PYTHON PROGRAMMING

LABORATORY MANUAL

**Subject Code :20CF00108**

**Regulations :Autonomous – R20**

**Class :ISemester**(MCA)

CHADALAWADA RAMANAMMA ENGINEERING COLLEGE (AUTONOMOUS)

Chadalawada Nagar, Renigunta Road, Tirupati – 517 506

Department of Master of Computer Applications

**1.1 IMPLEMENTING DIFFERENT DATA TYPES**

**Aim****:**To write a program to perform different Arithmetic Operations on numbers in Python

**Procedure:**

# Python program to demonstrate numeric value

a = 5

print("Type of a: ", type(a))

b = 5.0

print("\nType of b: ", type(b))

c = 2 + 4j

print("\nType of c: ", type(c))

**OutPut:**

Type of a: <class 'int'>

Type of b: <class 'float'>

Type of c: <class 'complex'>

|  |
| --- |
| **# Creating a String  with single Quotes**  String1 = 'Welcome to the Geeks World'  print("String with the use of Single Quotes: ")  print(String1)  # **Creating a String  with double Quotes**  String1 = "I'm a Geek"  print("\nString with the use of Double Quotes: ")  print(String1)  print(type(String1))    **# Creating a String with triple Quotes**  String1 = '''I'm a Geek and I live in a world of "Geeks"'''  print("\nString with the use of Triple Quotes: ")  print(String1)  print(type(String1))    **# Creating String with triple Quotes allows multiple lines**  String1 = '''Geeks              For              Life'''  print("\nCreating a multiline String: ")  print(String1) |

**Output:**

String with the use of Single Quotes:

Welcome to the Geeks World

String with the use of Double Quotes:

I'm a Geek

<class 'str'>

String with the use of Triple Quotes:

I'm a Geek and I live in a world of "Geeks"

<class 'str'>

Creating a multiline String:

Geeks

For

Life

**#Python Program to Access characters of String**

String1 = "GeeksForGeeks"

print("Initial String: ")

print(String1)

**# Printing First character**

print("\nFirst character of String is: ")

print(String1[0])

**# Printing Last character**

print("\nLast character of String is: ")

print(String1[-1])

Initial String:

GeeksForGeeks

First character of String is:

G

Last character of String is:

s

**# Python program to demonstrate Creation of List**

**# Creating a List**

List = []

print("Intial blank List: ")

print(List)

**# Creating a List with the use of a String**

List = ['GeeksForGeeks']

print("\nList with the use of String: ")

print(List)

**# Creating a List with the use of multiple values**

List = ["Geeks", "For", "Geeks"]

print("\nList containing multiple values: ")

print(List[0])

print(List[2])

**# Creating a Multi-Dimensional List (By Nesting a list inside a List)**

List = [['Geeks', 'For'], ['Geeks']]

print("\nMulti-Dimensional List: ")

print(List)

**Output:**

Intial blank List:

[]

List with the use of String:

['GeeksForGeeks']

List containing multiple values:

Geeks

Geeks

Multi-Dimensional List:

[['Geeks', 'For'], ['Geeks']]

|  |
| --- |
| **Python program to demonstrate accessing of element from list**    **# Creating a List with the use of multiple values**  List = ["Geeks", "For", "Geeks"]    **# accessing a element from the list using index number**  print("Accessing element from the list")  print(List[0])  print(List[2])    **# accessing a element using negative indexing**  print("Accessing element using negative indexing")    **# print the last element of list**  print(List[-1])    **# print the third last element of list**  print(List[-3]) |

**Output:**

Accessing element from the list

Geeks

Geeks

Accessing element using negative indexing

Geeks

Geeks

**Python program to demonstrate creation of Set**

# Creating an empty tuple

Tuple1 = ()

print("Initial empty Tuple: ")

print (Tuple1)

**# Creating a Tuple with the use of Strings**

Tuple1 = ('Geeks', 'For')

print("\nTuple with the use of String: ")

print(Tuple1)

**# Creating a Tuple with the use of list**

list1 = [1, 2, 4, 5, 6]

print("\nTuple using List: ")

print(tuple(list1))

**# Creating a Tuple with the use of built-in function**

Tuple1 = tuple('Geeks')

print("\nTuple with the use of function: ")

print(Tuple1)

**# Creating a Tuple with nested tuples**

Tuple1 = (0, 1, 2, 3)

Tuple2 = ('python', 'geek')

Tuple3 = (Tuple1, Tuple2)

print("\nTuple with nested tuples: ")

print(Tuple3)

**Output:**

Initial empty Tuple:

()

Tuple with the use of String:

('Geeks', 'For')

Tuple using List:

(1, 2, 4, 5, 6)

Tuple with the use of function:

('G', 'e', 'e', 'k', 's')

Tuple with nested tuples:

((0, 1, 2, 3), ('python', 'geek'))

**# Python program to demonstrate accessing tuple**

tuple1 = tuple([1, 2, 3, 4, 5])

**# Accessing element using indexing**

print("Frist element of tuple")

print(tuple1[0])

**# Accessing element from last negative indexing**

print("\nLast element of tuple")

print(tuple1[-1])

print("\nThird last element of tuple")

print(tuple1[-3])

**Output:**

Frist element of tuple

1

Last element of tuple

5

Third last element of tuple

3

|  |
| --- |
| **# Python program to demonstrateboolean type**  print(type(True))  print(type(False))    print(type(true)) |

**Output:**

<class 'bool'>

<class 'bool'>

Traceback (most recent call last):

File "/home/7e8862763fb66153d70824099d4f5fb7.py", line 8, in

print(type(true))

NameError: name 'true' is not defined

|  |
| --- |
| **# Python program to demonstrate Creation of Set in Python**    #Creating a Set  set1 = set()  print("Intial blank Set: ")  print(set1)    **# Creating a Set with the use of a String**  set1 = set("GeeksForGeeks")  print("\nSet with the use of String: ")  print(set1)    **# Creating a Set with the use of a List**  set1 = set(["Geeks", "For", "Geeks"])  print("\nSet with the use of List: ")  print(set1)    **# Creating a Set with**  **# a mixed type of values**  # (Having numbers and strings)  set1 = set([1, 2, 'Geeks', 4, 'For', 6, 'Geeks'])  print("\nSet with the use of Mixed Values")  print(set1) |

**Output:**

Intial blank Set:

set()

Set with the use of String:

{'F', 'o', 'G', 's', 'r', 'k', 'e'}

Set with the use of List:

{'Geeks', 'For'}

Set with the use of Mixed Values

{1, 2, 4, 6, 'Geeks', 'For'}

|  |
| --- |
| **# Creating an empty Dictionary**  Dict = {}  print("Empty Dictionary: ")  print(Dict)    **# Creating a Dictionary with Integer Keys**  Dict = {1: 'Geeks', 2: 'For', 3: 'Geeks'}  print("\nDictionary with the use of Integer Keys: ")  print(Dict)    **# Creating a Dictionary with Mixed keys**  Dict = {'Name': 'Geeks', 1: [1, 2, 3, 4]}  print("\nDictionary with the use of Mixed Keys: ")  print(Dict)    **# Creating a Dictionary with dict() method**  Dict = dict({1: 'Geeks', 2: 'For', 3:'Geeks'})  print("\nDictionary with the use of dict(): ")  print(Dict)    **# Creating a Dictionary  with each item as a Pair**  Dict = dict([(1, 'Geeks'), (2, 'For')])  print("\nDictionary with each item as a pair: ")  print(Dict) |

**Output:**

Empty Dictionary:

{}

Dictionary with the use of Integer Keys:

{1: 'Geeks', 2: 'For', 3: 'Geeks'}

Dictionary with the use of Mixed Keys:

{1: [1, 2, 3, 4], 'Name': 'Geeks'}

Dictionary with the use of dict():

{1: 'Geeks', 2: 'For', 3: 'Geeks'}

Dictionary with each item as a pair:

{1: 'Geeks', 2: 'For'}

**1.2 IMPLEMENTING ARITHMETIC OPERATIONS**

**Aim**: To write a program to perform different Arithmetic Operations on numbers in Python

num1 = int(input('Enter First number: '))

num2 = int(input('Enter Second number '))

add = num1 + num2

dif = num1 - num2

mul = num1 \* num2

div = num1 / num2

floor\_div = num1 // num2

power = num1 \*\* num2

modulus = num1 % num2

print('Sum of ',num1 ,'and' ,num2 ,'is :',add)

print('Difference of ',num1 ,'and' ,num2 ,'is :',dif)

print('Product of' ,num1 ,'and' ,num2 ,'is :',mul)

print('Division of ',num1 ,'and' ,num2 ,'is :',div)

print('Floor Division of ',num1 ,'and' ,num2 ,'is :',floor\_div)

print('Exponent of ',num1 ,'and' ,num2 ,'is :',power)

print('Modulus of ',num1 ,'and' ,num2 ,'is :',modulus)

**Out Put:**

**2.1 FINDING PRIME NUMBER**

**Aim:** To Write a Python program to find first n prime numbers.

**Procedure:**

numr=int(input("Enter range:"))

print("Prime numbers:",end=' ')

for n in range(1,numr):

    for i in range(2,n):

        if(n%i==0):

            break

    else:

        print(n,end=' ')

**Output:**

Enter range: 50

Prime numbers: 1 2 3 5 7 11 13 17 19 23 29 31 37 41 43 47

**2.1 FINDING MAXIMUM ELEMENT**

**Ai**[**m:** To](Im:To)Write a Python Program to find the maximum from a list of numbers.

|  |
| --- |
| list1 = [10, 20, 4, 45, 99]  # printing the maximum element  print("Largest element is:", max(list1)) |

**Output:**

Largest element is: 99

**3.1 FINDING GCD OF TWO NUMBERS**

**Aim:** To Write a Python program to find GCD of two numbers**.**

**Procedure:**

**Method1:**

#python program to find GCD of Two Numbers using While loop

Num1=int(input(“Enter first number:”))

Num2=int(input(“Enter second number:”))

I=1

While(i<=num1 and i<=num2):

If(num1%i==0 and num2%i==0):

Gcd=i

I=i+1

Print(“GCD is”,gcd)

**input:**

5

15

**Output:**

5

**Method 2:**

# Python code to demonstrate the working of gcd()

# importing "math" for mathematical operations

import math

# prints 12

print("The gcd of 60 and 48 is : ", end="")

print(math.gcd(60, 48))

**Output:**

The gcd of 60 and 48 is : 12

**Method 3:**

# Python code to demonstrate naive

# method to compute gcd( recursion )

def hcf(a, b):

    if(b == 0):

        return a

    else:

        return hcf(b, a % b)

a = 60

b = 48

  # prints 12

print("The gcd of 60 and 48 is : ", end="")

print(hcf(60, 48))

**Output**

The gcd of 60 and 48 is : 12

**3.2 FINDING THE SQUARE OF A NUMBER**

**Aim:** To Write a Python Program to find the square root of a number by Newton’s Method.

**Procedure:**

def newton\_method(number, number\_iters = 100):

    a = float(number)

    for i in range(number\_iters):

        number = 0.5 \* (number + a / number)

    return number

a=int(input("Enter first number:"))

b=int(input("Enter second number:"))

print("Square root of first number:",newton\_method(a))

print("Square root of second number:",newton\_method(b))

**Output:**

Enter first number:81

Enter second number:5

Square root of first number: 9.0

Square root of second number: 2.23606797749979

**4.1 MULTIPLICATION OF TWO MATRICES**

**Aim:** To Write a Python program to multiply matrices.

# Program to multiply two matrices using nested loops

# 3x3 matrix

X = [[12,7,3],

[4 ,5,6],

[7 ,8,9]]

# 3x4 matrix

Y = [[5,8,1,2],

[6,7,3,0],

[4,5,9,1]]

# result is 3x4

result = [[0,0,0,0],

[0,0,0,0],

[0,0,0,0]]

# iterate through rows of X

for i in range(len(X)):

# iterate through columns of Y

for j in range(len(Y[0])):

# iterate through rows of Y

for k in range(len(Y)):

result[i][j] += X[i][k] \* Y[k][j]

for r in result:

print(r)

**Output**

[114, 160, 60, 27]

[74, 97, 73, 14]

[119, 157, 112, 23]

**Method 2:**

# Program to multiply two matrices using list comprehension

# take a 3x3 matrix

A = [[12, 7, 3],

    [4, 5, 6],

    [7, 8, 9]]

# take a 3x4 matrix

B = [[5, 8, 1, 2],

    [6, 7, 3, 0],

    [4, 5, 9, 1]]

# result will be 3x4

result = [[sum(a \* b for a, b in zip(A\_row, B\_col))

                        for B\_col in zip(\*B)]

                                for A\_row in A]

for r in result:

    print(r)

**Output:**

[114, 160, 60, 27]

[74, 97, 73, 14]

[119, 157, 112, 23]

import sys

def revline(x):

i=0

z=len(open(x).readlines())

rev=[None]\*z

f=open(x)

while(i<z):

rev[i]=f.readline()

rev[i]=rev[i].strip()

print rev[i][::-1]

i=i+1

z=sys.argv[1]

revline(z)

**CONVERT A LIST OF TUPLES IN A DICTIONARY**

**Aim:** To Write a program to convert a list of tuples in a dictionary.

# Python code to convert into dictionary

def Convert(tup, di):

    for a, b in tup:

        di.setdefault(a, []).append(b)

    return di

# Driver Code

tups = [("akash", 10), ("gaurav", 12), ("anand", 14),

     ("suraj", 20), ("akhil", 25), ("ashish", 30)]

dictionary = {}

print (Convert(tups, dictionary))

Output:

{'akash': [10], 'gaurav': [12], 'anand': [14],

'ashish': [30], 'akhil': [25], 'suraj': [20]}

|  |
| --- |
| # Python code to convert into dictionary  list\_1=[("Nakul",93), ("Shivansh",45), ("Samved",65),             ("Yash",88), ("Vidit",70), ("Pradeep",52)]  dict\_1=dict()    for student,score in list\_1:      dict\_1.setdefault(student, []).append(score)  print(dict\_1) |

**Output:**

{'Nakul': [93], 'Shivansh': [45], 'Samved': [65], 'Yash': [88], 'Vidit': [70], 'Pradeep': [52]}

**TUPLE IMPLEMENTATION**

**Aim:**To Write a Python program to check whether an element exists within a tuple.

|  |
| --- |
| Python3 code to demonstrate working of  # Check if element is present in tuple  # using loop    # initialize tuple  test\_tup = (10, 4, 5, 6, 8)    # printing original tuple  print("The original tuple : " + str(test\_tup))    # initialize N  N = 6    # Check if element is present in tuple  # using loop  res = False  for ele in test\_tup :      if N == ele :          res = True          break    # printing result  print("Does tuple contain required value ? : " + str(res)) |

**Output :**

The original tuple : (10, 4, 5, 6, 8)

Does tuple contain required value ? : True

**Method 2**

|  |
| --- |
| # Python3 code to demonstrate working of Check if element is present in tuple Using in operator    # initialize tuple  test\_tup = (10, 4, 5, 6, 8)    # printing original tuple  print("The original tuple : " + str(test\_tup))    # initialize N  N = 6    # Check if element is present in tuple  # Using in operator  res = N in test\_tup    # printing result  print("Does tuple contain required value ? : " + str(res)) |

**Output :**

The original tuple : (10, 4, 5, 6, 8)

Does tuple contain required value ? : True

**LIST IMPLEMENTATION**

**Aim:** To Write a function dups to find all duplicates in the list.

def find\_duplicates(list\_of\_numbers):

#start writing your code here

x=set(list\_of\_numbers)

y=[]

dup=[]

count=0

for i in x:

y.append(i)

for i in y:

for j in list\_of\_numbers:

if(j==i):

count+=1

if count>=2:

dup.append(i)

break

count=0

return dup

list\_of\_numbers=[1,2,3]

list\_of\_duplicates=find\_duplicates(list\_of\_numbers)

print(list\_of\_duplicates)

**DICTIONARIES IMPLEMENTATION**

**Aim:** To Write a Python script to concatenate following dictionaries to create a new one.

Sample Dictionary :

dic1={1:10, 2:20}

dic2={3:30, 4:40}

dic3={5:50,6:60}

**Procedure:**

dic1={1:10, 2:20}

dic2={3:30, 4:40}

dic3={5:50,6:60}

dic4 = {}

for d in (dic1, dic2, dic3): dic4.update(d)

print(dic4)

**Output:**

{1: 10, 2: 20, 3: 30, 4: 40, 5: 50, 6: 60}

**REPLACINING DICTINARIES VALUES**

**Aim:** To Write a Python program to replace dictionary values with their average.

**Procedure:**

def sum\_math\_v\_vi\_average(list\_of\_dicts):

for d in list\_of\_dicts:

n1 = d.pop('V')

n2 = d.pop('VI')

d['V+VI'] = (n1 + n2)/2

return list\_of\_dicts

student\_details= [

{'id' : 1, 'subject' : 'math', 'V' : 70, 'VI' : 82},

{'id' : 2, 'subject' : 'math', 'V' : 73, 'VI' : 74},

{'id' : 3, 'subject' : 'math', 'V' : 75, 'VI' : 86}

]

print(sum\_math\_v\_vi\_average(student\_details))

**Output:**

[{'subject': 'math', 'id': 1, 'V+VI': 76.0}, {'subject': 'math', 'id': 2, 'V+VI': 73.5}, {'subject': 'math', '

id': 3, 'V+VI': 80.5}]

**FILE IMPLEMENTATION**

**Aim:** To write a python program to Read a file line by line in Python

# Python code to

# demonstrate readlines()

L = ["Geeks\n", "for\n", "Geeks\n"]

# writing to file

file1 = open('myfile.txt', 'w')

file1.writelines(L)

file1.close()

# Using readlines()

file1 = open('myfile.txt', 'r')

Lines = file1.readlines()

count = 0

# Strips the newline character

for line in Lines:

count += 1

print("Line{}: {}".format(count, line.strip()))

**Output:**

Line1: Geeks

Line2: for

Line3: Geeks

**# Python program to demonstrate readline()**

# Python program to

# demonstrate readline()

L = ["Geeks\n", "for\n", "Geeks\n"]

# Writing to a file

file1 = open('myfile.txt', 'w')

file1.writelines((L))

file1.close()

# Using readline()

file1 = open('myfile.txt', 'r')

count = 0

while True:

count += 1

# Get next line from file

line = file1.readline()

# if line is empty

# end of file is reached

if not line:

break

print("Line{}: {}".format(count, line.strip()))

file1.close()

**Output:**

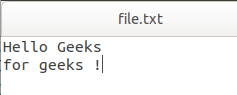
Line1 Geeks

Line2 for

Line3 Geeks

# Demonstrated Python Program

# to read file character by character



file = open('file.txt', 'r')

while 1:

# read by character

char = file.read(1)

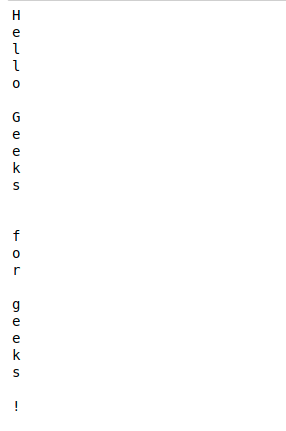
if not char:

break

print(char)

file.close()

Output



Write a program to print number of lines, words& characters present in the given file.