**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'>

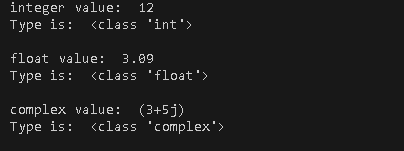
#complex

b = 3 + 5j

print("complex value: ", b)        #complex value: (3+5j)

print("Type is: ", type(b), "\n")  #Type is: <class 'complex'>

**OUTPUT:**



**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

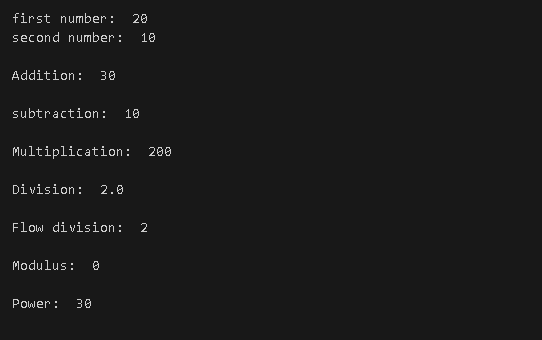
#Modulus

print("Modulus: " ,number\_1 % number\_2) #Modulus: 0

#Power

print("Power: " ,number\_1 ^ number\_2) #Power: 30

**OUTPUT:**



**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

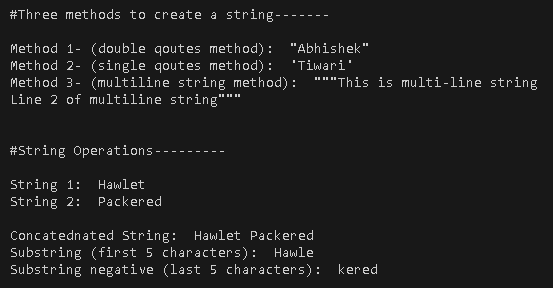
#using negative indexing

substring\_negative = concatenated[-5:]

print("Substring negative (last 5 characters): ", substring\_negative)

#Substring negative (last 5 characters): kered

**OUTPUT:**



**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:**



**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

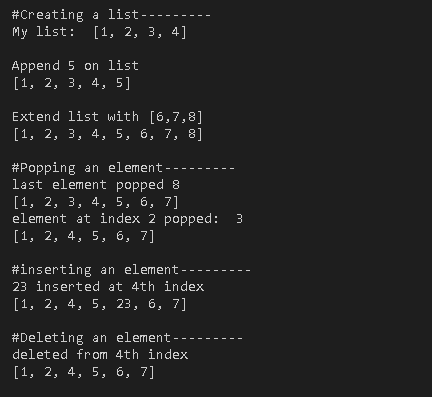
print("#Deleting an element---------")

del list[4]

print("deleted from 4th index ")

print(list) #[1, 2, 4, 5, 6, 7]

**OUTPUT:**



**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 ")

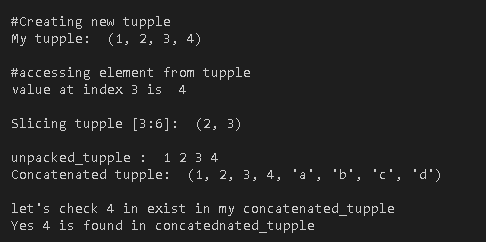
if (4 in concatenated\_tuple):

print("Yes 4 is found in concatenated\_tuple") #true condition

else:

print("No 4 is not found") #false condition

**OUTPUT:**



**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 ")

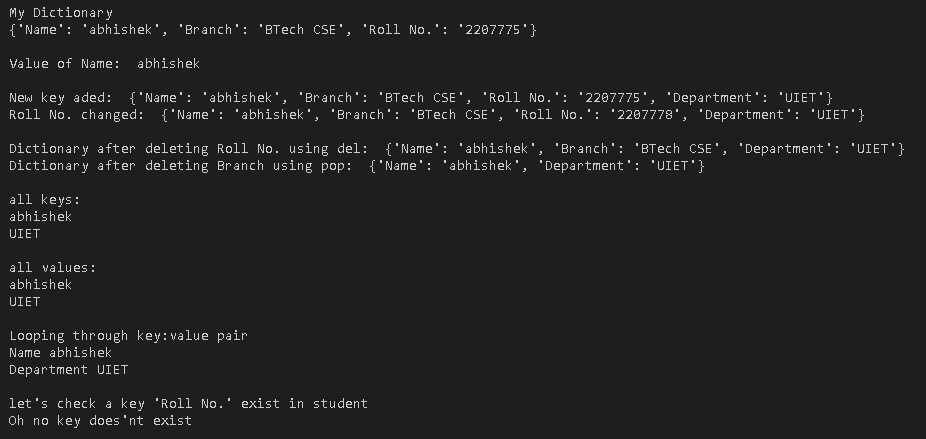
if ('Roll No.' in student[keys]):

print("Yeh! key found") #true condition

else:

print("Oh no key does'nt exist") #false condition

**OUTPUT:**



**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:**



**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

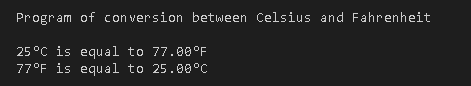
print(f"{celsius}°C is equal to {celsius\_to\_fahrenheit(celsius):.2f}°F")

#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:**



**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:**



**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:**



**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 P2 + B2 = H2

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:**



**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):**



**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

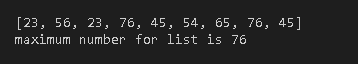
list = [23,56,23,76,45,54,65,76,45]

print(list)

print(f"maximum number for list is {find\_max(list)}")

#Maximum number for list is 76

**OUTPUT (main):**



**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.")

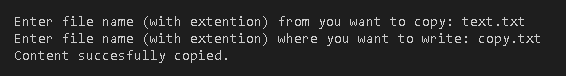
#main code starts here

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:**



**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

