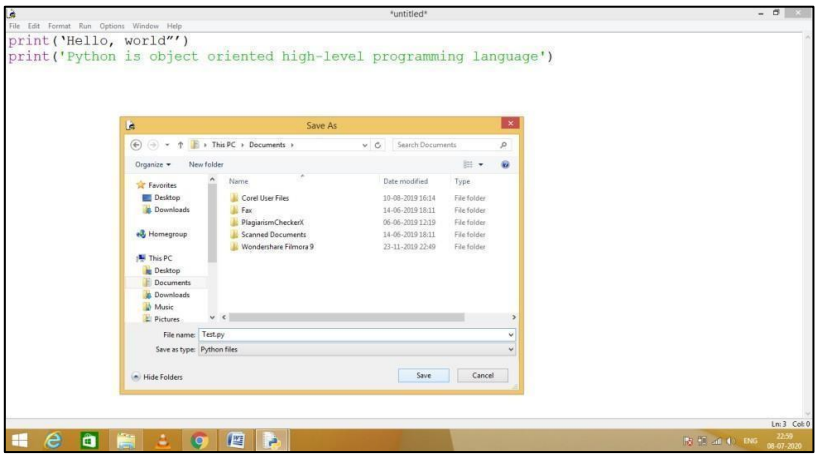
Write the following commands in the python script file print (‘Hello, world”’)

print ('Python is object-oriented high-level programming language') This is how the command would look like in the “untitled” box:

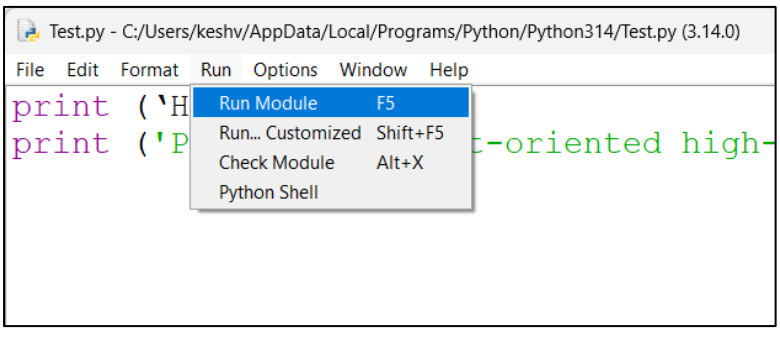


Save this with. py extension

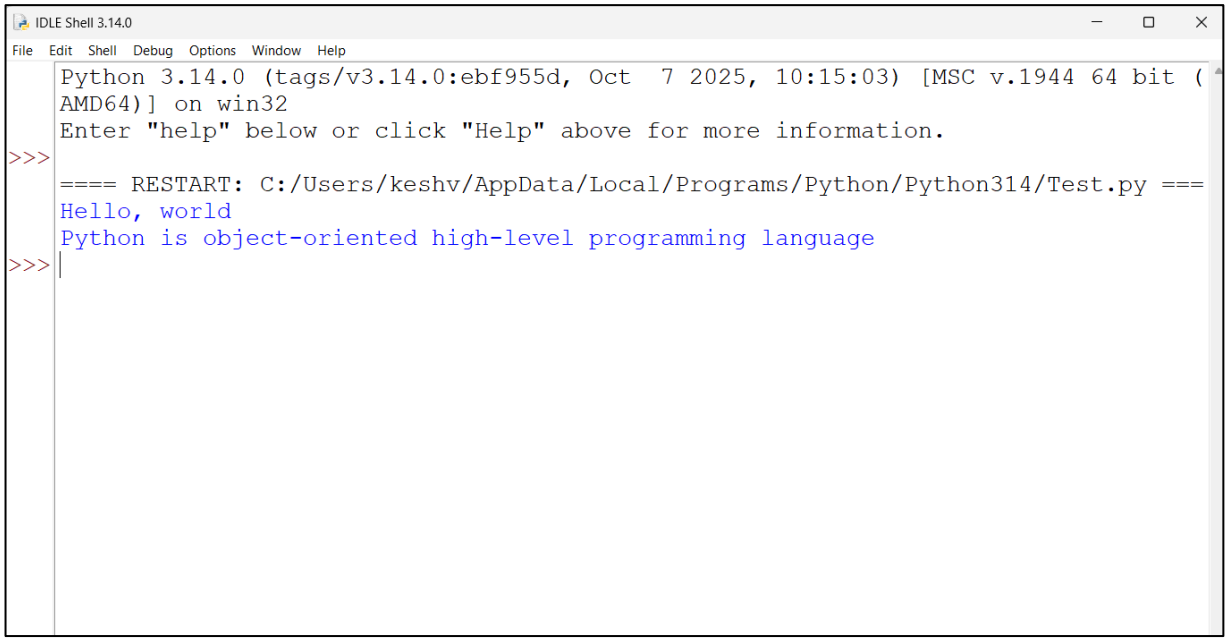
Choose a location where the Python file will be saved on your computer. You’ll also need to type a name for your file. For example, I named the file as “Test”



After saving file, press F5 or Run Module in Run



The output will be printed on your python shell.



WEEK #2

OBJECTIVES

- ❒ The basic printing command is print.
- ❒ Different operators (+, -, *, /, %).
- ❒ Integer and floating number arithmetic.

OUTCOMES

After completing this, the students would be able to:

- ❒ To write and run a simple Python statement/ program in Shell and Script mode.
- ❒ **Note: When naming a Python program, a .py extension must be used.** Failing to do so will result in code that will not be recognized by the system as a Python program.

PROBLEMS

1# Use interactive Shell to print:

- User said "Great app!"
- C:\new_folder\test\

Example 1: (*\n)*

• Input: User said "Great app!"\nC:\new_folder\test\

• Output:

User said "Great app!"

C:\new_folder\test\

Example 2: (*\t)*

• Input: User said\t"Great app!"

• Output:

User said "Great app!"

2# A shopkeeper uses Python shell for quick billing calculations. Evaluate: 8+12, 94, 15-7, 18/3, 19/4, 19%4, 12%3, 37-5, 10-3*6-2

Example 1:

• Input: 10 / 0

• Output: ZeroDivisionError

Example 2:

• Input: 999999999999999 * 888888888888888

• Output: 888888888888887111111111111112

3# A college form stores student names. Create variables firstName and lastName with your own names.

4# Print firstName and lastName in single line. (use f"")

(Try: print(firstName, lastName)

5# A heritage museum kiosk needs an automated display-and-billing script. Create a single Python file kioskSystem.py that performs the following:

i) **Billing Module:**

The museum café calculates totals for simple arithmetic checks.

Write code to compute and print results for:

- 10 + 5
- 12 * 4
- 25 - 8
- 30 / 7
- 30 % 3

ii) **Information Display Module:**

After billing output, print two separate informational messages for visitors:

- "Welcome to the National Art Museum"

- "Digital exhibits were introduced in 2015"

6# An engineer calculates load using a polynomial. Given a=3, b=-2, c=6, d=1, e=4, f=7 and input x,

Compute: $ax^5 + bx^4 + cx^3 + dx^2 + ex + f^{1/2}$

7# Write a program to compute the value of following algebraic expression-

$$\frac{1 + \frac{x}{y} + x^y}{2 + \frac{y}{x} + y^x}$$

The value of x and y will be read using input () function.

8# A technician optimizes multiplication using bitwise operations. Read an integer and multiply it by 2 using <<.

Example 1:

• Input: n = 5

• Operation: n << 1

• Output: 10

Example 2:

• Input: n = 20

• Operation: n >> 1

• Output: 10

WEEK #3

OBJECTIVES

- ❒ To learn different data types (int, float and string) in Python.
- ❒ To learn the rules to define variables in Python.
- ❒ To learn the use of Assignment (=) and operator in Python.
- ❒ To learn the basic input receiving command in Python.

OUTCOMES

After completing this, the students would be able to:

- ❒ To learn how to input character, variables/keywords, evaluate expressions.
- ❒ To Learn three basic types of simple variables in Python: integers (whole numbers, commonly used as an index), floats (that is, numbers with a decimal point, AKA real numbers), and strings (collections of alpha numeric characters such as names, sentences, or numbers that are not manipulated mathematically such as a part number or zip code).
- ❒ The first assignment to a variable creates it. The variable types don't need to be declared. Python figures out the variable types on its own.

PROBLEMS

1# A bookstore's billing app needs a quick utility script. Write a Python program that reads N integers (prices) and computes:

i) SUM

ii) PRODUCT

iii) AVERAGE

2# A weather station needs temperature conversion. Write a program to convert Celsius to Fahrenheit. (Celsius values can be fractional.)

3# An architecture tool calculates dimensions. Write a program to compute the Surface Area and Volume of a Cuboid given length, breadth, height.

4# Write a program to add two numbers without using arithmetic operators, using bitwise operators.

5# A hardware technician tests divider circuits. Read an integer and divide it by 4 using the bitwise >> operator.

Example 1:

• Input: n = 20

• Output: 5

Example 2:

• Input: n = -20

• Output: -5

6# A robotics controller swaps sensor IDs efficiently. Read two integers and exchange their values using XOR (^) operator.

7# A gaming console optimizes score calculation. Multiply a number by 10 using bitwise operators.

8# A data-logging device swaps configuration settings. Read two integers and exchange their values using addition and subtraction (avoid using a third variable).

9# A biology simulation needs repeated multiplication. Write a program to compute the factorial of an integer using a loop.

Example 1:

• Input: 0
• Output: 1

Example 2:

• Input: -4
• Output: Invalid input

10# A math module checks sequence validity. Write a program to determine whether a number is a Fibonacci number.

11# A security system verifies prime-number-based keys. Write a program to check whether an integer is prime, or else output its first factor.

Example 1:

• Input: 1

• Output: Not a prime number

Example 2:

• Input: -7

• Output: Invalid input

12# A signal-processing unit finds harmony between two frequencies. Write a program to compute the GCD of two numbers.

Example 1:

• Input: a = 0, b = 18

• Output: 18

Example 2:

• Input: a = 8, b = 15

• Output: 1

WEEK #4

OBJECTIVES

- ❒ To learn the concept of functions.

OUTCOMES

After completing this, the students would be able to:

- ❒ To explain the concept and creation of functions.

PROBLEMS

1# A teacher wants to automate small mathematical tasks done repeatedly while checking assignments. Define separate Python functions to perform the following for the teacher:

- Compute the average of any five marks obtained by a student.
- Convert a given temperature from Celsius to Fahrenheit for recording lab conditions.
- Calculate the perimeter of a rectangular notice board kept in the classroom using its length and width.

  Example 1:

  • Input: Marks = [-10, 0, 20, 30, 40]

  • Output: Average = 16.0

  Example 2:

  • Input: Temperature = 36.5

  • Output: 97.7°F

2# A science lab technician is calculating the volume of spherical containers used for experiments. Write a Python function that accepts the radius and returns the volume of the sphere using the formula:

$$V = \frac{4}{3}\pi r^3$$

Find the volume when **radius = 6 cm**.

Example 1:

• Input: radius = 0.5

• Output: 0.52 cm³

Example 2:

• Input: radius = -3

• Output: Invalid radius

3# A landscaping company maintains circular flower beds, each with an outer radius R and an inner unused radius r. The effective area used for fertilizer is Effective Area $= \boldsymbol{\pi(R^2 - r^2)}$

Write a Python function that takes R and r from the user, validates that R > r, and returns the effective usable area.

Example 1:

• Input: R=50.5, r=12.3

• Output: Effective Area = 7504.67

Example 2:

• Input: R=10, r=10

• Output: Invalid input

4# A data analyst needs factorial values to compute permutations. Write a Python function to calculate the factorial of a non-negative integer.

Example 1:

• Input: 0

• Output: 1

Example 2:

• Input: 1

• Output: 1

5# An accountant wants a utility that extracts a digit-sum check value from transaction IDs. Write a Python function that returns the sum of digits of a given integer.

Example:

• Input: "0123"

• Output: 6

6# A math research assistant is verifying test values for numerical experiments. Write a Python function to check whether a number is a perfect number. In number theory, a perfect number is a positive integer that is equal to the sum of its proper positive divisors, that is, the sum of its positive divisors excluding the number itself. Example:6, 28 etc.

Example 1:

• Input: 1

• Output: Not a perfect number

Example 2:

• Input: 28

• Output: Perfect number

7# Write a Python function to check whether a given number is a prime number or not.

Example 1:

• Input: 0

• Output: Not prime

Example 2:

• Input: 1

• Output: Not prime

8# A text-analysis tool needs statistics from a sentence. Write a Python function that accepts a string and counts the number of uppercase and lowercase letters using isupper(), islower(), upper(), lower().

Example 1:

- Input: "Hello@123"
- Output: Uppercase=1, Lowercase=4

Example 2:

- Input: "PYTHON"
- Output: Uppercase=6, Lowercase=0

WEEK #5

OBJECTIVES

- ❒ To learn the concept of function in Python.
- ❒ To learn Built-in functions: Type conversion and Math functions in Python.
- ❒ To learn the user-defined functions: Definitions and use in Python.

OUTCOMES

After completing this, the students would be able to:

- ❒ To write, debug and run a simple Python function.
- ❒ To learn how to define user-defined functions and pass arguments to functions in Python.

PROBLEMS

1# A data automation team is creating a utility module that performs multiple independent calculations based on user inputs. Write a Python program that accepts required inputs from the user and defines separate functions to perform:

- Find the largest of three numbers (positional arguments).
- Compute the volume based on user's chosen shape: – cylinder (r, h), cube (a), or rectangular box (l, w, h).
- Compute the area of a rectangle.
- Compute the circumference of a circle.
- Exchange the values of two variables.
- Find the distance between two points (x1, y1) and (x2, y2) using math.dist().

(Use switch case to execute the required function)

2# A billing system receives an unknown count of numeric entries per invoice. Write a Python function that accepts arbitrary integers and returns their sum.

(Hint: use *args)

Example:

• Input: ()

• Output: 0

3# A text-filtering tool needs to remove noisy characters occurring at odd positions. Write a Python program that takes a string and removes all characters at even index values.

Example 1:

• Input: "H@e#l$l"

• Output: "el"

Example 2:

• Input: ""

• Output: ""

4# Write a Python script that takes input from the user and displays that input back in upper and lower cases.

5# A simple analytics module must calculate keyword frequency in user feedback. Write a Python program to count occurrences of each word in a sentence.

Example 1:

• Input: "Hello hello world"

• Output: {'hello': 2, 'world': 1}

Example 2:

• Input: "Hi, Hi!"

• Output: {'hi': 2}

6# A document cleanup tool needs to normalize text copied from notes. Write a Python program to remove indentation from each line of a multi-line text. \

7# Write a function that reverses a string. The input string is given as an array of characters s. You must do this by modifying the input array in-place with O(1) extra memory.

8# Given a string s, reverse the order of characters in each word within a sentence while still preserving whitespace and initial word order.

9# A parser converts messages into list-based tokens. Write a Python program to convert a string into a list of characters.

Example 1:

- Input: "Hi 5"
- Output: ['H','i',' ','5']

Example 2:

- Input: "1234"
- Output: ['1','2','3','4']

10# Write a Python program to count and display all vowels present in the text. Also display consonants and its count.

WEEK #6

OBJECTIVES

- ❒ To learn Boolean Expression in Python.
- ❒ To learn the logical operators, relational operators.
- ❒ To learn the Conditional (if), Alternative (if-else).
- ❒ To learn the Chained conditional (if-elf-else), Nested Conditional.

OUTCOMES

After completing this, the students would be able to:

- ❒ To write, debug and run a simple Python program based on conditions.

PROBLEMS

1# A grading system compares three internal-assessment scores. Write a program to find the highest score among the three, using only if statements.

2# A sensor module collects readings from three temperature probes. Write a program to determine the lowest reading using if–else logic.

3# Write a program to check a year for leap year. (Use if and else)

4# Write a program to print number of days in a month. (Chained Conditional)

5# Write a Python program that takes a month (1–12) and a year, uses a list of days in each month, and prints the correct number of days. Make sure of leap year for February.

Example 1:

• Input: month=2, year=2024

• Output: 29

Example 2:

• Input: month=2, year=1900

• Output: 28

6# An architectural software checks the type of triangular panels used in design. Write a program to:

- Compute the area of a triangle using its three sides.
- Print whether it is equilateral, isosceles, or scalene.

7# A device logs data indexes between two given IDs. Given two numbers r1 and r2 (r1 < r2): Write a program to create a list of integers from r1 to r2 (inclusive) using range().

8# A parser receives a mixed list of characters, symbols, and integers. Write a program to extract and add only numeric items using isinstance().

Example 1:

• Input: [1, 2.5, 3, -4]

• Output: 0

Example 2:

• Input: [10, -5, 20]

• Output: 25

9# A game engine verifies whether level IDs are arranged in a proper continuous sequence. Write a Python program to check whether a given list of numbers is consecutive.

Example 1:

• Input: [4, 2, 3, 5]

• Output: True

Example 2:

• Input: [1, 2, 2, 3]

• Output: False

10# Given an integer n, return a string array answer (1-indexed) where:

- answer[i] == "FizzBuzz" if i is divisible by 3 and 5.
- answer[i] == "Fizz" if i is divisible by 3.
- answer[i] == "Buzz" if i is divisible by 5.
- answer[i] == i (as a string) if none of the above conditions are true.

Example :

Input: n = 15

Output:
["1","2","Fizz","4","Buzz","Fizz","7","8","Fizz","Buzz","11","Fizz","13","14","FizzBuzz"]

WEEK #7

OBJECTIVES

- ❒ To learn definite iteration (For loop) and conditional iteration (While loop) in Python.
- ❒ To learn the Conditional (if), Alternative (if-else), break and pass statements in Python.
- ❒ To learn the recursion, flow of execution, infinite recursion in Python.

OUTCOMES

After completing this, the students would be able to:

- ❒ To write, debug and run a simple Python function based on iteration and recursion.

PROBLEMS

1# A cryptographic security module uses "Narcissistic Numbers" (Armstrong numbers) as a part of a checksum validation. Write a Python program using a while loop to check if a user-provided numeric key is an Armstrong number.

Example 1:

• Input: 153

• Output: Armstrong number

Example 2:

• Input: 9474

• Output: Armstrong number

2# A biological simulation tool models the growth of a cell culture following the Fibonacci sequence. Write a recursive python function to generate the first n terms of the sequence to predict population growth steps.

3# A data science team is building a module to calculate permutations (combinations of arrangements) for a logistics algorithm. To compute the number of possible routes, they need a core component that calculates factorials efficiently. Write a recursive Python function named calculate_routes(n) that accepts a positive integer n and returns its factorial (the product of all positive integers less than or equal to n). Example: Input: n=5 → Output: 120

4# A telecommunications engineer is designing a system to synchronize two signals with different frequencies. To find the moment when both signals align perfectly, the system needs to determine the greatest common divisor (GCD) of their cycle times. Write a recursive Python function named find_signal_sync(a, b) that takes two integers a and b (representing the cycle times) and returns their Greatest Common Divisor using the Euclidean algorithm. Example: Input: a=48, b=18 → Output: 6

5# Write a program in Python to print the following Pattern. (for)

* * * *

* * *

* *

*

6# A network packet analyzer receives a batch of data packet IDs (integers). Write a Python program to filter these IDs, placing even IDs into one list (low priority) and odd IDs into another list (high priority).

7# Given an integer x, return true *if* x *is a palindrome, and* false *otherwise*.

Example:

Input: x = 121

Output: true

Explanation: 121 reads as 121 from left to right and from right to left.

8# An inventory management system is consolidating records from two different warehouses. Write a Python program to merge two lists of product IDs and display the final list in sorted order.

Example 1:

• Input: [1,2], [3,4,5]

• Output: [1,2,3,4,5]

Example 2:

• Input: [1,2], [2,3]

• Output: [1,2,2,3]

9# An algorithm visualization tool demonstrates how different sorting logic works. Write a Python program to find the second largest number in a list. (use Bubble Sort algorithm)

WEEK #8

OBJECTIVES

- ❒ To learn the concept of string in Python.
- ❒ To learn the different String functions and methods.

OUTCOMES

After completing this, the students would be able to:

- ❒ To access individual characters in a string.
- ❒ To retrieve a substring from a string.
- ❒ Top search for a substring in a string.
- ❒ To use string methods to manipulate strings.

PROBLEMS

1# A file management system needs to organize a batch of filenames based on the length of the name rather than alphabetical order. Write a Python program to sort a list of strings (filenames) based on the length of each element.

2# A marketing team has two separate lists of email addresses from two different campaigns. Write a Python program to create a master list containing all unique emails from both lists (Union). (An email can be part of both lists)

3# A cyber-security tool is comparing two server access logs to identify suspicious IP addresses that appear in both logs. Write a Python program to find the common elements between two lists (Intersection).

4# A data visualization tool needs to plot a curve. Write a Python program that generates a list of tuples, where the first element is a number and the second is five times its cube, for a range of inputs.

5# A Natural Language Processing (NLP) pre-processor needs to clean and analyze raw text data. Assume the variable data contains the string:

"Different string Methods!". Write a Python script using string methods to:

- Tokenize the string into a list of individual words.
- Normalize the text by converting it to uppercase.
- Find the index position of the substring "rules".
- Search for a specific character (e.g., 'o') to see if it exists.
- Sanitize the text by replacing the exclamation point with a question mark.

6# A data compression utility needs to analyze the entropy of a text file. Write a Python program to calculate the frequency of every character present in a given string.

7# A simple encryption tool obfuscates passwords by reading them backward. Write a Python program to reverse the order of characters in a user-provided string.

Example:

- Input: "Security"
- Output: "ytiruceS"

8# A product labeling system generates short "SKU codes" based on product names. Write a Python program that takes a string and generates a new string made of the first 2 and last 2 characters. *(Constraint: If the product name is shorter than 2 characters, return an empty string)*

Example:

- Input: "Panasonic"
- Output: "Paic"

9# A library catalog system contains sorted Book IDs. To optimize lookup speeds, write a Python program to implement Binary Search using the divide and conquer approach to find a specific Book ID.

10# An educational algorithm visualizer needs to show students how sorting works step-by-step. Write a Python program to implement the Insertion Sort algorithm that prints the state of the list after every iteration to visualize the intermediate steps.

WEEK #9 & #10

OBJECTIVES

- ❒ To learn about list and its associated operations in Python.
- ❒ To learn the various list operations in Python.

OUTCOMES

After completing this, the students would be able to:

- ❒ to construct lists and access items in those lists.
- ❒ to use methods to manipulate lists.
- ❒ to perform traversals of lists to process items in the lists
- ❒ to define simple functions that expect parameters and return values.

PROBLEMS

1# A financial risk analysis tool needs to determine the volatility of a specific stock. Write a Python program that takes a list of daily closing prices and identifies the Maximum (peak) and Minimum (trough) values in that list.

2# An inventory checking system allows a store manager to scan a product ID to see if it is currently in stock. Write a Python program to search for a specific element in a list of product IDs.

Example 1:

• Input: List=[10,20,30], Item=20

• Output: Found

Example 2:

• Input: List=[10,20,30], Item=40

• Output: Not Found

3# A music playlist app needs to organize songs for the user. Write a Python program to take a raw list of song titles and sort the list alphabetically.

4# An election tally system has a dictionary where keys are candidate names and values are vote counts. Write a Python script to sort this dictionary by value:

- First, display the winner (Descending order).
- Second, display the candidate with the fewest votes (Ascending order).

5# A Data Science team is compiling a "Complete Patient Record" from three different hospital departments (General Info, Vitals, and Medical History). Since these departments track completely different metrics, there are no overlapping keys. Write a Python script to concatenate these three distinct dictionaries into one new, single Patient_Profile dictionary.

Key Point: Concatenation joins distinct pieces of data together.

Example:

- Input:
  - Dept_A = {'Name': 'Shivi', 'Age': 23}
  - Dept_B = {'Height': 165}
  - Dept_C = {'BloodType': 'B+'}
- Output: {'Name': 'Shivi', 'Age': 23, 'Height': 165, 'BloodType': 'B+'}

6# An embedded system with very limited memory needs to sort a small buffer of data packets. Write a Python script to implement the Optimized Bubble Sort algorithm. (Implement flag mechanism)

7# A Big Data processing engine needs to sort a massive log file efficiently. Write a Python program to implement the Merge Sort algorithm. You must use recursion to demonstrate the divide-and-conquer methodology. Explain its best, average, and worst-case complexities.

8# A real-time gaming engine needs a high-speed sorting algorithm for rendering objects. Write a Python program to implement the Quick Sort algorithm. Explain its best, average, and worst-case complexities.

WEEK #11 & #12

OBJECTIVES

- ❒ To learn about dictionary, how to create a dictionary.
- ❒ To learn the various dictionary operations and methods.
- ❒ To learn the reverse search method.

OUTCOMES

After completing this, the students would be able to:

- ❒ To construct dictionaries and access entries in those dictionaries
- ❒ To use methods to manipulate dictionaries
- ❒ To decide whether a list or a dictionary is an appropriate data structure for a given application.

PROBLEMS

1# A DevOps engineer is managing a server configuration object and needs to verify how Python dictionaries handle access, modification, and deletion. Assume the variable server_config is initialized as: {"timeout": 300, "status": "active"}. Write a Python script that acts as a "Debug Console" to perform the following operations in order:

- Read & Inspect:
    1. Print the value associated with the key "status".
    2. Safely try to get the value for the key "admin_email", defaulting to "Not Set" if it doesn't exist.
    3. Print the total number of settings (len) in the dictionary.

4. Print a list of all Keys and a list of all Values.

- Modify:
  1. Replace the value of "timeout" with its negative (e.g., -300).
  2. Add a new key/value pair: "max_connections": 100.
- Clean Up: Remove the "timeout" key safely from the dictionary.
- Sort: Print the remaining keys of the dictionary in alphabetical order.

2# A linguistic research tool needs to analyze the character distribution of different languages. Write a Python program to read a text string and create a Histogram (dictionary) that counts how often each letter appears.

Example 1:

• Input: "mississippi"

• Output: {'m':1,'i':4,'s':4,'p':2}

Example 2:

• Input: "hello world"

• Output: {'h':1,'e':1,'l':3,'o':2,' ':1,'w':1,'r':1,'d':1}

3# A corporate security system stores employee data as {Name: Badge_ID}. However, the security guard often finds a lost badge and needs to identify the owner using only the ID number.

Write a Python program to perform a Reverse Lookup:

- Input: A specific Badge ID (Value).
- Action: Search through the dictionary to find and return the corresponding Name (Key).
- Constraint: Handle the case where the ID does not exist.

Example:

- Data: {'Alice': 5001, 'Bob': 5002, 'Charlie': 5003}
- Input: 5002 -> Output: "Bob"
- Input: 9999 -> Output: "ID not found"

4# An e-commerce pricing engine needs to apply a "Flash Sale". Create a dictionary Catalog_Prices containing the standard price of all items. You

receive a second dictionary Sale_Updates that contains new prices *only* for items on sale. Write a Python script to merge Sale_Updates into Catalog_Prices.

- Crucial Requirement: If an item exists in both, the value from Sale_Updates must overwrite the value in Catalog_Prices.

Key Point: Merging updates existing data. The new value wins during a collision.

Example:

- Input:
  - Catalog = {'TV': 50000, 'Mouse': 500}
  - Sale = {'TV': 45000}
- Output: {'TV': 45000, 'Mouse': 500} *(TV price is updated, Mouse remains same)*

5# A Math Learning App needs a quick lookup table for squares to speed up calculations for students. Write a Python script to generate a dictionary, take user input for range of keys:

- Keys: Integers from 1 to 15 (inclusive).
- Values: The square of the key (key^2).

Example:

- Output Snippet: {1: 1, 2: 4, 3: 9 ... 15: 225}

6# Implement the Matrix Chain Multiplication problem using dynamic programming in Python.

7# A GPS Navigation System needs to calculate the fastest route between cities connected by highways with different distances (weights). Write a Python program to find the Shortest Path from a starting node to a destination node in a weighted graph using Dijkstra's Algorithm

WEEK #13 & #14

OBJECTIVES

- ❒ To learn the concept Algorithm Analysis

OUTCOMES

After completing this, the students would be able to:

- ❒ To understand and implement MSP TSP and vertex cover problems.
- ❒ To write, debug and run related programs.

PROBLEMS

1# A government infrastructure agency plans to connect a set of remote islands with underwater power cables. They have a list of possible connections and the cost (distance) for each. Write a Python program to implement Kruskal's Algorithm to find the Minimum Spanning Tree (MST).

- Goal: Connect *all* islands using the minimum total length of cable, avoiding any redundant loops.

2# A logistics company uses a delivery drone that must visit a set of customer locations to drop off packages and then return to the central warehouse. The drone has limited battery life, so distance efficiency is critical. Write a Python program to implement the Traveling Salesman Problem (TSP) (using a brute-force or heuristic approach).

- Goal: Find the shortest possible route that visits every location exactly once and returns to the start.

3# A museum security team needs to place guards at hallway intersections (vertices) to ensure every corridor (edge) is monitored by at least one guard. Since hiring guards is expensive, they want to minimize the number of guards. Write a Python script to approximate a solution to the Vertex Cover Problem using a Greedy Algorithm. (Logic: Repeatedly pick the intersection connected to the most uncovered corridors until all corridors are covered.)

INDEX

WEEK NO.		PROBLEMS WITH DESCRIPTION	PAGE NO.	SIGNATURE OF THE TEACHER WITH DATE
1	1#	Discuss the process by which Python executes a program, focusing on the stages involved from source code to execution.		
	2#	Discuss the fundamental features of the Python programming language that make it widely used for modern software development.		
	3#	Install IDLE interpreter for Python programming by following the mentioned steps.		
2	1#	Use interactive Shell to print: User said "Great app!" and directory paths.		
	2#	A shopkeeper uses Python shell for quick billing calculations. Evaluate various arithmetic expressions (addition, subtraction, multiplication, division, modulus).		
	3#	A college form stores student names. Create variables firstName and lastName with your own names.		
	4#	Print firstName and lastName in a single line using f-strings.		
	5#	A heritage museum kiosk needs an automated display-and-billing script. Create a single Python file kioskSystem.py that performs billing calculations and displays information messages.		
	6#	An engineer calculates load using a polynomial. Given specific variable values, compute the result of the algebraic expression.		
	7#	Write a program to compute the value of a specific algebraic expression where x and y are read using the input() function.		
	8#	A technician optimizes multiplication using bitwise operations. Read an integer and multiply it by 2 using the left shift operator (<<).		
3	1#	A bookstore's billing app needs a quick utility script. Write a Python program that reads N integers (prices) and computes: Sum, Product, and Average.		
	2#	A weather station needs temperature conversion. Write a program to convert Celsius to Fahrenheit.		
	3#	An architecture tool calculates dimensions. Write a program to compute the Surface Area and Volume of a Cuboid given length, breadth, and height.		
	4#	Write a program to add two numbers without using arithmetic operators, using bitwise operators.		

WEEK NO.	PROBLEMS WITH DESCRIPTION		PAGE NO.	SIGNATURE OF THE TEACHER WITH DATE
3	5#	A hardware technician tests divider circuits. Read an integer and divide it by 4 using the bitwise >> operator.		
	6#	A robotics controller swaps sensor IDs efficiently. Read two integers and exchange their values using the XOR (^) operator.		
	7#	A gaming console optimizes score calculation. Multiply a number by 10 using bitwise operators.		
	8#	A data-logging device swaps configuration settings. Read two integers and exchange their values using addition and subtraction (avoid using a third variable).		
	9#	A biology simulation needs repeated multiplication. Write a program to compute the factorial of an integer using a loop.		
	10#	A math module checks sequence validity. Write a program to determine whether a number is a Fibonacci number.		
	11#	A security system verifies prime-number-based keys. Write a program to check whether an integer is prime, or else output its first factor.		
	12#	A signal-processing unit finds harmony between two frequencies. Write a program to compute the GCD of two numbers.		
4	1#	A teacher wants to automate small mathematical tasks. Define functions to: compute average of marks, convert temperature, and calculate perimeter of a rectangle.		
	2#	A science lab technician is calculating the volume of spherical containers. Write a function that accepts the radius and returns the volume of the sphere.		
	3#	A landscaping company maintains circular flower beds. Write a function that takes outer radius R and inner radius r, validates them, and returns the effective usable area.		
	4#	A data analyst needs factorial values to compute permutations. Write a Python function to calculate the factorial of a non-negative integer.		
	5#	An accountant wants a utility that extracts a digit-sum check value. Write a Python function that returns the sum of digits of a given integer.		
	6#	A math research assistant is verifying test values. Write a Python function to check whether a number is a Perfect Number.		
	7#	Write a Python function to check whether a given number is a prime number or not.		
	8#	A text-analysis tool needs statistics from a sentence. Write a function that accepts a string and counts the number of uppercase and lowercase letters.		

WEEK NO.	PROBLEMS WITH DESCRIPTION		PAGE NO.	SIGNATURE OF THE TEACHER WITH DATE
5	1#	A data automation team is creating a utility module. Write a program that accepts inputs and uses functions to: find largest of three numbers, compute volume of shapes, area of rectangle, circumference of circle, swap variables, and find distance between points.		
	2#	A billing system receives an unknown count of numeric entries. Write a function that accepts arbitrary integers (*args) and returns their sum.		
	3#	A text-filtering tool needs to remove noisy characters. Write a program that takes a string and removes all characters at even index values.		
	4#	Write a Python script that takes input from the user and displays that input back in upper and lower cases.		
	5#	A simple analytics module must calculate keyword frequency. Write a program to count occurrences of each word in a sentence.		
	6#	A document cleanup tool needs to normalize text. Write a program to remove indentation from each line of a multi-line text.		
	7#	Write a function that reverses a string by modifying the input array in-place with O(1) extra memory.		
5	8#	Given a string, reverse the order of characters in each word within a sentence while preserving whitespace and word order.		
	9#	A parser converts messages into tokens. Write a program to convert a string into a list of characters.		
	10#	Write a Python program to count and display all vowels present in the text, as well as consonants and their count.		
6	1#	A grading system compares three internal-assessment scores. Write a program to find the highest score among the three using only if statements.		
	2#	A sensor module collects readings from three probes. Write a program to determine the lowest reading using if–else logic.		
	3#	Write a program to check if a year is a leap year.		
	4#	Write a program to print the number of days in a month using chained conditionals.		
	5#	Write a program that takes a month and year, and prints the correct number of days, accounting for leap years.		
	6#	An architectural software checks triangular panels. Write a program to compute area and identify if a triangle is equilateral, isosceles, or scalene.		
	7#	A device logs data indexes. Given two numbers r1 and r2, create a list of integers from r1 to r2 using range().		

WEEK NO.	PROBLEMS WITH DESCRIPTION		PAGE NO.	SIGNATURE OF THE TEACHER WITH DATE
	8#	A parser receives a mixed list. Write a program to extract and add only numeric items using isinstance().		
	9#	A game engine verifies level IDs. Write a program to check whether a given list of numbers is consecutive.		
	10#	FizzBuzz: Given integer n, return a string array where multiples of 3 are "Fizz", 5 are "Buzz", and both are "FizzBuzz".		
7	1#	A cryptographic security module uses "Narcissistic Numbers". Write a program using a while loop to check if a numeric key is an Armstrong number.		
	2#	A biological simulation tool models cell growth. Write a recursive function to generate the first n terms of the Fibonacci sequence.		
	3#	A data science team needs to calculate permutations. Write a recursive function calculate_routes(n) to compute the factorial of a number.		
	4#	A telecommunications engineer synchronizes signals. Write a recursive function find_signal_sync(a, b) to find the GCD using the Euclidean algorithm.		
	5#	Write a program to print a specific inverted right-angle triangle pattern using a for loop.		
	6#	A network packet analyzer receives packet IDs. Filter these into two lists: even IDs (low priority) and odd IDs (high priority).		
	7#	Given an integer x, return true if x is a palindrome, and false otherwise.		
7	8#	An inventory management system consolidates records. Merge two lists of product IDs and display the final list in sorted order.		
	9#	An algorithm visualization tool demonstrates sorting. Write a program to find the second largest number in a list using the Bubble Sort algorithm.		
8	1#	A file management system needs to organize filenames. Write a program to sort a list of strings based on string length.		
	2#	A marketing team has two email lists. Create a master list containing all unique emails from both lists (Union).		
	3#	A cyber-security tool compares server logs. Find the common IP addresses between two lists (Intersection).		
	4#	A data visualization tool needs to plot a curve. Generate a list of tuples where the first element is a number and the second is five times its cube.		
	5#	An NLP pre-processor needs to clean text. Use string methods to tokenize, normalize, find index, search for characters, and sanitize text.		

WEEK NO.	PROBLEMS WITH DESCRIPTION		PAGE NO.	SIGNATURE OF THE TEACHER WITH DATE
	6#	A data compression utility analyzes entropy. Calculate the frequency of every character present in a given string.		
	7#	A simple encryption tool obfuscates passwords. Write a program to reverse the order of characters in a string.		
	8#	A product labeling system generates "SKU codes". Generate a new string made of the first 2 and last 2 characters of a product name.		
	9#	A library catalog system contains sorted Book IDs. Implement Binary Search to find a specific Book ID.		
	10#	An educational algorithm visualizer needs to show sorting steps. Implement Insertion Sort and print the state of the list after every iteration.		
9 & 10	1#	A financial risk analysis tool needs to determine volatility. Identify the Maximum and Minimum values in a list of daily closing prices.		
	2#	An inventory checking system allows a store manager to scan a product ID. Write a program to search for a specific element in a list of product IDs.		
9 & 10	3#	A music playlist app needs to organize songs. Sort a raw list of song titles alphabetically.		
	4#	An election tally system has a dictionary of votes. Sort the dictionary to display the winner and the candidate with the fewest votes.		
	5#	A Data Science team is compiling a "Complete Patient Record". Write a script to concatenate three distinct dictionaries into one single Patient Profile.		
	6#	An embedded system needs to sort a small buffer. Implement the Optimized Bubble Sort algorithm (using a flag mechanism).		
	7#	A Big Data processing engine needs to sort a massive log file. Implement the Merge Sort algorithm using recursion.		
	8#	A real-time gaming engine needs a high-speed sorting algorithm. Implement the Quick Sort algorithm.		
11 & 12	1#	A DevOps engineer manages server configuration. Write a script to inspect, modify, clean up, and sort a dictionary of settings.		
	2#	A linguistic research tool analyzes character distribution. Create a Histogram (dictionary) that counts how often each letter appears in a string.		
	3#	A corporate security system stores employee data. Write a program to perform a Reverse Lookup (find Key by Value) for a badge ID.		

WEEK NO.	PROBLEMS WITH DESCRIPTION		PAGE NO.	SIGNATURE OF THE TEACHER WITH DATE
	4#	An e-commerce pricing engine applies a "Flash Sale". Merge a Sale_Updates dictionary into a Catalog_Prices dictionary, overwriting existing values.		
	5#	A Math Learning App needs a lookup table. Generate a dictionary where keys are integers 1-15 and values are their squares.		
	6#	Implement the Matrix Chain Multiplication problem using dynamic programming.		
	7#	A GPS Navigation System calculates routes. Find the Shortest Path in a weighted graph using Dijkstra's Algorithm.		
13 & 14	1#	A government agency connects remote islands. Implement Kruskal's Algorithm to find the Minimum Spanning Tree (MST).		
	2#	A logistics company uses a delivery drone. Implement the Traveling Salesman Problem (TSP) to find the shortest route visiting all locations.		
	3#	A museum security team places guards at intersections. Approximate a solution to the Vertex Cover Problem using a Greedy Algorithm.		