Task: 3-1
Date: 13/8/25

Importing Python modules and Packages in Python Programming

AIMS:-

Write a program to Perform mathematical calculation using math module.

Algorithm:-

Step 1:- Open Python script file
Step 2:- Import math module
Step 3:- math.log10 used to find log base 10
Step 4:- ** Used to calculate Power
Step 5:- math.floor(), math.ceil() used to calculate floor & ceil value.
Step 6:- abs() method used to display absolute value.
Step 7:- Factorial() method used to display the factorial value.

Input/Output:-

Logarithm 1.1139433523068364
Power 100.0
Floor 10
ceil 11
absolute 10.024
Factorial 120

RESULT:-

Thus a Program to perform mathematical calculation using math module is compiled & executed succesfully.

Task: 3-1
Date: 13/8/25

PROGRAMS:-

```
import math
Print("Logarithm", math.log10(100))
Print("Power", math.pow(10,2))
Print("Floor", math.floor(10.25201))
Print("ceil", math.ceil(10.25201))
Print("absolute", abs(-10.024))
Print("Factorial", math.factorial(5))
```

TASK No: 3.2
DATE:- 13/8/25

TOTAL WEIGHT CALCULATION

AIM :-

To write a Python program using packages & modules that reads the number of widgets & gizmos sold by an online retailer, calculates their total weight based on predefined weights, and displays the result to the user.

ALGORITHM:-

Step 1 :- Start

Step 2 :- Create a package folder named shop with:

- A module file weight_calc.py containing a function to calculate total weight
- An __init__.py file to make it a package.

Step 3 :- In weight_calc.py,

- Define constants:

  WIDGET_WEIGHT = 75 grams and GIZMO_WEIGHT = 112 grams

- Create a function calculate_total_weight that :-
  - Multiplies the number of widgets by WIDGET_WEIGHT.
  - Multiplies the number of gizmos by GIZMO_WEIGHT
  - Adds the two results toget total_weight
  - Returns total_weight

PROGRAMS:-

```
shop/
    __init__.py
    weight_calc.py

main.py
```

1. Shop/weight_calc.py

```
#weight_calc.py
def calculate_total_weight(widgets, gizmos):
    WIDGET_WEIGHT = 75
    GIZMO_WEIGHT = 112
    total_weight = (widgets * WIDGET_WEIGHT) +
                   (gizmos * GIZMO_WEIGHT)
    return total_weight
```

2. Shop/init.py

```
# __init__.py
# This makes 'shop' a package
```

3. main.py

```
#main.py
from shop import weight_calc
def main():
    widgets = int(input("Enter the number of widgets:"))
    gizmos = int(input("Enter the number of gizmos:"))
    total_weight = weight_calc.calculate_total_weight
                   (widgets, gizmos)
```

AIM :-

To

modules

sold by

weight

the

ALGOR

Step 1

Step 2

Step

Step 4:- In the main program

- Import the weight_calc module from the shop package.
- Prompt the user to enter the number of widgets
- Prompt the user to enter the no. of gizmos
- Call the calculate_total_weight() function with these inputs.
- Display the total weight in grams.

Step 5:- End

Input / Output :-

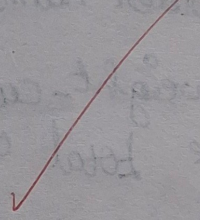
Enter the number of widgets : 5
Enter the number of gizmos : 3
The total weight of the parts is 731 grams

RESULT:-

Thus a Python Program to calculate the total weight is compiled and executed successfully.

```
    print(f"The total weight of the parts is
          {total_weight} grams.")

if __name__ == "__main__":
    main()
```



TASK : 8-3
DATE: 13/8/25

OS and Sys Modules

AIM :-

Write a Program to display os and sys details using os and sys module

ALGORITHMS :-

Step 1 :- open python script file
Step 2 :- import os and sys module
Step 3 :- os.name to display the name of the os
Step 4 :- os.getcwd() to display current working directory
Step 5 :- os.listdir() to display the directory list details
Step 6 :- Sys.platform used to display linux platform
Step 7 :- Platform.system(), platform.release()) method to display the platform details & platform release

OUTPUT :-

os.name nt /tmp/sessions/8ab0b2fead14b3db
['main.py']
<built-in function getlogin>
Posix.uname_result (sysname='Linux', nodename='5c10b589996b', release='5.11.0-1017-gcp', version='#19~20.04.1-Ubuntu SMP Thu Aug 12 05:25:25 UTC 2021', machine='x86_64')
Sys.platform Win32
Platform.System() windows sys.release() 11

RESULT :-

Thus a Program to display os and Sys modules is compiled and executed succesfully

PROGRAMS:-

```
import os
import sys
import platform
import sysconfig
print("os name:", os.name)
print(os.getcwd())
print(os.listdir())
print(os.getlogin())
print("platform name:", sys.platform)
print("platform system:", platform.system())
print("platform release():", platform.release())
```

TASK : 3.4
DATE: 13/8/25

RANDOM WISH

Aim:-

Write a program to display unique number wishes.

Algorithm:-

Step 1:- Open Python script file
Step 2:- import random module
Step 3:- list the wishes in fortunes
Step 4:- Random.choice() using to display the random wishes

PROGRAM:

Sample Output :-

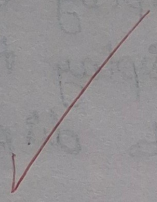
Patience is a virtue

RESULT:-

Thus a Python Program has been compiled & executed successfully.

PROGRAMS:-

```
import random
fortunes = [ "Good things come to those who wait!",
"patience is a virtue", "The early bird gets the
worm.", "A wise man once said everything
in its own time & place", "Fortune cookies
scarcely share fortunes." ]
Print (random.choice (fortunes))
```



TASK : 3.
DATE : 13/

AIM:-

wishes

Algo

TASK: 3.5
DATE: 13/8/25

MATHEMATICAL & STRING OPERATIONS

Aim:-

To write a Python program using packages and modules that perform mathematical operations (addition, multiplication, factorial) and string operations by organizing code into reusable components.

Algorithm:-

Step 1:- Create the package structure

1) Create a folder named my-package
2) Inside my-package, create the files:
   - __init__.py
   - math_utils.py
   - string_utils.py

Step 2:- Implement math_utils.py Module

- Define a function add(a,b) to return the sum of two numbers.
- Define a function multiply(a,b) to return the product of two numbers.

Step 3:- Implement string_utils.py Module

- Define a function reverse_string(s) that returns the reversed string using slicing (s[::-1]).
- Define a function count_vowels.

PROGRAMS:-

TASK: 3.5
DATE: 13/8/25

1. Folder structure :

```
my_package/
__init__.py
math_utils.py
string_utils.py

main.py
```

my_package/math_utils.py

```
# math_utils.py
def add(a,b):
    return a+b

def multiply(a,b):
    return a*b

def factorial(n):
    if n==0 or n==1:
        return 1
    return n * factorial(n-1)
```

my_packages/string_utils.py

```
# string_utils.py
def reverse_string(s):
    return s[::-1]

def count_vowels(s):
    vowels = "aeiouAEIOU"
    return sum(1 for char in s if char in
               vowels)
```

my_package/__init__.py

Aim:-
To write
and modules
(addition, mu
by organizi

Algorithm:-
Step 1:- C

step 2:

Step 4:- Initialize package with __init__.py

Step 5:- Create main.py program

- Import required function from my-packages call & display result.

Step 6:- Execute the program

- Run python main.py in the terminal
- Verify that outputs match expected results

OUTPUT:-

Addition : 15
Multiplication : 12
Factorial of 5 : 120
Reversed string : nohtyP
Number of vowels : 3

25/8

VELTECH	
Ex No.	3
PERFORMANCE (5)	5
RESULT AND ANALYSIS (5)	5
VIVA VOCE (5)	5
RECORD (5)	5
TOTAL (20)	20
SIGN WITH DATE	

RESULT:-

Thus a Python program has been compiled and executed successfully.

```
#__init__.py
from .math_utils import add, multiply, factorial
from .string_utils import
reverse_string, count_vowels
```

2. Main Program

```
# main.py
From my_package import add, multiply, factorial,
    reverse_string, count_vowels

# MATH Function

Print ("Addition:", add (10, 5))
Print ("Multiplication:", multiply (3, 4))
Print ("Factorial of 5:", factorial (5))

# String Function

Print ("Reversed string:", reverse_string ("python"))
Print ("Number of vowels:", count_vowels ("Hello
                                    world"))
```

Step 4:-

Step 5:-

Step 6:-