Department Of Electrical & Electronics Engineering

GEETHANJALI INSTITUTE OF SCIENCE & TECHNOLOGY

GANGAVARAM (V), KOVUR (M), NELLORE-524137

II Year - II - Semester

Regulation:RG22

PYTHON PROGRAMMING

GEETHANJALI INSTITUTE OF SCIENCE & TECHNOLOGY

GANGAVARAM (V), KOVUR (M), NELLORE-524137

COURSE	
DEPARTMENT	
YEAR	
SEMESTER	
REGULATION	
SUBJECT	Python programming
COURSE CODE	
CREDITS	
ROLLNO OF STUDENT	
NAME OF THE STUDENT	
SECTION	

S.NO	PROGRAM	DATE	SIGNATURE OF FACULTY
Tasks:1	**OPERATORS**		
	a. Read a list of numbers and write a program to check whether a particular element is present or not using membership operators.		
	b. Read your name and age and write a program to display the year in which you will turn 100 years old.		
	c. Read radius and height of a cone and write a program to find the volume of a cone.		
	d. Write a program to compute distance between two points taking input from the user (Hint: use Pythagorean theorem)		
Tasks:2	**CONTROL STRUCTURES**		
	a. Read your email id and write a program to display the no of vowels, consonants, digits and white spaces in it using if...elif...else statement		
	(b). Write a Program to find the sum of a Series 1/1! + 2/2! + 3/3! + 4/4! +.....+ n/n!. (Input :n = 5, Output : 2.70833)		
	(c). In number theory, an abundant number or excessive number is a number for which the sum of its proper divisors is greater than the number itself. Write a program to find out, if the given number is abundant. (Input: 12, Sum of divisors of 12 = 1 + 2 + 3 + 4 + 6 = 16, sum of divisors 16 > original number 12)		
Tasks:3	**LIST**		
	(a). Read a list of numbers and print the numbers divisible by x but not by y (Assume x = 4 and y = 5).		
	(b). Read a list of numbers and print the sum of odd integers and even integers from the list.(Ex: [23, 10, 15, 14, 63], odd numbers sum = 101, even numbers sum = 24)		
	(c). Read a list of numbers and print numbers present in odd index position. (Ex: [10, 25, 30, 47, 56, 84, 96], The numbers in odd index position: 25 47 84).		

	(d). Read a list of numbers and remove the duplicate numbers from it. (Ex: Enter a list with duplicate elements: 10 20 40 10 50 30 20 10 80, The unique list is: [10, 20, 30, 40, 50, 80])		
Tasks:4	**TUPLE**		
	(a). Given a list of tuples. Write a program to find tuples which have all elements divisible by K from a list of tuples. test_list = [(6, 24, 12), (60, 12, 6), (12, 18, 21)], K = 6, Output : [(6, 24, 12), (60, 12, 6)]		
	(b). Given a list of tuples. Write a program to filter all uppercase characters tuples from given list of tuples. (Input: test_list = [("GFG", "IS", "BEST"), ("GFg", "AVERAGE"), ("GfG",), ("Gfg", "CS")], Output : [(„GFG", „IS", „BEST")]).		
	(c). Given a tuple and a list as input, write a program to count the occurrences of all items of the list in the tuple. (Input : tuple = ('a', 'a', 'c', 'b', 'd'), list = ['a', 'b'], Output : 3)		
Tasks:5	**SET**		
	(a).Write a program to generate and print a dictionary that contains a number (between 1 and n) in the form (x, x*x).		
	(b).Write a program to perform union, intersection and difference using Set A and Set B.		
	(c).Write a program to count number of vowels using sets in given string (Input : "Hello World", Output: No. of vowels : 3)		
	(d).Write a program to form concatenated string by taking uncommon characters from two strings using set concept (Input : S1 = "aacdb", S2 = "gafd", Output : "cbgf").		
Tasks:6	**DICTIONARY**		
	(a). Write a program to do the following operations: i. Create a empty dictionary with dict() method ii. Add elements one at a time iii. Update existing key"s value iv. Access an element using a key and also get() method v. Deleting a key value using del() method		
	(b).Write a program to create a dictionary and apply the following methods: i. pop() method		
	(c). Given a dictionary, write a program to find the sum of all items in the dictionary		
	(d).Write a program to merge two dictionaries using update() method		

Tasks:7	**STRINGS**		
	(a).Write a program to read a line of text and remove the initial word from given text. (Hint: Use split() method, Input : India is my country. Output : is my country)		
	(b). Write a program to read a string and count how many times each letter appears. (Histogram).		
Tasks:8	**USER DEFINED FUNCTIONS**		
	(a).A generator is a function that produces a sequence of results instead of a single value. Write a generator function for Fibonacci numbers up to n.		
	(b).Write a function merge_dict(dict1, dict2) to merge two Python dictionaries.		
	(c).Write a fact() function to compute the factorial of a given positive number		
	(d).Given a list of n elements, write a linear_search() function to search a given element x in a list.		
Tasks:9	**BUILT-IN FUNCTIONS**		
	(a).Write a program to demonstrate the working of built-in trignometric functions sin(), cos(), tan(), hypot(), degrees(), radians() by importing math module.		
	(b).Write a program to demonstrate the working of built-in Logarithmic and Power functions exp(), log(), log2(), log10(), pow() by importing math module.		
	(c).Write a program to demonstrate the working of built-in numeric functions ceil(), floor(), fabs(), factorial(), gcd() by importing math module.		
Tasks:10	**CLASS AND OBJECTS**		
	(a)Write a program to create a BankAccount class. Your class should support the following methods for i) Deposit ii) Withdraw iii) GetBalanace		
	(b).Write a program to create an employee class and store the employee name, id, age, and salary using the constructor. Display the employee details by invoking employee_info() method and also using dictionary (__dict__).		

	(c). Access modifiers in Python are used to modify the default scope of variables. Write a program to demonstrate the 3 types of access modifiers: public, private and protected.		
Tasks:11	**FILE HANDLING**		
	(a).Write a program to read a filename from the user, open the file (say firstFile.txt) and then perform the following operations: i. Count the sentences in the file. ii. Count the words in the file. iii. Count the characters in the file.		
	(b). Create a new file (Hello.txt) and copy the text to other file called target.txt. The target.txt file should store only lower case alphabets and display the number of lines copied		

Tasks: 1. OPERATORS

(a).Read a list of numbers and write a program to check whether a particular element is present or not using membership operators.

AIM: To develop a program to check whether a particular element is present or not using membership operators.

Source code:

```
x = 24
y = 20
list = [10, 20, 30, 40, 50]
if (x not in list):
   print("x is NOT present in given list")
else:
   print("x is present in given list")

if (y in list):
   print("y is present in given list")
else:
   print("y is NOT present in given list")
```

Input/Output:

x is NOT present in given list

y is present in given list

(b). Read your name and age and write a program to display the year in which you will turn 100 years old.

AIM: To develop a python program to display the year in which you will turn 100 years old.

Source code:

```
name = input("Enter your name: ") # user input
current_age = int(input("Enter your age: ")) # user input
#calculating the 100th year, considering 2020 as the current year
hundredth_year = 2020 + (100 - current_age)
print(f'{name} will become 100 years old in the year {hundredth_year}.')
```

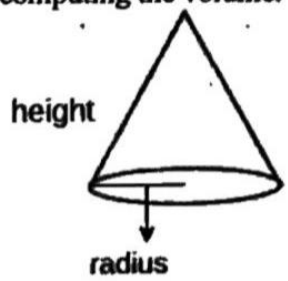
Input/Output:

```
Enter your name: venkat
Enter your age: 33
venkat will become 100 years old in the year 2087.
```

(c).Read radius and height of a cone and write a program to find the volume of a cone.

AIM: To develop a program to find the volume of a cone.

The cone can be defined as the three dimensional object with a circular base. The volume of the cone is measured in cubic units. Be sure that all of the measurements are in the same unit before computing the volume.



Formula

volume of cone= pie x r^2 x h/3

Algorithm

1. Define the height of the cone.
2. Define the radius of the cone.
3. Calculate the volume of the cone pie x r^2 x h/3
4. Define volume_cone and assign the volume of the cone to it.

Source code:

```
height=38
radius=35
pie=3.14285714286
volume=pie*(radius*radius)*height/3
print("volume of the cone="+str(volume))
```

Input/Output:

volume of the cone=48766.666666711004

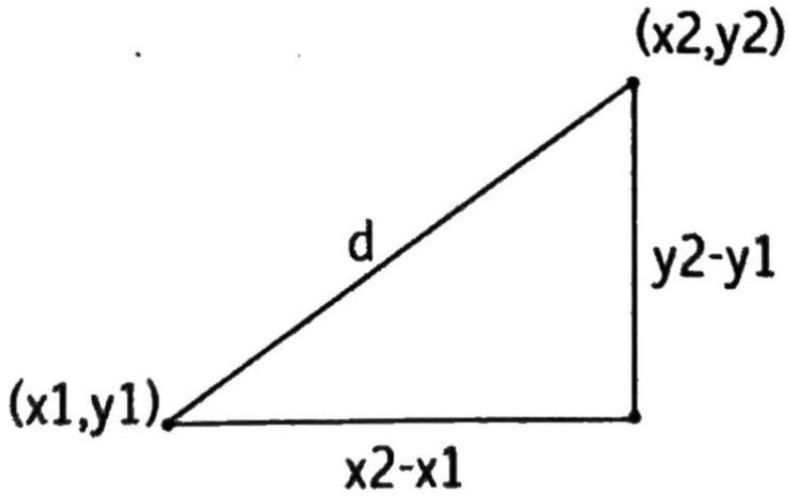
(d).Write a program to compute distance between two points taking input from the user (Hint: use Pythagorean theorem)

AIM: To develop a program to compute distance between two points taking input from the user (Hint: use Pythagorean theorem)

We will use the distance formula derived from Pythagorean theorem. The formula for distance between two point (x1, y1) and (x2, y2) is

Distance = $\sqrt{(x2-x1)^2+(y2-y1)^2}$

We can get above formula by simply applying Pythagoras theorem



Source code:

```
x1=int(input("enter x1 : "))
x2=int(input("enter x2 : "))
y1=int(input("enter y1 : "))
y2=int(input("enter y2 : "))
result= ((((x2 - x1 )**2) + ((y2-y1)**2) )**0.5)
print("distance between",(x1,x2),"and",(y1,y2),"is : ",result)
```

Input/Output:

enter x1 : 4

enter x2 : 6

enter y1 : 0

enter y2 : 6

distance between (4, 6) and (0, 6) is : 6.324555320336759

2. CONTROL STRUCTURES:

(a). Read your email id and write a program to display the no of vowels, consonants, digits and white spaces in it using if...elif...else statement.

AIM: To develop a program to display the no of vowels, consonants, digits and white spaces in it using if...elif...else statement.

Source code:

```
str = input("Enter the string : ")

vowels = 0

digits = 0

consonants = 0

spaces = 0

symbols = 0

str = str.lower()

for i in range(0, len(str)):
```

```
 if(str[i] == 'a'or str[i] == 'e' or str[i] == 'i' or str[i] == 'o' or str[i] == 'u'):
     vowels = vowels + 1
 elif((str[i] >= 'a'and str[i] <= 'z')):
     consonants = consonants + 1
 elif( str[i] >= '0' and str[i] <= '9'):
     digits = digits + 1
 elif (str[i] == ' '):
      spaces = spaces + 1
 else:
      symbols = symbols + 1
print("Vowels: ", vowels);
print("Consonants:", consonants);
print("Digits:", digits);
print("White spaces:", spaces);
print("Symbols :", symbols);
```

Input/Output:

Enter the string : venkateswarlu.venkey9@gmail.com

Vowels: 10

Consonants: 17

Digits: 1

White spaces: 0

Symbols : 3

(b). Write a Program to find the sum of a Series 1/1! + 2/2! + 3/3! + 4/4! +.....+ n/n!. (Input :n = 5, Output : 2.70833)

AIM: To develop a program to find the sum of a Series 1/1! + 2/2! + 3/3! + 4/4! +.....+ n/n!. (Input :n = 5, Output : 2.70833)

Source code:

```
def sumOfSeries(num):
    res = 0
    fact = 1
    for i in range(1, num+1):
        fact *= i
        res = res + (i/ fact)
  return res
n = 100
print("Sum: ", sumOfSeries(n))
```

Input/Output:

Sum: 2.7182818284590455

(c). In number theory, an abundant number or excessive number is a number for which the sum of its proper divisors is greater than the number itself. Write a program to find out, if the given number is abundant. (Input: 12, Sum of divisors of 12 = 1 + 2 + 3 + 4 + 6 = 16, sum of divisors 16 > original number 12)

AIM: To develop a python program to find out, if the given number is abundant. (Input: 12, Sum of divisors of 12 = 1 + 2 + 3 + 4 + 6 = 16, sum of divisors 16 > original number 12)

Source code:

```
def is_abundant(n):
    fctr_sum = sum([fctr for fctr in range(1, n) if n % fctr == 0])
    return fctr_sum > n
a = int(input("Enter the number?"));
if is_abundant(a):
    print("The number is Abundant.");
else:
    print("The number is not Abundant.")
```

Input/Output:

Enter the number?12

The number is Abundant.

3: LIST

(a). Read a list of numbers and print the numbers divisible by x but not by y (Assume x = 4 and y = 5).

AIM: To develop a python program to Read a list of numbers and print the numbers divisible by x but not by y (Assume x = 4 and y = 5).

Source code:

```
start = int(input("Enter start number:"))

end = int(input("Enter last number:"))

for i in range(start, end+1):

  if(i%4==0):

    print(i)
```

Input/Output:

```
>>>
========== RESTART: C:/Users/Dell Laptop/OneDrive/Desktop/pychar/five.py ==========
Enter start number:1
Enter last number:30
1
9
12
16
23
25
28
32
36
47
44
48
```

(b). Read a list of numbers and print the sum of odd integers and even integers from the list.(Ex: [23, 10, 15, 14, 63], odd numbers sum = 101, even numbers sum = 24)

AIM: To develop a python program to Read a list of numbers and print the sum of odd integers and even integers from the list.(Ex: [23, 10, 15, 14, 63], odd numbers sum = 101, even numbers sum = 24)

Source code:

```
NumList = [ ]

Even_Sum = 0

Odd_Sum = 0

Number = int(input("Please enter the Total Number of List Elements: "))

for i in range(1, Number + 1):

   value = int(input("Please enter the Value of %d Element : " %i))

   NumList.append(value)

for j in range(Number):

   if(NumList[j] % 2 == 0):

      Even_Sum = Even_Sum + NumList[j]
```

```
    else:
        Odd_Sum = Odd_Sum + NumList[j]

print("\nThe Sum of Even Numbers in this List = ", Even_Sum)
print("The Sum of Odd Numbers in this List = ", Odd_Sum)
```

Input/Output:

```
Please enter the Total Number of List Elements: 5
Please enter the Value of 1 Element : 23
Please enter the Value of 2 Element : 10
Please enter the Value of 3 Element : 15
Please enter the Value of 4 Element : 14
Please enter the Value of 5 Element : 63

The Sum of Even Numbers in this List =  24
The Sum of Odd Numbers in this List =  101
```

(c). Read a list of numbers and print numbers present in odd index position. (Ex: [10, 25, 30, 47, 56, 84, 96], The numbers in odd index position: 25 47 84).

AIM: To develop a python program to Read a list of numbers and print numbers present in odd index position. (Ex: [10, 25, 30, 47, 56, 84, 96], The numbers in odd index position: 25 47 84).

Source code:

```
my_list = [10,25,30,47,56,84,96]
print("The list is :")
print(my_list)
print("The elements in odd positions are : ")
for i in range(1, len(my_list), 2):
  print(my_list[i])
```

Input/Output:

```
>>>
========== RESTART: C:/Users/Dell Laptop/OneDrive/Desktop/pycises/oddnos1ex.py ==========
The list is :
[13, 25, 36, 47, 58, 34, 96]
The elements in odd positions are :
25
47
34
>>>
```

(d). Read a list of numbers and remove the duplicate numbers from it. (Ex: Enter a list with duplicate elements: 10 20 40 10 50 30 20 10 80, The unique list is: [10, 20, 30, 40, 50, 80])

AIM: To develop a python program to Read a list of numbers and remove the duplicate numbers from it. (Ex: Enter a list with duplicate elements: 10 20 40 10 50 30 20 10 80, The unique list is: [10, 20, 30, 40, 50, 80])

Source code:

```
list_value1=[10,20,40,10,50,30,20,10,80]
print("The initialized list is ",list_value1)
res_list=[]
for i in list_value1:
   if i not in res_list:
      res_list.append(i)
#printing the list after removing duplicate elements
print("The resultant list after removing duplicates is ",res_list)
```

Input/Output:

```
>>>
========== RESTART: C:/Users/Dell Laptop/OneDrive/Desktop/pyt2sem/duplicate.py ==========
The initialized list is  [10, 20, 40, 10, 50, 30, 20, 10, 80]
The resultant list after removing duplicates is  [10, 20, 40, 50, 30, 80]
>>>
```

4: TUPLE

(a). Given a list of tuples. Write a program to find tuples which have all elements divisible by K from a list of tuples. test_list = [(6, 24, 12), (60, 12, 6), (12, 18, 21)], K = 6, Output : [(6, 24, 12), (60, 12, 6)]

AIM: To develop a python program to find tuples which have all elements divisible by K from a list of tuples. test_list = [(6, 24, 12), (60, 12, 6), (12, 18, 21)], K = 6, Output : [(6, 24, 12), (60, 12, 6)]

Source code:

```
my_list = [(6, 24, 12), (60, 12, 6), (12, 18, 21)]

print("The list is : ")
print(my_list)

K = 6
print("The value of K has been initialized to ")
print(K)
my_result = [sub for sub in my_list if all(ele % K == 0 for ele in sub)]

print("Elements divisible by K are : " + str(my_result))
```

Input/Output:

```
== RESTART: C:/Users/Dell Laptop/OneDrive/Desktop/python/divisiblebyk.py ==
The list is :
[(6, 24, 12), (60, 12, 6), (12, 18, 21)]
The value of K has been initialized to
6
Elements divisible by K are : [(6, 24, 12), (60, 12, 6)]
>>>
```

(b). Given a list of tuples. Write a program to filter all uppercase characters tuples from given list of tuples. (Input: test_list = [("GFG", "IS", "BEST"), ("GFg", "AVERAGE"), ("GfG",), ("Gfg", "CS")], Output : [(„GFG", „IS", „BEST")]).

AIM: To develop a python program to filter all uppercase characters tuples from given list of tuples. (Input: test_list = [("GFG", "IS", "BEST"), ("GFg", "AVERAGE"), ("GfG",), ("Gfg", "CS")], Output : [(„GFG", „IS", „BEST")]).

Source code:

```
test_list = [("GFG", "IS", "BEST"), ("GFg", "AVERAGE"), ("GFG", ), ("Gfg",
"CS")]

# printing original list
print("The original list is : " + str(test_list))

res_list = []
for sub in test_list:
    res = True
    for ele in sub:

        # checking for uppercase
        if not ele.isupper():
            res = False
            break
    if res:
        res_list.append(sub)
```

```
print("Filtered Tuples : " + str(res_list))
```

Input/Output:

```
///
   ========== RESTART: C:/Users/Dell Laptop/OneDrive/Desktop/pythonn/tuplecapvowel.py ==========
   The original list is : [('Gfg', 'is', 'best'), ('Gfg', 'for', 'all'), ('Gfg', 'CS')]
   Filtered Tuples : [('Gfg', 'is', 'best'), ('Gfg',)]
>>>
```

(c). Given a tuple and a list as input, write a program to count the occurrences of all items of the list in the tuple. (Input : tuple = ('a', 'a', 'c', 'b', 'd'), list = ['a', 'b'], Output : 3)

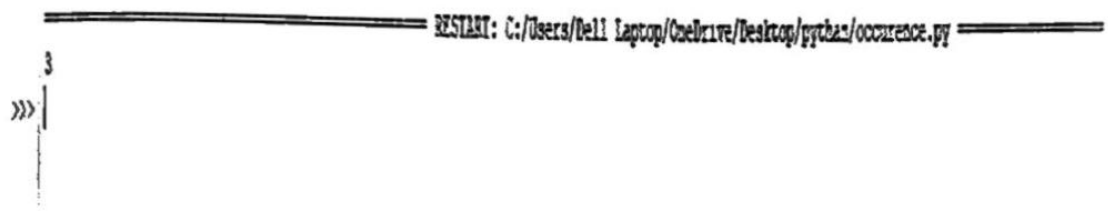
AIM: To develop a python program to count the occurrences of all items of the list in the tuple. (Input : tuple = ('a', 'a', 'c', 'b', 'd'), list = ['a', 'b'], Output : 3)

Source code:

```
def countOccurrence(tup, lst):
    dct = {}
    for i in tup:
        if not dct.get(i):
            dct[i] = 0
        dct[i] += 1
    return sum(dct.get(i, 0) for i in lst)

# Driver Code
tup = ('a', 'a', 'c', 'b', 'd')
lst = ['a', 'b']
print(countOccurrence(tup, lst))
```

Input/Output:



5: SET

(a).Write a program to generate and print a dictionary that contains a number (between 1 and n) in the form (x, x*x).

AIM: To develop a python program to generate and print a dictionary that contains a number (between 1 and n) in the form (x, x*x).

Source code:

```
n=int(input("Input a number "))
d = dict()

for x in range(1,n+1):
  d[x]=x*x

      print(d)
```

Input/Output:

Input a number 10

{1: 1, 2: 4, 3: 9, 4: 16, 5: 25, 6: 36, 7: 49, 8: 64, 9: 81, 10: 100}

(b). Write a program to perform union, intersection and difference using Set A and Set B.

AIM: To develop a python program to perform union, intersection and difference using Set A and Set B.

Source code:

```
# sets are define
A = {0, 2, 4, 6, 8};
B = {1, 2, 3, 4, 5};

# union
print("Union :", A | B)

# intersection
print("Intersection :", A & B)

# difference
print("Difference :", A - B)

# symmetric difference
print("Symmetric difference :", A ^ B)
```

Input/Output:

```
Union : {0, 1, 2, 3, 4, 5, 6, 8}
Intersection : {2, 4}
Difference : {0, 8, 6}
Symmetric difference : {0, 1, 3, 5, 6, 8}
```

(c).Write a program to count number of vowels using sets in given string (Input : "Hello World", Output: No. of vowels : 3)

AIM: To develop a python program to count number of vowels using sets in given string (Input : "Hello World", Output: No. of vowels : 3)

Source code:

```
def vowel_count(str):

    # Initializing count variable to 0
    count = 0

    # Creating a set of vowels
```

```
    vowel = set("aeiouAEIOU")

    # Loop to traverse the alphabet
    # in the given string
    for alphabet in str:

        # If alphabet is present
        # in set vowel
        if alphabet in vowel:
            count = count + 1

    print("No. of vowels :", count)

# Driver code
str = "hello world"

# Function Call
vowel_count(str)
```

Input/Output:

No. of vowels : 3

```
>>>
========== RESTART: C:/Users/Dell Laptop/AppData/Local/Programs/Python/Python310/setc.py ==========
No. of vowels : 3
>>>
```

(d). Write a program to form concatenated string by taking uncommon characters from two strings using set concept (Input : S1 = "aacdb", S2 = "gafd", Output : "cbgf").

AIM: To develop a python program to form concatenated string by taking uncommon characters from two strings using set concept (Input : S1 = "aacdb", S2 = "gafd", Output : "cbgf").

Source code:

```
def uncommonConcat(str1, str2):

    # convert both strings into set
    set1 = set(str1)
    set2 = set(str2)

    # take intersection of two sets to get list of
    # common characters
    common = list(set1 & set2)

    # separate out characters in each string
    # which are not common in both strings
    result = [ch for ch in str1 if ch not in common] + [ch for ch in str2 if ch not in
common]

    # join each character without space to get
    # final string
    print( ''.join(result) )

# Driver program
if __name__ == "__main__":
    str1 = 'aacdb'
    str2 = 'gafd'
    uncommonConcat(str1,str2)
```

Input/Output:

cbgf

```
>>>
= RESTART: C:/Users/Dell Laptop/AppData/Local/Programs/Python/Python310/setd.py
cbgf
>>>
```

..

6: DICTIONARY:

(a). Write a program to do the following operations:
i. Create a empty dictionary with dict() method
ii. Add elements one at a time
iii. Update existing key''s value
iv. Access an element using a key and also get() method
v. Deleting a key value using del() method

AIM: To develop a python program to do the following operations:
i. Create a empty dictionary with dict() method
ii. Add elements one at a time
iii. Update existing key''s value
iv. Access an element using a key and also get() method
v. Deleting a key value using del() method

i. Create a empty dictionary with dict() method

Source code:

```
# Creating an empty Dictionary
Dict = {}
print("Empty Dictionary: ")
print(Dict)
```

Output:

```
Created empty dictionary: {}
<class 'dict'>
```

ii. Add elements one at a time

Source code:

```
# Adding elements one at a time
Dict[0] = 'Geeks'
Dict[2] = 'For'
Dict[3] = 1
print("\nDictionary after adding 3 elements: ")
print(Dict)
```

Output:

Dictionary after adding 3 elements:
{0: 'Geeks', 2: 'For', 3: 1}

iii. Update existing key"s value

Source code:

```
# Updating existing Key's Value
Dict[2] = 'Welcome'
print("\nUpdated key value: ")
print(Dict)
```

Output:

Updated key value:
{0: 'Geeks', 2: 'Welcome', 3: 1, 'Value_set': (2, 3, 4)}

iv. Access an element using a key and also get() method

Source code:

```
# Creating a Dictionary
Dict = {1: 'Geeks', 'name': 'For', 3: 'Geeks'}

# accessing a element using key
print("Accessing a element using key:")
print(Dict['name'])

# accessing a element using key
print("Accessing a element using key:")
print(Dict[1])
```

Output:

RESTART: C:/Users/Dell Laptop/AppData/Local/Programs/Python/Python310/tina.py

Accessing a element using key:

For

Accessing a element using key:

Geeks

v. Deleting a key value using del() method

Source code:

```
# Initial Dictionary
Dict = { 5 : 'Welcome', 6 : 'To', 7 : 'Geeks',
     'A' : {1 : 'Geeks', 2 : 'For', 3 : 'Geeks'},
     'B' : {1 : 'Geeks', 2 : 'Life'}}
print("Initial Dictionary: ")
print(Dict)

# Deleting a Key value
del Dict[6]
print("\nDeleting a specific key: ")
print(Dict)
```

Output:

Initial Dictionary:
{5: 'Welcome', 6: 'To', 7: 'Geeks', 'A': {1: 'Geeks', 2: 'For', 3: 'Geeks'}, 'B': {1: 'Geeks', 2: 'Life'}}

Deleting a specific key:
{5: 'Welcome', 7: 'Geeks', 'A': {1: 'Geeks', 2: 'For', 3: 'Geeks'}, 'B': {1: 'Geeks', 2: 'Life'}}

(b). Write a program to create a dictionary and apply the following methods:
i. pop() method

AIM: To develop a python program to create a dictionary and apply the following methods:
i. pop() method
ii. popitem() method
iii. clear() method

(i). pop() method :

Source code:

```
# working of pop()

# initializing dictionary
test_dict = {"Nikhil": 7, "Akshat": 1, "Akash": 2}

# Printing initial dict
print("The dictionary before deletion : " + str(test_dict))

# using pop to return and remove key-value pair.
pop_ele = test_dict.pop('Akash')

# Printing the value associated to popped key
print("Value associated to poppped key is : " + str(pop_ele))

# Printing dictionary after deletion
print("Dictionary after deletion is : " + str(test_dict))
```

Input/Output:

```
The dictionary before deletion : {'Nikhil': 7, 'Akshat': 1, 'Akash': 2}
Value associated to poppped key is : 2
Dictionary after deletion is : {'Nikhil': 7, 'Akshat': 1}
```

(c). Given a dictionary, write a program to find the sum of all items in the dictionary.

AIM: To develop a python program to find the sum of all items in the dictionary.

Source code:

```
def Sum(dic):
  #sum variable
  sum=0
  #iterate through values
  for i in dic.values():
```

```
    sum=sum+i
  return sum

#initialisation
dic={ 'x':30, 'y':145, 'z':55 }

print("Dictionary: ", dic)

#print sum
print("sum: ",Sum(dic))
```

Input/Output:

```
======== RESTART: C:/Users/Dell Laptop/OneDrive/Desktop/python/pop.py ========
Dictionary:  {'x': 30, 'y': 145, 'z': 55}
sum:  230
>>>
```

(d). Write a program to merge two dictionaries using update() method

AIM: To develop a python program to merge two dictionaries using update() method

Source code

```
def Merge(dict1, dict2):
  res = dict1 | dict2
  return res

# Driver code
dict1 = {'x': 10, 'y': 8}
dict2 = {'a': 6, 'b': 4}
dict3 = Merge(dict1, dict2)
print(dict3)
```

Input/Output:

```
========== RESTART: C:/Users/Dell Laptop/OneDrive/Desktop/pyt2.py ==========
{'x': 10, 'y': 8, 'a': 6, 'b': 4}
>>>
```

7: STRINGS

(a).Write a program to read a line of text and remove the initial word from given text. (Hint: Use split() method, Input : India is my country. Output : is my country)

AIM: To develop a python program to read a line of text and remove the initial word from given text. (Hint: Use split() method, Input : India is my country. Output : is my country)

Source code

```
# Using split()

# initializing string
test_str = " India is my country"

# printing original string
print("The original string is : " + test_str)

# Using split()
# Removing Initial word from string
res = test_str.split(' ', 2)[2]

# printing result
print("The string after omitting first word is : " + str(res))
```

Input/Output:

```
========== RESTART: C:/Users/Dell Laptop/OneDrive/Desktop/python/p
The original string is : India is my country
The string after omitting first word is : is my country
>>>
```

(b). Write a program to read a string and count how many times each letter appears. (Histogram).

AIM: To develop a python program to read a string and count how many times each letter appears. (Histogram).

Source code

```
string=input("enter any string:")
c=input("enter character to check frequence:")
count=0
for i in string:
   if i==c:
      count+=1
print(c,"occurs",count,"time(s).")
```

Input/Output:

```
>>>
= RESTART: C:/Users/Dell Laptop/AppData/Local/Programs/Python/Python310/stringb.py
enter any string: hello world
enter character to check frequence:o
o occurs 2 time(s).
>>>
```

8: USER DEFINED FUNCTIONS

(a). A generator is a function that produces a sequence of results instead of a single value. Write a generator function for Fibonacci numbers up to n.

AIM: To develop a python program to generator function for Fibonacci numbers up to n.

Source code

```
def fib(num):
  a = 0
  b = 1
  for i in range(num):
    yield a
    a, b = b, a + b # Adds values together then swaps them

for x in fib(10):
  print(x)
```

Input/Output:

```
>>>
========== RESTART: C:/Users/Dell Laptop/AppData/Local/Programs/Python/Python310/
0
1
1
2
3
5
8
13
21
34
>>>
```

(b). Write a function merge_dict(dict1, dict2) to merge two Python dictionaries.

AIM: To develop a python program to merge_dict(dict1, dict2) to merge two Python dictionaries.

Source code

```
def Merge(dict1, dict2):
  res = {**dict1, **dict2}
  return res

# Driver code
dict1 = {'a': 10, 'b': 8}
dict2 = {'d': 6, 'c': 4}
dict3 = Merge(dict1, dict2)
print(dict3)
```

Input/Output:

```
///
========== RESTART: C:/Users/Dell Laptop/OneDrive/Desktop/pythan/functionc.py
{'a': 10, 'b': 8, 'd': 6, 'c': 4}
>>>
```

(c). Write a fact() function to compute the factorial of a given positive number

AIM: To develop a python program a fact() function to compute the factorial of a given positive number.

Source code

```
num = int(input("Enter a number: "))
factorial = 1
if num < 0:
  print(" Factorial does not exist for negative numbers")
elif num == 0:
  print("The factorial of 0 is 1")
else:
  for i in range(1,num + 1):
    factorial = factorial*i
  print("The factorial of",num,"is",factorial)
```

Input/Output:

```
========== RESTART: C:/Users/Dell Laptop/OneDrive/Desktop/pythan/f
Enter a number: 4
The factorial of 4 is 24
```

(d). Given a list of n elements, write a linear_search() function to search a given element x in a list.

AIM: To develop a python program a linear_search() function to search a given element x in a list.

Source code

```
def linear_Search(list1, n, key):
    # Searching list1 sequentially
    for i in range(0, n):
        if (list1[i] == key):
            return i
    return -1
list1 = [1 ,3, 5, 4, 7, 9]
key = 3
n = len(list1)
res = linear_Search(list1, n, key)
if(res == -1):
    print("Element not found")
else:
    print("Element found at index: ", res)
```

Input/Output:

```
>>>
======== RESTART: C:/Users/Dell Laptop/OneDrive/Desktop/pythan/liener.py ========
Element found at index:  1
>>>
```

9: BUILT-IN FUNCTIONS:

(a). Write a program to demonstrate the working of built-in trignometric functions sin(), cos(), tan(), hypot(), degrees(), radians() by importing math module.

AIM: To develop a python program to demonstrate the working of built-in trignometric functions sin(), cos(), tan(), hypot(), degrees(), radians() by importing math module.

Source code:

```
import math
a = math.pi/6
# returning the value of sine of pi/6
print ("The value of sine of pi/6 is : ", end="")
print (math.sin(a))

# returning the value of cosine of pi/6
print ("The value of cosine of pi/6 is : ", end="")
print (math.cos(a))

# Python code to demonstrate the working of
# tan() and hypot()
# importing "math" for mathematical operations
import math
a = math.pi/6
b = 3
c = 4
# returning the value of tangent of pi/6
print ("The value of tangent of pi/6 is : ", end="")
print (math.tan(a))
```

```
# returning the value of hypotenuse of 3 and 4
print ("The value of hypotenuse of 3 and 4 is : ", end="")
print (math.hypot(b,c))


# Python code to demonstrate the working of
# degrees() and radians()
# importing "math" for mathematical operations
import math
a = math.pi/6
b = 30
# returning the converted value from radians to degrees
print ("The converted value from radians to degrees is : ", end="")
print (math.degrees(a))
# returning the converted value from degrees to radians
print ("The converted value from degrees to radians is : ", end="")
print (math.radians(b))
```

Input/Output:

```
========== RESTART: C:/Users/Dell Laptop/OneDrive/Desktop/python/liener.py ==========
The value of sine of pi/6 is : 0.49999999999999994
The value of cosine of pi/6 is : 0.8660254037844387
>>>
========== RESTART: C:/Users/Dell Laptop/OneDrive/Desktop/python/liener.py ==========
The value of tangent of pi/6 is : 0.5773502691896257
The value of hypotenuse of 3 and 4 is : 5.0
>>>
========== RESTART: C:/Users/Dell Laptop/OneDrive/Desktop/python/liener.py ==========
The converted value from radians to degrees is : 29.999999999999996
The converted value from degrees to radians is : 0.5235987755982988
>>>
```

(b).Write a program to demonstrate the working of built-in Logarithmic and Power functions exp(), log(), log2(), log10(), pow() by importing math module.

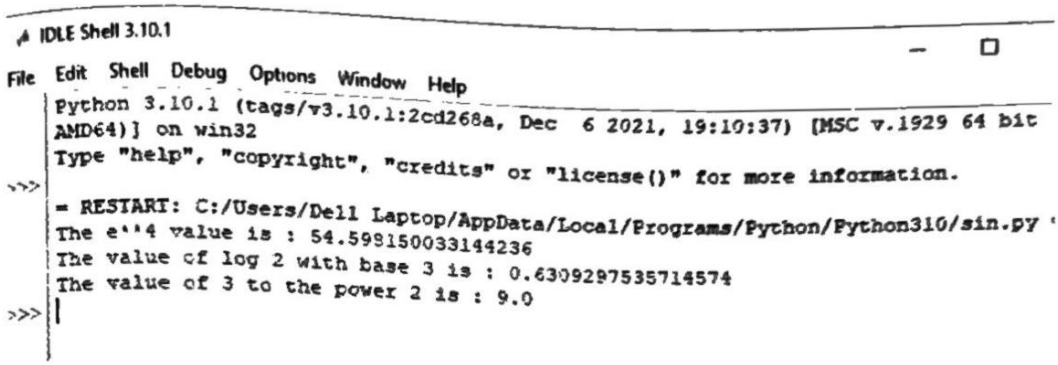
AIM: To develop a python program to demonstrate the working of built-in Logarithmic and Power functions exp(), log(), log2(), log10(), pow() by importing math module.

Source code:

```
# Python code to demonstrate the working of
# exp() and log()
# importing "math" for mathematical operations
import math
# returning the exp of 4
print ("The e**4 value is : ", end="")
print (math.exp(4))
# returning the log of 2,3
print ("The value of log 2 with base 3 is : ", end="")
print (math.log(2,3))

# Python code to demonstrate the working of
# pow()
# importing "math" for mathematical operations
import math
# returning the value of 3**2
print ("The value of 3 to the power 2 is : ", end="")
print (math.pow(3,2))
```

Input/Output:



(c). Write a program to demonstrate the working of built-in numeric functions ceil(), floor(), fabs(), factorial(), gcd() by importing math module.

AIM: To develop a python program to demonstrate the working of built-in numeric functions ceil(), floor(), fabs(), factorial(), gcd() by importing math module

Source code

```
# Python code to demonstrate the working of
# ceil() and floor()
# importing "math" for mathematical operations
import math
a = 2.3
# returning the ceil of 2.3
print ("The ceil of 2.3 is : ", end="")
print (math.ceil(a))
# returning the floor of 2.3
print ("The floor of 2.3 is : ", end="")
print (math.floor(a))
```

.....

```
# Python code to demonstrate the working of
# fabs() and factorial()
# importing "math" for mathematical operations
```

```
import math
a = -10
b= 5
# returning the absolute value.
print ("The absolute value of -10 is : ", end="")
print (math.fabs(a))
# returning the factorial of 5
print ("The factorial of 5 is : ", end="")
print (math.factorial(b))
```

.....

```
# Python code to demonstrate the working of
#gcd()
# importing "math" for mathematical operations
import math
a = -10
b = 5.5
c = 15
d = 5
# returning the gcd of 15 and 5
print ("The gcd of 5 and 15 is : ", end="")
print (math.gcd(5,15))
```

Input/Output:

```
>>>
========== RESTART: C:/Users/Dell Laptop/OneDrive/Desktop/python/liener.py ==========
The ceil of 2.3 is : 3
The floor of 2.3 is : 2
>>>
========== RESTART: C:/Users/Dell Laptop/OneDrive/Desktop/python/liener.py ==========
The absolute value of -10 is : 10.0
The factorial of 5 is : 120
>>>
========== RESTART: C:/Users/Dell Laptop/OneDrive/Desktop/python/liener.py ==========
The gcd of 5 and 15 is : 5
```

10. CLASS AND OBJECTS

(a) .Write a program to create a BankAccount class. Your class should support the following methods for

i) Deposit

ii) Withdraw

iii) GetBalanace

AIM: To develop a python program to create a BankAccount class. Your class should support the following methods for

i) Deposit

ii) Withdraw

iii) GetBalanace

Source code:

```
# Python program to create Bankaccount class
# with both a deposit() and a withdraw() function
class Bank_Account:
    def __init__(self):
```

```
    self.balance=0
    print("Hello!!! Welcome to the Deposit & Withdrawal Machine")

  def deposit(self):
    amount=float(input("Enter amount to be Deposited: "))
    self.balance += amount
    print("\n Amount Deposited:",amount)
.....
  def withdraw(self):
    amount = float(input("Enter amount to be Withdrawn: "))
    if self.balance>=amount:
      self.balance-=amount
      print("\n You Withdrew:", amount)
    else:
      print("\n Insufficient balance  ")

  def display(self):
    print("\n Net Available Balance=",self.balance)

# Driver code
# creating an object of class
s = Bank_Account()
# Calling functions with that class object
s.deposit()
s.withdraw()
s.display()
```

Input/Output:

```
========== RESTART: C:/Users/Dell Laptop/OneDrive/Desktop/pythan/oopj.py ==========
Hello!!! Welcome to the Deposit & Withdrawal Machine
Enter amount to be Deposited: 1000

 Amount Deposited: 1000.0
Enter amount to be Withdrawn: 500

 You Withdrew: 500.0

 Net Available Balance= 500.0
>>
```

(b). Write a program to create an employee class and store the employee name, id, age, and salary using the constructor. Display the employee details by invoking employee_info() method and also using dictionary (__dict__).

AIM: To develop a python program to_create an employee class and store the employee name, id, age, and salary using the constructor. Display the employee details by invoking employee_info() method and also using dictionary (__dict__).

Source code:

```
class Employee:
    count=0
    def __init__(self, name, desig, salary):
        self.name=name
        self.desig=desig
        self.salary=salary
        Employee.count+=1
    def displayCount(self):
        print("There are %d employees" % Employee.count)
    def displayDetails(self):
        print("Name:", self.name, ", Designation:", self.desig, ", Salary:",
self.salary)
```

```
e1=Employee("John", "Manager", 80000)
e2=Employee("Mike", "Team Leader", 50000)
e3=Employee("Derek", "Programmer", 30000)
e4=Employee("Raj", "Assistant", 25000)
e4.displayCount()
print("Details of all employee:")
e1.displayDetails()
e2.displayDetails()
e3.displayDetails()
e4.displayDetails()
```

Input/Output:

```
========== RESTART: C:/Users/Dell Laptop/OneDrive/Desktop/pyt
There are 4 employees
Details of all employee:
Name: John , Designation: Manager , Salary: 80000
Name: Mike , Designation: Team Leader , Salary: 50000
Name: Derek , Designation: Programmer , Salary: 30000
Name: Raj , Designation: Assistant , Salary: 25000
>>>
```

(c). Access modifiers in Python are used to modify the default scope of variables. Write a program to demonstrate the 3 types of access modifiers: public, private and protected.

AIM: To develop a python program to demonstrate the 3 types of access modifiers: public, private and protected.

Source code:

```
#program to illustrate public access modifier in a class


class Geek:
```

```
    # constructor
    def __init__(self, name, age):
        # public data members
        self.geekName = name
        self.geekAge = age
    # public member function
    def displayAge(self):
        # accessing public data member
        print("Age: ", self.geekAge)
# creating object of the class
obj = Geek("R2J", 20)
# accessing public data member
print("Name: ", obj.geekName)
# calling public member function of the class
obj.displayAge()
```

Output:

Name: R2J

Age: 20

```
# program to illustrate protected access modifier in a class

# super class
class Student:

    # protected data members
    _name = None
```

```
    _roll = None
    _branch = None

    # constructor
    def __init__(self, name, roll, branch):
        self._name = name
        self._roll = roll
        self._branch = branch

    # protected member function
    def _displayRollAndBranch(self):

        # accessing protected data members
        print("Roll: ", self._roll)
        print("Branch: ", self._branch)

# derived class
class Geek(Student):

    # constructor
    def __init__(self, name, roll, branch):
            Student.__init__(self, name, roll, branch)

    # public member function
    def displayDetails(self):

            # accessing protected data members of super class
```

```
        print("Name: ", self._name)

        # accessing protected member functions of super class
        self._displayRollAndBranch()

# creating objects of the derived class
obj = Geek("R2J", 1706256, "Information Technology")

# calling public member functions of the class
obj.displayDetails()

# program to illustrate private access modifier in a class

class Geek:

    # private members
    __name = None
    __roll = None
    __branch = None

    # constructor
    def __init__(self, name, roll, branch):
        self.__name = name
        self.__roll = roll
        self.__branch = branch

    # private member function
```

```
def __displayDetails(self):

    # accessing private data members
    print("Name: ", self.__name)
    print("Roll: ", self.__roll)
    print("Branch: ", self.__branch)

  # public member function
  def accessPrivateFunction(self):
      # accessing private member function
      self.__displayDetails()
# creating object
obj = Geek("R2J", 1706256, "Information Technology")

# calling public member function of the class
obj.accessPrivateFunction()
```

Input/Output:

```
========== RESTART: C:/Users/Dell Laptop/OneDrive/Desktop/python/emp.py ==========
Name:  R2J
Age:  20
>>>
========== RESTART: C:/Users/Dell Laptop/OneDrive/Desktop/python/emp.py ==========
Name:  R2J
Roll:  1706256
Branch:  Information Technology
>>>
========== RESTART: C:/Users/Dell Laptop/OneDrive/Desktop/python/emp.py ==========
Name:  R2J
Roll:  1706256
Branch:  Information Technology
>>>
```

11. FILE HANDLING:

(a). Write a program to read a filename from the user, open the file (say firstFile.txt) and then perform the following operations:

i. Count the sentences in the file.

ii. Count the words in the file.

iii. Count the characters in the file.

AIM: To develop a python program to read a filename from the user, open the file (say firstFile.txt) and then perform the following operations:

i. Count the sentences in the file.

ii. Count the words in the file.

iii. Count the characters in the file

Source code:

```
#counting number of lines, words and characters in a file
import os, sys
#open the file for reading data
fname=input("enter file name:")
if os.path.isfile(fname):
   f=open(fname,'r')
else:
   print(fname+'doed not exit')
   sys.exit()
#initialize the counters to 0
   c1=cw=cc=0
#read line by line from the file
   for line in f:
      words=line.split()
      c1+=1
      cw+=len(words)
```

```
    cc+=len(line)

print('no.of lines:',c1)
print('no.of words:',cw)
print('no.of characters:',cc)
#close the file
f.close()
```

Input/Output:

Enter file name:myfile.txt

No.of lines:3

No.of words:13

No.of characters:61

(b). Create a new file (Hello.txt) and copy the text to other file called target.txt. The target.txt file should store only lower case alphabets and display the number of lines copied

AIM: To develop a python program to Create a new file (Hello.txt) and copy the text to other file called target.txt. The target.txt file should store only lower case alphabets and display the number of lines copied.

Source code:

```
#create a file to store characters
#open the file for writing data
f=open('myfile.tex','w')
#enter characters from keyboard
str= input('enter text:')
#write the string into file
f.write(str)
```

```
#closing the file
f.close()


#open both files
with open('first.txt','r') as firstfile, open('second.txt','a') as secondfile:

    # read content from first file
    for line in firstfile:

        # append content to second file
        secondfile.write(line)
```

Input/Output:

```
========== RESTART: C:/Users/Dell Laptop/OneDrive/Desktop/pychan/text.py ==========
enter text:hello
========== RESTART: C:/Users/Dell Laptop/OneDrive/Desktop/pychan/text.py ==========
enter text:hello
```