Assignment 1

1. Write a Program to a number n and print a single digit answer showing sum of digits of n.
2. Write a Program to read a number n and print n terms of Fibonacci series using regular for loop and for each Loop.
3. Write a Program to Ask the user for a string and print out whether this string is a palindrome or not.
4. Write a program which accepts a sequence of comma-separated numbers from console and generate a list.
5. Write a Program to remove all duplicates words from a given sentence using Dictionaries.
6. Write a program to demonstrate all Operations on Dictionary in Python.

Assignment 2

7. Demonstrate Lambda function on various arithmetic operations.
8. Write a program in Python which will accept marks of five students of five subjects and check them if they are pass or fail. (Assume suitable conditions for pass and fail)
9. Write a generator in python to generate Fibonacci series.
10. Write Python programs to Create a Python class named Circle constructed by a radius and two methods which will compute the area and the perimeter of a circle
11. Define class called Computer. Define subclasses of computers called desk top and laptop. Accept details of 3 categories. Show accepted information by overriding display method (Define at least three attributes in each class. Assume suitable data).
12. Write a program in Python to calculate telephone bill for 10 no of customers (if Units less than are equal to 100 rate is Rs.3.5/-Unit otherwise Rs.10/Unit). Create Suitable Package to define methods).
13. Write a Program to Generate Students Mark list Using Inheritance for 5 subject's marks. (Use Concept Class Variable, Class Method and Static Method by Decorators).
14. Write a program to add and retrieve dynamic attributes of classes for arithmetic operations.

Assignment 3

15. Write Programs using Regular Expressions to

    i. Validate Email Entered by user
    ii. Validate Password Entered by User
    iii. Validate URL

16. Write a program to Display Mark sheet of a student (Consider Marks of 5 subjects out of 100 each) using concept of Multithreading implement appropriate exceptions.
17. Write a program in python to Demonstrate Implementation of Exception Handling for calculation of telephone bill.

Assignment 4

18. Create a text file "intro.txt" in python and ask the user to write a single line text by user input.
19. Create a text file "MyFile.txt" in python and ask the user to write separate 3 lines with three input statements from the user.
20. Write a program to read the contents of both the files created in the above programs and merge the contents into "merge.txt". Avoid using the close() function to close the files
21. Read first n no. letters from a text file, read the first line, read a specific line from a text file.
22. Write programs in Python, which will accept details of employees in Employee table and saves in database EMP (as ID, Fname, Lname, Address etc.).Create Dabase, Table as mentioned and perform Insert, Update and Delete operations on it by End User.

Assignment 5

23. Create two 2-D arrays and Plot them using matplotlib (All types of plots are expected)
24. Return array of odd rows and even columns from below numpy array

```
sampleArray = numpy.array([[3 ,6, 9, 12], [15 ,18, 21, 24],
                           [27 ,30, 33, 36], [39 ,42, 45, 48], [51 ,54, 57, 60]])
```

Expected Output:

```
Printing Input Array
[[ 3  6  9 12]
 [15 18 21 24]
 [27 30 33 36]
 [39 42 45 48]
 [51 54 57 60]]

Printing array of odd rows and even columns
[[ 6 12]
 [30 36]
 [54 60]]
```

25. Write a NumPy program to compute the histogram of nums against the bins. Output:
    nums: [0.5 0.7 1. 1.2 1.3 2.1]
    bins: [0 1 2 3]
26. Write a NumPy program to compute the mean, standard deviation, and variance of a given array along the
    second axis.
    Original array:
    [0 1 2 3 4 5]
27. Write a Pandas program to select the specified columns and rows from a given data frame.
28. Write a Program for performing CRUD operation with MongoDB and Python
29. Write a Program to find out Null values and Missing values of a dataset. (Assume Suitable Dataset)

Q1. Write a Program to a number n and print a single digit answer showing sum of digits of n.

```
# Sum of Digits
def sum_of_digits(n):
    sum1 = 0
    while (n != 0):
        digit = n % 10
        sum1 = sum1 + digit
        n = n // 10
    if(sum1>9):
        return sum_of_digits(sum1)
    return sum1
print("Enter the Number: ", end='')
num=int(input())
print("Sum is: ", sum_of_digits(num))
```

Q2. Write a Program to read a number n and print n terms of Fibonacci series using regular for loop and for each Loop.

```
# Fibonacci Series
print("Enter the number of terms for Fibonacci series: ", end='')
n=int(input())
a=0
b=1
print(a,b, end=' ')
for i in range(n-2):
    c=a+b
    print(c, end=' ')
    a=b
    b=c
```

3. Write a Program to Ask the user for a string and print out whether this string is a palindrome or not.

```
# String Palindrome
print("Enter the String to check for Palindrome: ",end='')
str=input()
reverse_str=str[::-1]
if(reverse_str==str):
    print("The String is Palindrome")
else:
    print("The String is not Palindrome”)
```

4. Write a program which accepts a sequence of comma-separated numbers from console and generate a list.

```
print("Enter the numbers with comma separation: ", end='')
a=input()
str_list=a.split(",")
num_list=[]
for i in str_list:
    num_list.append(int(i))

print("List is: “,num_list)
```

5. Write a Program to remove all duplicates words from a given sentence using Dictionaries.

```
from collections import Counter

def rem_duplicates(str):
    str=str.split(" ")
    words=Counter(str)          #Making Dictionary using Counter
    s=" ".join(words.keys())    #Counter stores words as keys & their frequency as value
    print(s)

str=input("Enter the String to remove duplicates: ")
print("After Removing Duplicates: ",end='')
rem_duplicates(str)
```

6. Write a program to demonstrate all Operations on Dictionary in Python.

```
# Perform all Operations on Dictionary in Python
print("Creating Dictionary...")
dict={1:"one",2:"two",3:"three"}
print("Created Dictionary: ",dict)

print("\nAccessing an element...")
print("Accessed value with key 2: ",dict[2])

print("\nGetting a list of all the keys...")
print("The list of keys are: ",dict.keys())

print("\nGetting a list of all the values...")
print("The list of values are: ",dict.values())

print("\nAdding an entry...")
dict[4]="four"
print("The Dictionary after adding an entry: ",dict)

print("\nChanging an entry...")
dict[1]="First"
print("The Dictionary after changing an entry: ",dict)
print("\nDeleting an entry...")
del dict[4]
print("The Dictionary after deleting an entry: ",dict)
print("\nMaking a copy of this dictionary...")
dict_copy=dict.copy()
print("The copy of Dictionary: ",dict_copy)
print("\nPrinting number of items this dictionary...")
print("The number of items in Dictionary: ",len(dict))

print("\nTest if this dictionary has a particular key...")
if 2 in dict:
    print("Yes, 2 is one of the keys in the dictionary")
print("\nLooping over keys...")
print("Keys are: ")
for i in dict.keys(): print(i,end=" ")
print("\n\nLooping over values...")
print("Values are: ")
for i in dict.values(): print(i,end=" ")
print("\n\nUsing if statement to get values...")
print("Printing Value if 1 in dictionary...")
if 1 in dict:
    print(dict[1])
print("\nPrinting not found if 4 not in dictionary...")
if 4 not in dict:
    print("4 key not found in Dictionary")

print("\nDeleting 3 if in dictionary...")
if 3 in dict:
    del dict[3]
print("Dictionary after deleting: ",dict)

print("\nRemoving all elements from dictionary...")
dict.clear()
print("Dictionary after clearing all elements: “,dict)
```

7. Demonstrate Lambda function on various arithmetic operations.

```
print("Addition by Lambda...")
x = lambda a : a + 10
print(x(5))
print("Subtraction by Lambda...")
x = lambda a : a - 10
print(x(20))
print("Multiplication by Lambda...")
x = lambda a, b : a * b
print(x(5,10))
print("Division by Lambda...")
x = lambda a, b : a / b
print(x(100,20))
print("Modulus by Lambda...")
x = lambda a, b : a % b
print(x(15, 7))
print("Floor Division by Lambda...")
x = lambda a, b : a // b
print(x(128,10))
```

8. Write a program in Python which will accept marks of five students of five subjects and check them if they are pass or fail. (Assume suitable conditions for pass and fail)

```
def pass_or_fail(x):
    for i in x:
        if min(x[i])<40:
            print(f"Student {i} has Failed")
        else:
            print(f"Student {i} has Passed")

for i in range(5):
    student={}
    name=input("Enter the Name of Student: ")
    marks=[]
    for j in range(5):
        print(f"Enter the Marks in subject {j+1}: ",end="")
        j=int(input())
        marks.append(j)
    student[name]=marks
    pass_or_fail(student)
```

9. Write a generator in python to generate Fibonacci series.

```
def fib_generator(n):
    a,b=0,1
    for i in range(n):
        yield a
        a, b = b, a+b
num=int(input("Enter the no. of terms: "))
print(list(fib_generator(num)))
```

10. Write Python programs to Create a Python class named Circle constructed by a radius and two methods which will compute the area and the perimeter of a circle

```
import math
class Circle:
    def __init__(self, r):
        self.r=r

    def calc_area(self, r):
        area=math.pi*(r**2)
        return area

    def calc_peri(self, r):
        peri=2*math.pi*r
        return peri

radius=float(input("Enter the Radius of Circle: "))
c=Circle(radius)
print("Perimeter is: ",c.calc_peri(radius))
print("Area is: ",c.calc_area(radius))
```

11. Define class called Computer. Define subclasses of computers called desk top and laptop. Accept details of 3 categories. Show accepted information by overriding display method (Define at least three attributes in each class. Assume suitable data).

```
class Computer:
    def __init__(self,m="",model="",pno=""):
        self.Manufacturer=m
        self.model=model
        self.pno=pno
    def display(self):
        print("Displaying Computer Details")
    def get_input(self):
        pass

class Desktop(Computer):
    def __init__(self,msize=0):
        self.monitor_size=msize
    def get_input(self):
        print("\nEnter the Desktop Details...")
        self.Manufacturer= input("Enter the Manufacturer: ")
        self.model = input("Enter the Model: ")
        self.pno = input("Enter the Product No.: ")
        self.monitor_size = input("Enter the Monitor Size: ")
    def display(self):
        print("\nDisplaying Desktop Details...")
        print("Manufacturer is:", self.Manufacturer)
        print("Model is:" , self.model)
        print("Product No. is:" , self.pno)
        print("Monitor Size is:" , self.monitor_size, "inches")

class Laptop(Computer):
    def __init__(self,ssize=0,wattage=0):
        self.screen_size=ssize
        self.charger_wattage=wattage
    def get_input(self):
        print("\nEnter the Laptop Details...")
        self.Manufacturer = input("Enter the Manufacturer: ")
        self.model = input("Enter the Model: ")
        self.pno = input("Enter the Product No.: ")
        self.screen_size = input("Enter the Screen Size: ")
        self.charger_wattage=input("Enter the Charger Wattage: ")
    def display(self):
        print("\nDisplaying laptop Details...")
        print("Manufacturer is:", self.Manufacturer)
        print("Model is: " , self.model)
        print("Product No. is:" , self.pno)
        print("Screen Size is:" , self.screen_size, "inches")
        print("Charger Wattage is:" , self.charger_wattage, "Watts")

d1=Desktop()
l1=Laptop()
l2=Laptop()
d1.get_input()
l1.get_input()
l2.get_input()
d1.display()
l1.display()
l2.display()
```

12. Write a program in Python to calculate telephone bill for 10 no of customers (if Units less than are equal to 100 rate is Rs.3.5/-Unit otherwise Rs.10/Unit). Create Suitable Package to define methods).

```
# Module made for A2Q6 (File 1)

def calc_bill(n):
    bill=0
    if n<=100:
        bill=n*3.5
    else:
        bill=n*10
    return bill


# File 2

import myModuleQ6 as mx

cust_no=int(input("Enter no. of customers: "))
customers={}
for i in range(cust_no):
    customers[i+1]=int(input(f"Enter units for Customer {i+1}: "))

for j in customers:
    print(f"Bill for Customer {j}:", mx.calc_bill(customers[j]))
```

13. Write a Program to Generate Students Mark list Using Inheritance for 5 subject's marks. (Use Concept Class Variable, Class Method and Static Method by Decorators).

```
class student:
    def __init__(self):
        self.name=''

    def get_name(self):
        self.name=input("Enter the Name of Student: ")

class marks(student):
    address="Pune"

    def __init__(self):
        self.marks_list=[]
        student.__init__(self)

    def set_marks(self):
        for i in range(5):
            print(f"Enter the Marks in subject {i+1}: ",end="")
            i=int(input())
            self.marks_list.append(i)

    def get_marks(self):
        print(f"Name: {self.name}")
        print(f"Marks: {self.marks_list}")

    @classmethod
    def change_add(cls,address_change):
        marks.address=address_change

    @staticmethod
    def change_name(name_chg):
        return name_chg.split()[0]

sm = marks()
sm.get_name()
sm.set_marks()
sm.get_marks()
print("Address (Class Variable) before change:",sm.address)
marks.change_add("Delhi")
print("Address after change (Using Class Method):",sm.address)
y=marks.change_name("Indira College")                    #Using Static Method
print("Printing after change (Using Static Method):",y)
```

14. Write a program to add and retrieve dynamic attributes of classes for arithmetic operations.

```
class attributes:
    pass

c1 = attributes()
c1.attr1=int(input("Enter 2 numbers to add as attributes: "))   # Dynamic Attribute 1
c1.attr2=int(input())                                          # Dynamic Attribute 2
print("Addition:", c1.attr1 + c1.attr2)
print("Subtraction:", c1.attr2 - c1.attr1)
print("Multiplication:", c1.attr1 * c1.attr2)
print("Division:", c1.attr2 / c1.attr1)
```

15. Write Programs using Regular Expressions to

i. Validate Email Entered by user
ii. Validate Password Entered by User
iii. Validate URL

```
import re

email = input("Enter your Email ID: ")
ematch = "^[a-zA-Z0-9]+[\._]?[a-zA-Z0-9]+[@]\w+[.]\w{2,3}$"      #Abc1_pq23@gmail.com
result1 = re.match(ematch, email)

# Password - At least 1 Capital Letter, 1 digit, 1 Special Character (Length Should be 8-18)

password = input("Enter your password: ")
pmatch = "^(?=.*[a-z])(?=.*[A-Z])(?=.*\d)(?=.*[@$!%*#?&])[A-Za-z\d@$!#%*?&]{8,18}$"
result2 = re.match(pmatch, password)

# URL must start with http/https, followed by :// , then it must contain www.
# followed by subdomain of length (2, 256) and
# last part should contain top level domain like .com, .org etc.

url = input("Enter any URL: ")
umatch = "((http|https)://)(www.)?[a-zA-Z0-9@:%._\\+~#?&//=]{2,256}\\.[a-z]{2,6}\\b([a-zA-Z0-9@:
%._\\+~#?&//=]*)"
result3 = re.match(umatch, url)

if result1:
    print("Email Validated...")
else:
    print("Email ID Validation failed!!!")

if result2:
    print("Password format Validated...")
else:
    print("Wrong format for Password!!!")

if result3:
    print("URL Validated...")
else:
    print("Wrong URL Format!!!")
```

16. Write a program to Display Mark sheet of a student (Consider Marks of 5 subjects out of 100 each) using concept of Multithreading implement appropriate exceptions.

```
import threading
import time

def calcTotalMarks(marks):
    time.sleep(1)
    print("Total Marks:", sum(marks.values()))

def calcPercentage(marks):
    time.sleep(2)
    print("Percentage:", float(sum(marks.values()) / (len(marks.values()))),"%")

def calcGrade(marks):
    time.sleep(3)
    percentage = float(sum(marks.values()) / (len(marks.values())))
    if(percentage>=90):
        print("Grade: A")
    elif(percentage>=80 and percentage<90):
        print("Grade: B")
    elif(percentage>=70 and percentage<80):
        print("Grade: C")
    elif(percentage>=60 and percentage<70):
        print("Grade: D")
    else:
        print("Grade: F")

student_name = "ABC"
student_marks = {"Maths":80, "DSA":88, "Java":77, "Python":90, "AIT":65}
t1 = threading.Thread(target=calcTotalMarks, args=(student_marks,))
t2 = threading.Thread(target=calcPercentage, args=(student_marks,))
t3 = threading.Thread(target=calcGrade, args=(student_marks,))

print("---------- MARKSHEET ----------")
print("Name:", student_name, "\n")
for sub in student_marks:
    print(f"{sub}: {student_marks[sub]}")
print()
t1.start()
t2.start()
t3.start()

t1.join()
t2.join()
t3.join()
```

17. Write a program in python to Demonstrate Implementation of Exception Handling for calculation of telephone bill.

```
try:
    calls = int(input("Enter number of calls: "))
    if calls>=0 and calls <= 100:
        bill = 3.5*calls                #3.5/unit till 100
    elif calls > 100 and calls <= 150:
        bill = (3.5*100) + (5*(calls-100))     #Additional 5/unit till 150
    elif calls > 150 and calls <= 200:
        bill = (3.5*100) + (5*50) + (7*(calls-150))     #Additional 7/unit till 200
    elif calls>200:
        bill = (3.5*100) + (5*50) + (7*50) + (10*(calls-200))       #Additional 10/unit over 200
    elif calls<0:
        raise ValueError

    print("Total Bill Amount is:", bill)

except ValueError:
    print("ValueError Exception... Please Enter Valid Number!!")
except NameError:
    print("NameError Exception... Please Try Again With Valid Input!!")
```

18. Create a text file “intro.txt” in python and ask the user to write a single line text by user input.

```
def write_in_file():
    f = open("intro.txt","w")
    text=input("Enter the text: ")
    f.write(text)
    f.close()

write_in_file()
```

19. Create a text file “MyFile.txt” in python and ask the user to write separate 3 lines with three input statements from the user.

```
def write_lines():
    f = open("MyFile.txt","w")
    new_line="\n"
    for i in range(1,4):
        line=input(f"Enter line {i}: ")
        f.write(line)
        f.write(new_line)
    f.close()

write_lines()
```

20. Write a program to read the contents of both the files created in the above programs and merge the contents into “merge.txt”. Avoid using the close() function to close the files

```
def merge_files():
    with open("intro.txt","r") as f1:
        data1=f1.read()
    with open("MyFile.txt","r") as f2:
        data2=f2.read()
    with open("merge.txt","w") as f3:
        f3.write(data1)
        new_line="\n"
        f3.write(new_line)
        f3.write(data2)

merge_files()
```

21. Read first n no. letters from a text file, read the first line, read a specific line from a text file.

```
def read_from_file():
    n = int(input("Enter no. of letters to read: "))
    nth_line=int(input("Enter specific line no. to read: "))
    with open("merge.txt","r") as f1:
        nchar=f1.read(n)
        print(f"First {n} letters (including newline characters):", nchar)
    with open("merge.txt","r") as f2:
        line1=f2.readline()
        print("The first line of file:", line1)
    with open("merge.txt","r") as f3:
        nline=f3.readlines()
        print(f"Line {nth_line} is:", nline[nth_line-1])

read_from_file()
```

22. Write programs in Python, which will accept details of employees in Employee table and saves in database EMP (as ID, Fname, Lname, Address etc.).Create Dabase, Table as mentioned and perform Insert, Update and Delete operations on it by End User.

```
import pymongo

myclient = pymongo.MongoClient("mongodb://localhost:27017/")

mydb = myclient["EMP"]
mycol = mydb["Employee"]

while True:
    print("\nWhat do you want to do?.....")
    print("1. Insert\n2. Update\n3. Delete\n4. Print Table\n5. Exit")
    choice=int(input("Choice -> "))

    if(choice==1):
        n=int(input("How many records do you want to enter?: "))
        for i in range(n):
            id=int(input("Enter ID: "))
            fname=input("Enter First Name: ")
            lname=input("Enter Last Name: ")
            address=input("Enter the Address: ")
            mydict = { "ID": id, "FName": fname, "LName": lname, "Address": address }
            x = mycol.insert_one(mydict)

    elif(choice==2):
        update_col=input("Enter the name of attribute to be updated: ")
        present_value=input("Enter the value to be updated: ")
        query={update_col: present_value}
        new_value=input("Enter the new value: ")
        new_query={"$set":{update_col: new_value}}
        mycol.update_one(query, new_query)
        print("Value Updated...")

    elif(choice==3):
        del_col=input("Enter the name of attribute in which you will delete: ")
        del_value=input("Enter the value to be deleted: ")
        query={del_col: del_value}
        mycol.delete_one(query)
        print("Tuple Deleted...")

    elif(choice==4):
        for i in mycol.find(projection={'_id': False}):
            print(i)

    elif(choice==5):
        break

    else:
        print("Wrong Choice...!!")
        break
```

23. Create two 2-D arrays and Plot them using matplotlib (All types of plots are expected)

```
import numpy as np
import matplotlib.pyplot as plt

array1 = np.array([[3, 6, 9], [9, 12, 15]])
array2 = np.array([[9, 12, 15], [18, 21, 24]])

plt.title("Matplotlib Plot NumPy Array")
plt.xlabel("X-axis")
plt.ylabel("Y-axis")

plt.plot(array1, array2)
plt.scatter(array1, array2)
plt.step(array1, array2)

plt.show()
```

24. Return array of odd rows and even columns from below numpy array

```
sampleArray = numpy.array([[3 ,6, 9, 12], [15 ,18, 21, 24],
                           [27 ,30, 33, 36], [39 ,42, 45, 48], [51 ,54, 57, 60]])
```

Expected Output:

```
Printing Input Array
[[ 3  6  9 12]
 [15 18 21 24]
 [27 30 33 36]
 [39 42 45 48]
 [51 54 57 60]]

Printing array of odd rows and even columns
[[ 6 12]
 [30 36]
 [54 60]]
```

Solution:

```
import numpy as np

arr = np.array([[3, 6, 9, 12], [15, 18, 21, 24], [27, 30, 33, 36], [39, 42, 45, 48], [51, 54, 57, 60]])
print("Printing Input Array...")
print(arr)

print("\nPrinting Array of Odd Rows and Even Columns...")
new_arr = arr[::2, 1::2]
print(new_arr)
```

25. Write a NumPy program to compute the histogram of nums against the bins. Output:
    nums: [0.5 0.7 1. 1.2 1.3 2.1]
    bins: [0 1 2 3]

```
import numpy as np
import matplotlib.pyplot as plt

nums = np.array([0.5, 0.7, 1.0, 1.2, 1.3, 2.1])
bins = np.array([0, 1, 2, 3])
print("Nums:", nums)
print("Bins:", bins)
plt.hist(nums, bins)
plt.show()
```

26. Write a NumPy program to compute the mean, standard deviation, and variance of a given array along the
second axis.
Original array:
[0 1 2 3 4 5]

```
import numpy as np

x = np.arange(6)
print("Original Array:", x)
mean = np.mean(x)
print("\nMean:", mean)

sd = np.std(x)
print("Standard Deviation:", sd)

var= np.var(x)
print("Variance:", var)
```

27. Write a Pandas program to select the specified columns and rows from a given data frame.

```
import pandas as pd
import numpy as np

exam_data = {'Name': ['Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew',
'Laura', 'Kevin', 'Jonas'],
        'Score': [12.5, 9, 16.5, np.nan, 9, 20, 14.5, np.nan, 8, 19],
        'Attempts': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],
        'Qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes']}

labels = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

df = pd.DataFrame(exam_data, index=labels)
print("Printing given Data Frame...\n")
print(df)
print("\nSelecting Specific Rows and Columns...\n")
print(df.iloc[[1, 3, 5, 6], [1, 3]])
```

28. Write a Program for performing CRUD operation with MongoDB and Python

```
import pymongo

myclient = pymongo.MongoClient("mongodb://localhost:27017/")

mydb = myclient["STUDENT"]
mycol = mydb["Student"]

while True:
    print("\nWhat do you want to do?.....")
    print("1. Insert\n2. Update\n3. Delete\n4. Print Table\n5. Exit")
    choice=int(input("Choice -> "))

    if(choice==1):
        n=int(input("How many records do you want to enter?: "))
        for i in range(n):
            roll=input("Enter Roll No.: ")
            name=input("Enter Name: ")
            mydict = { "Roll": roll, "Name": name }
            x = mycol.insert_one(mydict)

    elif(choice==2):
        update_col=input("Enter the name of attribute to be updated: ")
        present_value=input("Enter the value to be updated: ")
        query={update_col: present_value}
        new_value=input("Enter the new value: ")
        new_query={"$set":{update_col: new_value}}
        mycol.update_one(query, new_query)
        print("Value Updated...")

    elif(choice==3):
        del_col=input("Enter the name of attribute in which you will delete: ")
        del_value=input("Enter the value to be deleted: ")
        query={del_col: del_value}
        mycol.delete_one(query)
        print("Tuple Deleted...")

    elif(choice==4):
        for i in mycol.find(projection={'_id': False}):
            print(i)

    elif(choice==5):
        break

    else:
        print("Wrong Choice...!!")
        break
```

29. Write a Program to find out Null values and Missing values of a dataset. (Assume Suitable Dataset)

```
import pandas as pd
import numpy as np

exam_data = {'Name': ['James', 'Emily', 'Michael', 'Matthew', 'Laura'],
        'Score': [np.nan, 9, 20, 14.5, np.nan],
        'Attempts': [3, 2, np.nan, 1, 1],
        'Qualify': ['no', 'no', 'yes', np.nan, 'no']}

labels = [1, 2, 3, 4, 5]

df = pd.DataFrame(exam_data, index=labels)
print("Original Data Frame...\n")
print(df)
print("\nMissing and Null Values of the Data Frame (Represented by 'True')...\n")
print(pd.isnull(df))
```