**“Software Design And Construction”**

**BY:** Bushra Ashraf Bhatti

**Reg\_no:** 2023-BSE-015

**Submitted to:** Engr. Muhammad Shahzad

**Lab tittle:** Functions & Modules

**Functions:**  
A **function** is a block of reusable code that performs a specific task.  
🔹 You define it using def.  
🔹 It helps avoid repeating code.

For example:

def my\_function(country = "Norway"):

    print("I am from " + country)

my\_function("Sweden")

my\_function("India")

my\_function()

my\_function("Brazil")

**output:**



**Passing a List as an Argument:**

def my\_function(food):

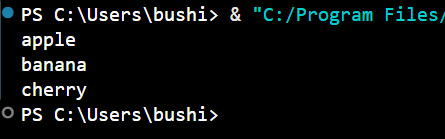
    for x in food:

        print(x)

fruits = ["apple", "banana", "cherry"]

my\_function(fruits)

**output:**



**Python Library Functions:**

**Python Library Functions** are **built-in functions** that come with Python. They are **predefined and ready to use**, we don’t have to write their code yourself.

**For example:**

import math

num = 25

print("Square root of", num, "is:", math.sqrt(num))

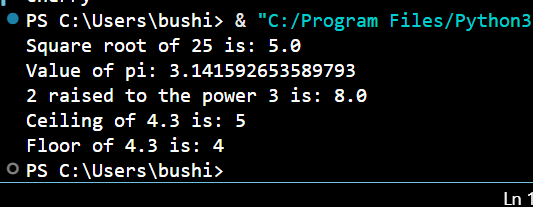
print("Value of pi:", math.pi)

print("2 raised to the power 3 is:", math.pow(2, 3))

print("Ceiling of 4.3 is:", math.ceil(4.3))

print("Floor of 4.3 is:", math.floor(4.3))

**output:**



**Recursive function:**

**Example 1:**

def factorial(n):

    if n == 0:  # Base case

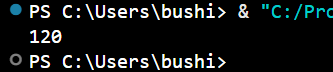
        return 1

    else:

        return n \* factorial(n - 1)  # Recursive call

print(factorial(5))  # Output: 120

**output:**



**Example 2:**

def factorial(x):

    """This is a recursive function

    to find the factorial of an integer"""

    if x == 1:

        return 1

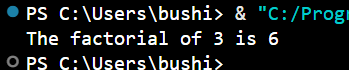
    else:

        return x \* factorial(x - 1)

num = 3

print("The factorial of", num, "is", factorial(num))

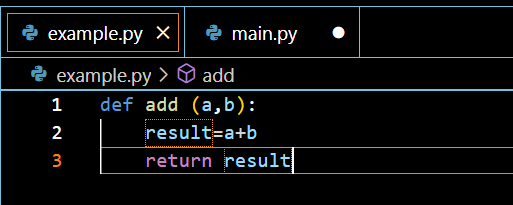
**output:**



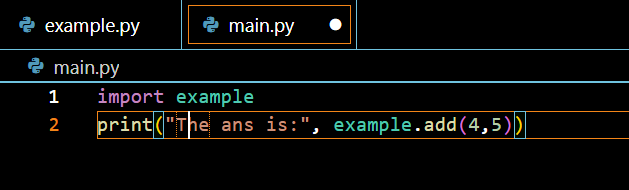
**Modules:**  
A **module** is a **file** that contains Python code (functions, variables, or classes) which you can import and reuse in other programs.  
🔹 Each .py file is a module.  
🔹 You use the import statement to use it.

Module is a file that contains code to perform a specific task. A module may contain variables, functions, classes etc. Let's see an example:

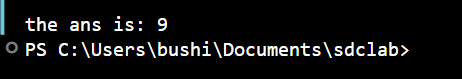
Let us create a module. Type the following and save it as example.py.



Make another file:



**And run:**



**Lab tasks**

**Task 1.**

Understand all the examples and implement all the programs.

(DONE ABOVE)

**Task 2.**

String Reversal: Write a Python function called reverse\_string that takes one argument, a string, and returns a new string with the characters in reverse order. For example, reverse\_string("hello") should return "olleh".

**Code:**

def reverse\_string(s):

    """This function takes a string and returns it in reverse order."""

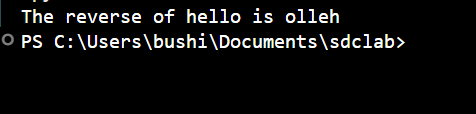
    return s[::-1]

word = "hello"

reversed\_word = reverse\_string(word)

print("The reverse of", word, "is", reversed\_word)

**output:**



**Task 3.**

Even/Odd Checker: Create a Python function named is\_even that accepts a single integer argument. The function should use the modulo operator (%) and return True if the number is even and False if the number is odd.

**Code:**

#-------------------------------------------

def is\_even(number):

    """

    This function takes an integer as input.

    It returns True if the number is even, and False if it is odd.

    """

    if number % 2 == 0:

        return True

    else:

        return False

num = int(input("Enter an integer: "))

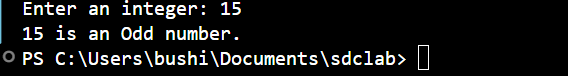
if is\_even(num):

    print(num, "is an Even number.")

else:

    print(num, "is an Odd number.")

**output:**



**Task 4.**

Develop a Python function named sum\_list that takes a single argument, which is a list of numbers. The function must iterate through the list and return the sum of all the numbers within the list. You must not use the built-in sum() function.

**Code:**

def sum\_list(numbers):

    """

    This function takes a list of numbers as input.

    It adds each number in the list to a total variable

    and finally returns the total sum.

    """

    total = 0

    for num in numbers:

        total += num

    return total

my\_numbers = [5, 10, 15, 20, 25]

result = sum\_list(my\_numbers)

print("List of numbers:", my\_numbers)

print("Sum of all numbers:", result)

**output:**



**Task 5.**

Area of a Rectangle: Define a Python function called calculate\_rectangle\_area that requires two arguments: length and width (both expected to be numbers). The function should calculate and return the area of the rectangle (length width).

**Code:**

def calculate\_rectangle\_area(length, width):

    """

    This function takes two arguments:

    - length: the length of the rectangle

    - width: the width of the rectangle

    It returns the area of the rectangle using the formula:

        area = length \* width

    """

    area = length \* width

    return area

length = float(input("Enter the length of the rectangle: "))

width = float(input("Enter the width of the rectangle: "))

result = calculate\_rectangle\_area(length, width)

print("----------------------------------")

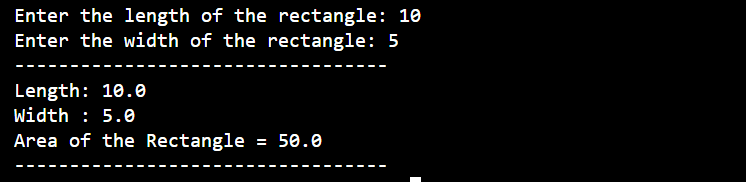
print("Length:", length)

print("Width :", width)

print("Area of the Rectangle =", result)

print("----------------------------------")

**output:**



**Task 6.**

Implement a Python function named count\_vowels that takes a string as input. The function should count how many vowels (a, e, i, o, u, case-insensitive) are present in the string and return the total count.

**Code:**

def count\_vowels(text):

    """

    This function takes a string as input and counts how many vowels

    (a, e, i, o, u) are present in it, regardless of uppercase or lowercase.

    It returns the total count of vowels.

    """

    vowels = "aeiou"

    count = 0

    text = text.lower()

    for char in text:

        if char in vowels:

            count += 1

    return count

user\_input = input("Enter a string: ")

vowel\_count = count\_vowels(user\_input)

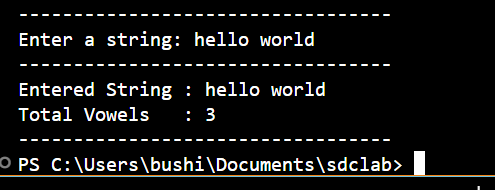
print("----------------------------------")

print("Entered String :", user\_input)

print("Total Vowels   :", vowel\_count)

print("----------------------------------")

**output:**



**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***