Index

Sr.No	Task	Date	Page No.	Teacher Sign
1.	Write a program to demonstrate different number data types in Python.			
2.	Write a program to perform different Arithmetic Operations on numbers in Python.			
3.	Write a program to create, concatenate and print a string and accessing sub-string from a given string.			
4.	Write a python script to print the current date in the following format "Sun May 29 02:26:23 IST 2017"			
5.	Write a program to create, append, and remove lists in python.			
6.	Write a program to demonstrate working with tuples in python.			
7.	Write a program to demonstrate working with dictionaries in python.			
8.	Write a python program to find largest of three numbers.			
9.	Write a Python program to convert temperatures to and from Celsius, Fahrenheit. [Formula: c/5 = f-32/9]			
10.	Write a Python program to construct the following pattern, using a nested for loop * * * * * * * * * * * * * * * * *			
11.	Write a Python script that prints prime numbers less than 20.			
12.	Write a python program to find factorial of a number using Recursion.			
13.	Write a program that accepts the lengths of three sides of a triangle as inputs. The program output should indicate whether or not the triangle is a right triangle			

	(Recall from the Pythagorean Theorem that in a right triangle, the square of one side equals the sum of the squares of the other two sides).			
14.	Write a python program to define a module to find Fibonacci Numbers and import the module to another program.			
15.	Write a python program to define a module and import a specific function in that module to another program.			
16.	Write a script named copyfile.py. This script should prompt the user for the names of two text files. The contents of the first file should be input and written to the second file.			
17.	Write a program that inputs a text file. The program should print all of the unique words in the file in alphabetical order.			
18.	Write a Python class to convert an integer to a roman numeral.			
19.	Write a Python class to implement pow(x, n)			
20.	Write a Python class to reverse a string word by word.			

TASK 1

AIM: PROGRAM TO DEMONSTRATE DIFFERENT NUMBERS OF DATA TYPES IN PYTHON

INPUT:

```
#int
a = 12
print("integer value: ",a)          #integer value: 12
print("Type is: ", type(a) ,"\n")   #Type is: <class int 'int'>

#float
c = 3.09
print("float value: ",c)           #float value: 3.09
print("Type is: ", type(c), "\n")  #Type is: <class 'float'>

#complex
b = 3 + 5j
print("complex value: ", b)        #complex value: (3+5j)
print("Type is: ", type(b), "\n")  #Type is: <class 'complex'>
```

OUTPUT:

```
integer value:  12
Type is:  <class 'int'>

float value:  3.09
Type is:  <class 'float'>

complex value:  (3+5j)
Type is:  <class 'complex'>
```

TASK 2

AIM: PROGRAM TO PERFORM DIFFERENT ARITHMETIC OPERATIONS ON NUMBERS IN PYTHON

INPUT:

```
number_1 = 20
number_2 = 10

print()
print("first number: ", number_1)          #first  number: 20
print("second number: ", number_2)         #second number: 10

#Addition
print("Addition: " ,number_1 + number_2)    #Addition: 30

#subtraction
print("subtraction: " ,number_1 - number_2) #Subtraction: 10

#Multiplication
print("Multiplication: " ,number_1 * number_2)  #Multiplication: 200

#Division
print("Division: " ,number_1 / number_2)        #Division: 2.0

#flow division
print("Flow division: " ,number_1 // number_2)  #Flow division: 2

#Modulus
print("Modulus: " ,number_1 % number_2)         #Modulus: 0

#Power
print("Power: " ,number_1 ^ number_2)           #Power: 30
```

OUTPUT:

```
first number:  20
second number:  10

Addition:  30

subtraction:  10

Multiplication:  200

Division:  2.0

Flow division:  2

Modulus:  0

Power:  30
```

TASK 3

AIM: PROGRAM TO CREATE, CONCATENATE AND PRINT A STRING AND ACCESSING SUB-STRING FROM A GIVEN STRING.

INPUT:

```
print("#Three methods to create a string-------\n")       -

#method 1
method_1 = "\"Abhishek\""
print("Method 1- (double quotes method): ", method_1)
#Method 1- (double quotes method):  "Abhishek"

#method 2
method_2 = '\'Tiwari\"
print("Method 2- (single quotes method): ", method_2)
#Method 2- (single quotes method):  'Tiwari '

#method 3
method_3 = """\"""This is multi-line string
Line 2 of multiline string\""" """
print("Method 3- (multiline string method): ", method_3)
#Method 3- (multiline string method):  """This is multi-line string
#Line 2 of multiline string"""

print("\n#String Operations--------- \n")

string_1 = "Hawlet"
string_2 = "Packered"
print("String 1: ", string_1)
print("String 2: ", string_2)

#concatenation
concatenated = string_1 + " " + string_2
print("Concatenated String: ",concatenated)
#Concatenated String:  Hawlet Packered

#accesing substring
substring = concatenated[0:5]
print("Substring (first 5 characters): ", substring)
#Substring (first 5 characters):  Hawle

#using negative indexing
substring_negative = concatenated[-5:]
print("Substring negative (last 5 characters): ", substring_negative)
 #Substring negative (last 5 characters):  kered
```

OUTPUT:

```
#Three methods to create a string-------

Method 1- (double qoutes method):  "Abhishek"
Method 2- (single qoutes method):  'Tiwari'
Method 3- (multiline string method):  """This is multi-line string
Line 2 of multiline string"""


#String Operations---------

String 1:  Hawlet
String 2:  Packered

Concatednated String:  Hawlet Packered
Substring (first 5 characters):  Hawle
Substring negative (last 5 characters):  kered
```

TASK 4

AIM: WRITE A PYTHON SCRIPT TO PRINT THE CURRENT DATE IN THE FOLLOWING FORMAT “SUN MAY 29 02:26:23 IST 2017”

INPUT:

```
#importing time library
import time

print()
#fetching and formatting time
print("Date and time fetched from OS")
current_time = time.strftime("%a %b %d %H:%M:%S %Z %Y")

#printing date and time
print("current date and time is: ",current_time)   #current date and time is:  Wed Aug 28 23:04:51 India Standard Time 2024
```

OUTPUT:

```
Date and time fetched from os
current date and time is:  Wed Aug 28 23:06:38 India Standard Time 2024
```

TASK 5

AIM: WRITE A PROGRAM TO CREATE, APPEND, AND REMOVE LISTS IN PYTHON.

INPUT:

```
#list creation
print("#Creating a list--------- ")
list = [1,2,3,4]
print("My list: ",list)   #My list:  [1, 2, 3, 4]

#append list
print("Append 5 on list")
list.append(5)
print(list)         #Append 5 on list

#extend list
print("Extend list with [6,7,8]")
list.extend([6,7,8])
print(list)        #[1, 2, 3, 4, 5, 6, 7, 8]

#pop from list
print("#Popping an element---------")
print("last element popped", list.pop())
print(list)        #[1, 2, 3, 4, 5, 6, 7]
print("element at index 2 popped: ", list.pop(2))
print(list)        #[1, 2, 4, 5, 6, 7]

#insert element to list
print("#inserting an element---------")
print("23 inserted at 4th index")
list.insert(4,23)
print(list)        #[1, 2, 4, 5, 23, 6, 27]

#deleting element form list
print("#Deleting an element---------")
del list[4]
print("deleted from 4th index ")
print(list)        #[1, 2, 4, 5, 6, 7]
```

OUTPUT:

```
#Creating a list---------
My list:  [1, 2, 3, 4]

Append 5 on list
[1, 2, 3, 4, 5]

Extend list with [6,7,8]
[1, 2, 3, 4, 5, 6, 7, 8]

#Popping an element---------
last element popped 8
[1, 2, 3, 4, 5, 6, 7]
element at index 2 popped:  3
[1, 2, 4, 5, 6, 7]

#inserting an element---------
23 inserted at 4th index
[1, 2, 4, 5, 23, 6, 7]

#Deleting an element---------
deleted from 4th index
[1, 2, 4, 5, 6, 7]
```

TASK 6

AIM: WRITE A PROGRAM TO DEMONSTRATE WORKING WITH TUPLES IN PYTHON.

INPUT:

```
print("#Creating new tupple")
tuple = (1,2,3,4)
print("My tuple: ", tuple)   #(1, 2, 3, 4)

print("#accessing element from tuple")
print("value at index 3 is ", tuple[3])   #4

#tuple slicing
print("Slicing tuple [3:6]: ", tuple[1:3])   #(2, 3)

#unpacking
a,b,c,d = tuple
print("unpacked_tuple : ", a,b,c,d)    #1 2 3 4

#Concatenation
concatenated_tuple = tuple + ('a','b','c','d')
print("Concatenated tuple: ", concatenated_tuple)       #(1, 2, 3, 4, 'a', 'b', 'c', 'd')

#check element is exist in tuple or not using Membership operator
print("let's check 4 in exist in my concatenated_tuple ")
if (4 in concatenated_tuple):
   print("Yes 4 is found in concatenated_tuple")          #true condition
else:
   print("No 4 is not found")                              #false condition
```

OUTPUT:

```
#Creating new tupple
My tupple:  (1, 2, 3, 4)

#accessing element from tupple
value at index 3 is  4

Slicing tupple [3:6]:  (2, 3)

unpacked_tupple :  1 2 3 4
Concatenated tupple:  (1, 2, 3, 4, 'a', 'b', 'c', 'd')

let's check 4 in exist in my concatenated_tupple
Yes 4 is found in concatednated_tupple
```

TASK 7

AIM: WRITE A PROGRAM TO DEMONSTRATE WORKING WITH DICTIONARIES IN PYTHON.

INPUT:

```
#creating key value pair
print("\nMy Dictionary")
student = {
   'Name':'abhishek',
   'Branch':'BTech CSE',
   'Roll No.':'2207775'
   }
print(student,'\n')

#Accessing values by key
print("Value of Name: ", student['Name']) #Value of Name:  abhishek
print()

#add or update keys
student['Department'] = 'UIET'
print("New key aded: ", student)
#{'Name': 'abhishek', 'Branch': 'BTech CSE', 'Roll No.': '2207775', 'Department': 'UIET'}
student['Roll No.'] = '2207778'
print("Roll No. changed: ", student)
#{'Name': 'abhishek', 'Branch': 'BTech CSE', 'Roll No.': '2207778', 'Department': 'UIET'}
print()

#removing key:value pair using del
del student['Roll No.']
print("Dictionary after deleting Roll No. using del: ", student)
#{'Name': 'abhishek', 'Branch': 'BTech CSE', 'Department': 'UIET'}

#removing key:value pair using pop
student.pop('Branch')
print("Dictionary after deleting Branch using pop: ", student)
#{'Name': 'abhishek', 'Department': 'UIET'}
print()

#printing all keys
print("all keys: ")
for keys in student:
   print(student[keys])
#abhishek
#UIET

print()
#printing all values
print("all values: ")
for values in student.values():
   print(values)
#abhishek
#UIET
```

```
print()
#iterating key:value pairs
print("Looping through key:value pair")
for keys, values in student.items():
    print(keys,values)
#Name abhishek
#Department UIET

print()

#checking if a key is exist
print("let's check a key 'Roll No.' exist in student ")
if ('Roll No.' in student[keys]):
    print("Yeh! key found")          #true condition
else:
    print("Oh no key does'nt exist")     #false condition
```

OUTPUT:

```
My Dictionary
{'Name': 'abhishek', 'Branch': 'BTech CSE', 'Roll No.': '2207775'}

Value of Name:  abhishek

New key aded:  {'Name': 'abhishek', 'Branch': 'BTech CSE', 'Roll No.': '2207775', 'Department': 'UIET'}
Roll No. changed:  {'Name': 'abhishek', 'Branch': 'BTech CSE', 'Roll No.': '2207778', 'Department': 'UIET'}

Dictionary after deleting Roll No. using del:  {'Name': 'abhishek', 'Branch': 'BTech CSE', 'Department': 'UIET'}
Dictionary after deleting Branch using pop:  {'Name': 'abhishek', 'Department': 'UIET'}

all keys:
abhishek
UIET

all values:
abhishek
UIET

Looping through key:value pair
Name abhishek
Department UIET

let's check a key 'Roll No.' exist in student
Oh no key does'nt exist
```

TASK 8

AIM: WRITE A PYTHON PROGRAM TO FIND LARGEST OF THREE NUMBERS.

INPUT:

```
def find_max(a, b, c):
    if (a>=b) & (a>=c):
        return a
    elif (b>=a) & (b>=c):
        return b
    else:
        return c

#main
a = 200
b = 300
c = 150

print("Values: ",a,b,c)
print(f"{find_max(a,b,c)} is largest")   #300 is largest
```

OUTPUT:

```
Values:  200 300 150
300 is largest
```

TASK 9

AIM: WRITE A PYTHON PROGRAM TO CONVERT TEMPERATURES TO AND FROM CELSIUS, FAHRENHEIT. [Formula: c/5 = f-32/9]

INPUT:

```
print("Program of conversion between Celsius and Fahrenheit\n")

#function to calculate celsius to fahrenheit
def celsius_to_fahrenheit(celsius):
    fahrenheit = (celsius * 9/5) + 32
    return fahrenheit

#function to calculate celsius fahrenheit to celsius
def fahrenheit_to_celsius(fahrenheit):
    celsius = (fahrenheit - 32) * 5/9
    return celsius

# Example Conversion
celsius = 25  # Celsius temperature
fahrenheit = 77  # Fahrenheit temperature

# Directly print both conversions
print(f"{celsius}°C is equal to {celsius_to_fahrenheit(celsius):.2f}°F")
#25°C is equal to 77.00°F

print(f"{fahrenheit}°F is equal to {fahrenheit_to_celsius(fahrenheit):.2f}°C")
#77°F is equal to 25.00°C
```

OUTPUT:

```
Program of conversion between Celsius and Fahrenheit

25°C is equal to 77.00°F
77°F is equal to 25.00°C
```

TASK 10

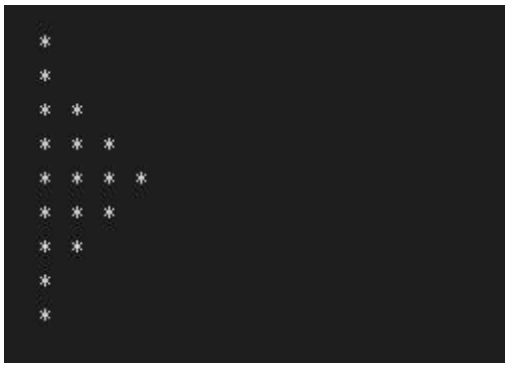
AIM: WRITE A PYTHON PROGRAM TO CONSTRUCT THE FOLLOWING PATTERN, USING A NESTED FOR LOOP.

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

INPUT:

```
n = 4  # The middle row count (max stars)

print("*")

# First part: increasing stars

for i in range(1, n + 1):
    print("* " * i)

# Second part: decreasing stars
for i in range(n - 1, 0, -1):
    print("* " * i)

print("*")
```

OUTPUT:



TASK 11

AIM: WRITE A PYTHON SCRIPT THAT PRINTS PRIME NUMBERS LESS THAN 20.

INPUT:

```
#function to print prime numbers
def is_prime(num):
    if num < 2:
        return False
    for i in range(2, num):
        if num % i == 0:
            return False
    return True

# Print prime numbers less than 20
print("Prime numbers less than 20 are:")
for number in range(2, 20):
    if is_prime(number):
        print(number, end=" ")
# 2 3 5 7 11 13 17 19
```

OUTPUT:

```
Prime numbers less than 20 are:
2 3 5 7 11 13 17 19
```

TASK 12

AIM: WRITE A PYTHON PROGRAM TO FIND FACTORIAL OF A NUMBER USING RECURSION.

INPUT:

```
#function to find factorial of a number
def factorial(num):
    if num == 1:
        return 1
    else:
        return num * factorial(num-1)

#main
number = 5
print(f"factorial of {number} is {factorial(number)}")
#factorial of 5 is 120
```

OUTPUT:

```
factorial of 5 is 120
```

TASK 13

AIM: WRITE A PROGRAM THAT ACCEPTS THE LENGTHS OF THREE SIDES OF A TRIANGLE AS INPUTS. THE PROGRAM OUTPUT SHOULD INDICATE WHETHER OR NOT THE TRIANGLE IS A RIGHT TRIANGLE

INPUT:

```
def is_right_traingle(a, b, c):
    sides = sorted([a,b,c])

    if sides[0]**2 + sides[1]**2 == sides[2]**2:        #Pythagorean Theorem P^2 + B^2 = H^2
        return True
    else:
        return False

#main
a = 3
b = 5
c = 4

if is_right_traingle(a,b,c):
    print(f"The traingle with sides {a ,b, c} is right triangle.")   #True Condition
else:
    print(f"The traingle with sides {a ,b, c} is not a right triangle.")  #False Condition
```

OUTPUT:

```
The traingle with sides (3, 5, 4) is right triangle.
```

TASK 14

AIM: WRITE A PYTHON PROGRAM TO DEFINE A MODULE TO FIND FIBONACCI NUMBERS AND IMPORT THE MODULE TO ANOTHER PROGRAM.

INPUT (module: practical_14_module.py):

```
#This program is a module
#funtion to find fibonacci sequence of a given number
def fibonacci(num):
    if num <= 1:
        return num
    else:
        return fibonacci(num-1) + fibonacci(num-2)
```

INPUT (main):

```
import practical_14_module as f

num = 5
print(f"fibonacci sequece for {num} is: ")
for i in range(num):
    print(f.fibonacci(i), end=" ")
# 0 1 1 2 3 5 8 13 21 34
```

OUTPUT (main):

```
fibonacci sequece for 10 is:
0 1 1 2 3 5 8 13 21 34
```

TASK 15

AIM: WRITE A PYTHON PROGRAM TO DEFINE A MODULE AND IMPORT A SPECIFIC FUNCTION IN THAT MODULE TO ANOTHER PROGRAM.

INPUT (module: practical_15_module.py):

```
def find_max(list):
    return max(list)

def add(num_1, num_2):
    return num_1 + num_2
```

INPUT (main):

```
from practical_15_module import find_max

list = [23,56,23,76,45,54,65,76,45]
print(list)
print(f"maximum number for list is {find_max(list)}")
#Maximum number for list is 76
```

OUTPUT (main):

```
[23, 56, 23, 76, 45, 54, 65, 76, 45]
maximum number for list is 76
```

TASK 16

AIM: WRITE A SCRIPT NAMED COPYFILE.PY. THIS SCRIPT SHOULD PROMPT THE USER FOR THE NAMES OF TWO TEXT FILES. THE CONTENTS OF THE FIRST FILE SHOULD BE INPUT AND WRITTEN TO THE SECOND FILE.

INPUT:

```
#funtion to copy conent from a file to another file
def copycontent(r_file, w_file):
    with open(r_file, 'r') as file:
        content = file.read()

    with open(w_file, 'w') as file:
        file.write(content)
        print("Content succesfully copied.")


#main code starts here
r_file = input("Enter file name (with extention) from you want to copy: ")
w_file = input("Enter file name (with extention) where you want to write: ")
copycontent(r_file, w_file)  #calling function
```

OUTPUT:

```
Enter file name (with extention) from you want to copy: text.txt
Enter file name (with extention) where you want to write: copy.txt
Content succesfully copied.
```

Copy.txt

hello sir i am jarvis

text.txt

hello sir i am jarvis

TASK 17

AIM: WRITE A PROGRAM THAT INPUTS A TEXT FILE. THE PROGRAM SHOULD PRINT ALL OF THE UNIQUE WORDS IN THE FILE IN ALPHABETICAL ORDER.

INPUT:

```
#function to print sorted and unrepeated alphabets
def print_order(text_file):
    with open(text_file, 'r') as file:
        content = file.read()
        print("File content: \n", content)
        seen = set()        #creating a set named seen
        for char in content:
            if char.isalpha():
                seen.add(char)   #adding the unrepeated alphabets into seen

        sorted_seen = sorted(seen)    #sorting all alphabets

        print("\nOrdered alphabets: ")
        for char in sorted_seen:
            print(char, end=" ")

#main code starts here
print_order('text.txt')                          #a e h i j l m o r s v
```

OUTPUT:

text.txt

hello sir i am jarvis

```
File content:
 hello sir i am jarvis

Ordered alphabets:
a e h i j l m o r s v
```