Hindi Vidya Prachar Samiti's

RAMNIRANJAN JHUNJHUNWALA COLLEGE OF ARTS, SCIENCE & COMMERCE (AUTONOMUS)

Python for Data Science

NAAC Re-Accredited 'A' Grade

विद्यया अमृतं अश्नुते

ESTD. 1963

RJC

ISO 9001:2015

Best College

UNIVERSITY OF MUMBAI

Name: Siddhi Chavan

Roll No-: 711

Class: MSc Data Science and Artificial Intelligence Part-I

Ramniranjan Jhunjhunwala College of Arts, Science and Commerce

Department of Data Science and Artificial Intelligence

CERTIFICATE

This is to certify Ms. Siddhi Chavan of MSc. Data Science and Artificial Intelligence, Roll no. 711 has successfully completed the practical of PYTHON FOR DATA SCIENCE during the Academic Year 2023-2024.

Date:

Prof. Mujtaba Shaikh
(Prof-In-Charge)

External Examiner

INDEX			
Sr. No	Practical Name	Date	Signature
UNIT-I			
1.1	Understanding Variables	07-08-2023	
1.2	Understanding operators andData types	08-08-2023	
1.3	Understanding control flowStatements	14-08-2023	
1.4	Understanding data structures	22-08-2023	
1.5	Understanding file handling	04-09-2023	
UNIT-II			
2.1	Understanding functions	16-09-2023	
2.2	Understanding Object OrientedProgramming Classes and Objects Inheritance and it's type	09-10-2023	
	Encapsulation	10-10-2023	
	Polymorphism	10-10-2023	
2.3	Understanding Exception Handling	10-10-2023	
UNIT-III			
3.1	Understanding Advanced PythonProgramming	16-10-2023	
3.2	Perform CRUD operation in streamlit	25-09-2023	
3.3	Understanding module pandas	18-09-2023	
UNIT-IV			
4.1	Understanding module NumPy	18-09-2023	
4.2	Understanding Visualization	27-09-2023	

PRACTICAL – I

Date: 07-08-2023

AIM: Understanding Variables

```
#Camel Case
myHappyday="07"
print(myHappyday)
#Pascal Case
MyHappyDay="07"
print(MyHappyDay)
#Snake Case
my_happy_day="07"
print("my_happy_day")

x = 10
print(type(x))
```

```
<class 'int'>
```

```
x='DSAI'
type(x)
```

```
str
```

```
#Assingment Operator Usage
a,b,c='apple','mango','cherry'
print(a)
print(b)
print(c)
```

```
apple
mango
cherry
```

```
x=y=z='dsai'
print(x,y,z)
```

```
dsai dsai dsai
```

```
bazar=['fish','sugar','rice']
f,s,r=bazar
r
```

```
'rice'
```

```
animals=['snake','elephat','frog','worm','mud frog']
reptiles,mammals,*amphibains=animals
print(reptiles,mammals,amphibains)
```

```
snake elephat ['frog', 'worm', 'mud frog']
```

```
food=['apple','ginger','garlic','cumin','fish']
fruit,*vegetable,meat=food
print(fruit)
print(vegetable)
print(meat)
```

```
apple
['ginger', 'garlic', 'cumin']
fish
```

```
a="Msc" #Concatenation
c="DSAI"
print(a+c)
```

```
MscDSAI
```

```
x="apple" #Golden Variable
def myfuc():
  print("Fruit name",x)
myfuc()
```

```
Fruit name apple
```

```
#Local Variable
x="abc"
def d():
   x="def"
   print("after abc:",x)
d()
print("first three letters:",x)
```

```
after abc: def
first three letters: abc
```

SIGNATURE: _____

PRACTICAL – II

Date: 08-08-2023

AIM: Understanding Operators and Data types

ARITHEMATIC OPERATOR

```
a = 9
b = 4
# Addition of numbers
add = a + b
# Subtraction of numbers
sub = a - b
# Multiplication of number
mul = a * b
# Modulo of both number
mod = a % b
# Power
p = a ** b
# Division
d=a/b
#Floor Division
fd=a//b
# print results
print(add)
print(sub)
print(mul)
print(mod)
print(p)
print(d)
print(fd)
```

```
13
5
36
1
6561
2.25
2
```

```
"""W.A.P to 3 i/p from users and perform addition\n
of 2 numbers and resultant of addn value is multiplied by the third
number of \n
user"""
a=int(input("enter first numberer:"))
b=int(input("enter second number:"))
c=int(input("enter third number:"))
print("Result")
print((a+b)*c)
```

```
5
6
9
99
```

COMPARSION OPERATOR

```
a = 13
b = 33
# a > b is False
print(a > b)
# a < b is True
print(a < b)
# a == b is False
print(a == b)
# a != b is True
print(a != b)
# a >= b is False
print(a >= b)
# a <= b is True
print(a <= b)
```

```
False
True
False
True
False
True
```

LOGICAL OPERATOR

```
#AND
x = 10
print(x > 5 and x < 15)
#This returns True because 10 is greater than 5 AND 10 is less than 15.

#OR
x = 10
print(x > 5 or x < 2)
#This returns True because one of the conditions are true.
#10 is greater than 5, but 10 is not less than 2.

#NOT
x = 10
print(not(x > 5 and x < 15))
#This returns False because not reverses the result of and.
```

```
True
True
False
```

MEMBERSHIP OPERATOR

```
#in
x = "Hello, World!"
print("ello" in x) #Returns true as it exists
print("hello" in x) #Returns false as 'h' is in lowercase

#not in
x = "Hello, World!"
print("ello" not in x) #Returns false as it exists
print("hello" not in x) #Returns true as it does not exist
```

```
True
False
False
True
```

DATA TYPES

```
a=int(input()) #int
print(a)

b=str('abc') #string
print(b)

e=["apple","banana","cherry"]   #for list
print(e)

f=("apple","banana","cherry")   #for tuple
print(f)

g=dict(name="arjun",age=20)    #for dictonary
print(g)

h={"apple","banana","cherry"}   #for set
print(h)

i={("apple","banana","cherry")} #for frozenset
print(i)

j=range(6)                      #for range
print(j)

g=bool(0)
print(g)

f=bytes(5)
print(f)
```

```
3
3
abc
['apple', 'banana', 'cherry']
('apple', 'banana', 'cherry')
{'name': 'arjun', 'age': 20}
{'banana', 'cherry', 'apple'}
{('apple', 'banana', 'cherry')}
range(0, 6)
False
b'\x00\x00\x00\x00\x00'
```

SIGNATURE: _____

PRACTICAL – III

Date: 14-08-2023

AIM: Understanding Control flow statements

If…elif…else statements

```
#if statement
## W.a.p to check even number in py the number is taken from user
a=int(input("enter a number"))
if a%2==0:
   print("it is even")
```

```
enter a number12
it is even
```

```
#if-else
#W.A.P to check for Voting criteria
age=int(input("Enter your age. .."))
if age>=18:
     print("YOU ARE ELIGIBLE TO VOTE")
else:
    print("YOU ARE NOT ELIGIBLE TO VOTE YOU HAVE TO WAIT FOR ",18-age,"more
years")
```

```
Enter your age....18
YOU ARE ELIGIBLE TO VOTE
```

```
#if...elif.  else
num=int(input("Enter the number?"))
if num==10:
    print("number is equals to 10")
elif num==50:
    print("number is equals to 50")
elif num==100:
    print("number if equals to 100")
else:
    print("number is not equal to 10 50 100")
```

```
Enter the number?10
number is equals to 10
```

```
#Nested
number = 5
# outer if statement
if (number >= 0):
   # inner if statement
   if number == 0:
     print('Number is 0')
   #Inner else statement
   else:
     print('Number is positive')
# outer else statement
else:
print('Number is negative')
```

```
Number is positive
```

for loop

```
L1=["a","b","c","d"]
for val in L1:
     print(val)
```

```
a
b
c
d
```

While loop

```
#While Loop
n=10
sum=0
i=1
while i <=10:
    sum=sum+i
    i+=1
print(sum)
```

```
55
```

Break statement

```
a='dsai'
for i in a:
  print(a)
  break #to break after one iteration
```

```
dsai
```

Continue statement

```
for i in range(1, 11):
  if i == 6:
     continue
  else:
    print(i, end = " ")
```

```
1 2 3 4 5 7 8 9 10
```

Pass statement

```
for x in [0, 1, 2]:
  pass
```

SIGNATURE: _____

PRACTICAL – IV

Date: 22-08-2023

AIM: Understanding Data Structures

List and some of it's method

```
L4=[]
for i in range (1,11): #append() to add in the last
     L4.append(i)
print(L4)
```

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

```
x=list(range(1,11))
x
```

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

```
#Nested Lists
L=['ab',1,2,3,['d']]
L
```

```
['ab', 1, 2, 3, ['d']]
```

```
# How to access the items in the list
#using indexing
#list name[specific index]
L=['ab',1,2,3,['d']]
print(L[1])
L=['ab',1,2,3,['d']]
print(L[-1]) #negative indexing for accessing last item
```

```
1
['d']
```

```
#Slicing
L=['r','i','c','h','a','r','d']
print(L[2:5])
print(L[:-1])
print(L[-1:])
print(L[0:5:2]) #start:stop:jump
```

```
['c', 'h', 'a']
['r', 'i', 'c', 'h', 'a', 'r']
['d']
['r', 'c', 'a']
```

```
# desirable location save
L[2]=3
L
```

```
['r', 'i', 3, 'h', 'a', 'r', 'd']
```

```
L.remove('r')
L
```

```
['i', 3, 'h', 'a', 'r', 'd']
```

```
L.pop(0)
L
```

```
[3, 'h', 'a', 'r', 'd']
```

```
L.clear()
L
```

```
[]
```

```
del(L)
```

Tuple

```
#Tuple
t1=('rohan','roshni','roshan','rohini')
for x in t1:
   print(x)
```

```
rohan
roshni
roshan
rohini
```

```
tuplee=()
a=input()
tuplee=a
print(tuplee)
```

```
ri
ri
```

Dictionary

```
#Dictionary
dic={1:'apple',2:'ball'}
type(dic)
```

```
dict
```

```
mydict=dict([(1,'apple'),(2,'ball')])
type(mydict)
```

```
dict
```

```
student={1:'arjun',2:'varun','age':20}
print(student)
print(student.get(2))
print(student[1])
student['age']=21
print(student)
```

```
{1: 'arjun', 2: 'varun', 'age': 20}
varun
arjun
{1: 'arjun', 2: 'varun', 'age': 21}
```

SIGNATURE: _____

PRACTICAL – V

Date: 04-09-2023

AIM: Understanding File Handling

```
# opening a file with python
myFile = open('/content/drive/MyDrive/python/alan.txt','r')
if myFile:
      print('file opened succcesfully')
      content = myFile.read()
      print(content)
myFile.close()
```

```
file opened succcesfully
Born in Maida Vale, London, Turing was raised in southern England.
He graduated at King's College, Cambridge, with a degree in mathematics.
he was a fellow at Cambridge, he published a proof demonstrating
that some purely mathematical yes-no questions can never be answered
by computation and defined a Turing machine, and went on to prove that
the halting problem for Turing machines is undecidable. In 1938,
he obtained his PhD from the Department of Mathematics at Princeton University.
During the Second World War, Turing worked for the Government Code and Cypher School
at Bletchley Park, Britain's codebreaking centre that produced Ultra intelligence.
For a time he led Hut 8, the section that was responsible for German naval cryptanalysis.
Here, he devised a number of techniques for speeding the breaking of German ciphers,
including improvements to the pre-war Polish bomba method,
an electromechanical machine that could find settings for the Enigma machine. Turing played a crucial
role in cracking intercepted coded messages that enabled the Allies to defeat the
Axis powers in many crucial engagements, including the Battle of the Atlantic.[11][12]
```

```
myFile1 = open('/content/drive/MyDrive/python/test.csv','r')
if myFile1:
   print('file opened succcesfully\n')
   content = myFile1.read()
   print(content)
myFile1.close()
```

```
file opened succcesfully

c1,c2
1,a
2,b
3,c
4,d
5,e
```

```
# with keyword ensures that the file is closed after performing operations like
set of statements to be executed

with open('/content/drive/MyDrive/python/test.csv','r') as file3:
    content = file3.read()
    print(content)
```

```
c1,c2
1,a
2,b
3,c
4,d
5,e
```

```
myFile4 = open('/content/drive/MyDrive/python/empty.txt', 'w')
myFile4.write('hello garbage')
myFile4.close()
with open('/content/drive/MyDrive/python/empty.txt','r') as file5:
     content = file5.read()
     print(content)
```

```
hello garbage
```

Q write a python program to take input from user in form of paragraph, and this input is stored into a text file. initially the text file is empty, after that append your own data and display the record to the user.

```
myUserInput = input('enter user input to save in file ')
myInput = '\nYour own data.'

with open('/content/drive/MyDrive/Content/Python/test.csv','w') as fileUser:
  fileUser.write(myUserInput)
with open('/content/drive/MyDrive/Content/Python/test.csv','a') as fileUser1:
  fileUser1.write(myInput)
with open('/content/drive/MyDrive/Content/Python/test.csv','r') as fileUser2:
  content = fileUser2.read()
  print(content)
```

```
enter user input to save in file hello there
hello there
Your own data.
```

SIGNATURE: _____

PRACTICAL – VI

Date: 16-09-2023

AIM: Understanding functions

```
#Non-Parametric
def hello():
  print("hello i am function")
hello()
```

```
hello i am function
```

```
#Parametric
def printer(fname,lname):
   print("Your first name is ",fname)
   print("Your last name is ",lname)
printer("ri","ca")
```

```
Your first name is  ri
Your last name is  ca
```

```
import numpy as np
def dis():
print("Enter the values x1,x2,y1,y2")
   x1=int(input())
   x2=int(input())
   y1=int(input())
   y2=int(input())
   a=y2-y1
   b=x2-x1
   p=a*a
   w=a**2
   q=b*b
   z=p+q
   print("final answer is ",np.sqrt(z))
dis()
```

```
Enter the values x1,x2,y1,y2
3
4
5
6
final answer is  1.4142135623730951
```

SIGNATURE: _____

PRACTICAL – VII

Date: 09-10-2023

AIM: Understanding Object Oriented Programming

a) SIMPLE CLASS AND OBJECT CREATION

```
class Bike:
    def_init_(self,name,speed,model): #constructor need to initialize
        self.name=name
        self.speed = speed
        self.model = model
    def style(self,design):
        return f'style of the bike is,{design}'
    def tyre(self,name,model):
        return 'The tyre of the {} of {} is so good'.format(name,model)
        return 'The tyre of the {} of {} is so good'.format(self.name,self.model)
#we use self along with when from object
bk=Bike('Pulsar',120,100)
print(bk.style('good'))
print(bk.tyre('pul',130))
```

```
style of the bike is,good
The tyre of the pul of 130 is so good
```

```
class Parrot:
# instance attributes
    def_init_(self, name, age):
        self.name = name
        self.age = age
# instance method
    def sing(self, song):
        return "{} sings {}".format(self.name, song)
    def dance(self):
        return "{} is now dancing".format(self.name)
# instantiate the object
blu = Parrot("Blu", 10)
# call our instance methods
print(blu.sing("'Happy'"))
print(blu.dance())
```

```
Blu sings 'Happy'
Blu is now dancing
```

SIGNATURE: _____

PROPERTIES OF OBJECT ORIENTED PROGRAMMING

a) INHERITANCE 10-10-2023

SINGLE LEVEL:

```
class parent:
    def_init_(self,name,age,sal):
        self.name = name
        self.age = age
      self.sal  =  sal
    def details(self):
       print("Name of employee is :",self.name)
       print("Age of employee is :",self.age)
       print("Salary of employee is :",self.sal)
e = parent('chandan',22,25000)
e.details()
```

```
Name of employee is : chandan
Age of employee is : 22
Salary of employee is : 25000
```

```
class child(parent):
  def department(self):
    print("Department is DSAI")
d = child('shivam',22,44000)
d.department()
```

```
Department is DSAI
```

```
d.details()
```

```
Name of employee is : shivam
Age of employee is : 22
Salary of employee is : 44000
```

MULTIPLE INHERITANCE:

```
# multiple inheritance example
class Dad: # first parent class
    def func1(self):
      print("Hello Parent1")
class Mom:                      # second parent class
    def func2(self):
      print("Hello Parent2")
class Child(parent1, parent2)     # child class
    def func3(self):                    # we include the parent classes
      print("Hello Child")        # as an argument comma separated
test = Child()          # object created
test.func1()          # parent1 method called via child
test.func2()            # parent2 method called via child instead of parent3
test.func3()            # child method called
# to find the order of classes visited by the child class, we use_mro_on the
child class #print(child.__mro__)
```

```
Hello Parent1
Hello Parent2
Hello Child
```

MULTI-LEVEL INHERITANCE:

```
# multi-level inheritance example
class grandparent:                    # first level
    def func1(self):
       print("Hello Grandparent")
class parent(grandparent):            # second level
    def func2(self):
       print("Hello Parent")
class child(parent):                  # third level
    def func3(self):
       print("Hello Child")
# Driver Code
test = child()                        # object created
test.func1()                          # 3rd level calls 1st level
test.func2()                          # 3rd level calls 2nd level
test.func3()                          # 3rd level calls 3rd level
```

```
Hello Grandparent
Hello Parent
Hello Child
```

HIERARCHICAL INHERITANCE:

```
# hierarchical inheritance example
class parent:                          # parent class
   def func1(self):
     print(""Hello Parent”")
class child1(parent):                  # first child class
   def func2(self):
     print(""Hello Child1”")
class child2(parent):                  # second child class
   def func3(self):
     print(""Hello Child2”")
# Driver Code
test1 = child1()                        # objects created
test2 = child2()
test1.func1()                          # child1 calling parent method
test1.func2()                          # child1 calling its own method
test2.func1()                          # child2 calling parent method
test2.func3()                          # child2 calling its own method
```

```
“Hello Parent”
“Hello Child1”
“Hello Parent”
“Hello Child2”
```

SIGNATURE: _____

a) ENCAPSULATION

10-10-2023

PUBLIC MEMBER

```
class Employee:
# constructor
     def __init__(self, name, salary):
# public data members
         self.name = name
         self.salary = salary
# public instance methods
     def show(self):
# accessing public data member
         print("Name: ", self.name, 'Salary:', self.salary)
# creating object of a class
emp = Employee('Jason', 10000)
# accessing public data members
print("Name: ", emp.name, 'Salary:', emp.salary)
# calling public method of the class
emp.show()
```

```
Name:  Jason Salary: 10000
Name:  Jason Salary: 10000
```

PRIVATE MEMBER

```
class Employee:
# constructor
     def __init__(self, name, salary):
# public data member
         self.name = name
# private member
         self.__salary = salary
# creating object of a class
emp = Employee('Jessa', 10000)
# accessing private data members
print('Salary:', emp.__salary)
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-14-6545f17ab6db> in <module>
     11
     12 # accessing private data members
---> 13 print('Salary:', emp.__salary)

AttributeError: 'Employee' object has no attribute '__salary'

 SEARCH STACK OVERFLOW
```

PROTECTED MEMBER

```
# base class
class Company:
     def __init__(self):
# Protected member
        self._project = "NLP"
# child class
class Employee(Company):
     def_init_(self, name):
         self.name = name
         Company.__init__(self)
     def show(self):
         print("Employee name :", self.name)
# Accessing protected member in child class
         print("Working on project :", self._project)
c = Employee("Jessa")
c.show()
# Direct access protected data member
print('Project:', c._project)
```

```
Employee name : Jessa
Working on project : NLP
Project: NLP
```

NAME MANGLING TO ACCESS PRIVATE MEMBERS

```
class Employee:
# constructor
    def __init__(self, name, salary):
# public data member
       self.name = name
# private member
       self.__salary = salary
# creating object of a class
emp = Employee('Jessa', 10000)
print('Name:', emp.name)
# direct access to private member using name mangling
print('Salary:', emp._Employee__salary)
```

```
Name: Jessa
Salary: 10000
```

```
# Example: Access Private member outside of a class using an instance method
class Employee: # constructor
def_init_(self, name, salary):# public data member
self.name = name # public data member
self._salary = salary  # private member
def show(self): # public instance methods
# private members are accessible from a class
print("Name: ", self.name, 'Salary:', self.__salary)
# creating object of a class
emp = Employee('Jasica', 20000)
# calling public method of the class
emp.show()
```

```
Name:  Jasica Salary: 20000
```

SIGNATURE: _____

POLYMORPHISM 10-10-2023

```
"""Polymorphism in Python Example
Let’s try to understand polymorphism and its implementation even better through
this simple example - """
print(10+256) # addition
print("A"+"DSAI") # Concatination

class Bird:
     def intro(self):
        print("There are many types of birds.")
     def fly(self):
        print("Most of the birds can fly but some cannot.")
class sparrow(Bird):
     def                fly(self):
        print("Sparrows can fly.")
class ostrich(Bird):
     def fly(self):
       print("Ostriches cannot fly.")
obj_bird = Bird()
obj_spr = sparrow()
obj_ost = ostrich()
obj_bird.intro()
```

```
There are many types of birds.
```

```
obj_bird.fly()
```

```
 Most of the birds can fly but some cannot.
```

```
obj_spr.intro()
obj_spr.fly()
```

```
There are many types of birds.
Sparrows can fly.
```

```
obj_ost.intro()
obj_ost.fly()
```

```
There are many types of birds.
Ostriches cannot fly.
```

SIGNATURE: _____

PRACTICAL – VIII

Date: 10-10-2023

AIM: Understanding Exception Handling

```
try:
  numerator = int(input())
  denominator = int(input())
  print(numerator / denominator)
except:
  print("Cannot divisible by zero")
print("Program ends here!")
```

```
5
0
Cannot divisible by zero
Program ends here!
```

```
try:
   numerator = int(input())
   denominator = int(input())
   print(numerator / denominator)
except Exception as e:
   print("Cannot divisible by zero",e)
print("Program ends here!")
```

```
4
t
Cannot divisible by zero invalid literal for int() with base 10: 't'
Program ends here!
```

```
try:
   numerator = int(input())
   denominator = int(input())
   print(numerator / denominator)
   my_list = [1,2,3]
   i = int(input("Enter index : "))
   print(my_list[i])
except ZeroDivisionError:
   print("Cannot divisible by zero")
except IndexError:
   print("you are calling index number which does not exisit!!!")
print("Program ends here!")
```

```
4
4
1.0
Enter index : 4
you are calling index number which does not exisit!!!
Program ends here!
```

```python
try:
  print(1/0)
except:
  print("Any number cannot devided by zero")
finally:
  print("I am going to print anyways")

"""
finally block is used to execute any program is so what we have error
example : Supppose you writting on text there are some unknown error so we can
write file.close
in finally block!
"""
```

```
Any number cannot devided by zero
I am going to print anyways
'\nfinally block is used to execute any program is so what we have error\n'
```

```python
# Raise an error and stop the program if x is lower than 0:
x = -1
if x < 0:
  raise Exception("Sorry, no numbers below zero")
```

```
---------------------------------------------------------------------------
Exception                                 Traceback (most recent call last)
Input In [43], in <cell line: 4>()
      2 x = -1
      4 if x < 0:
----> 5   raise Exception("Sorry, no numbers below zero")

Exception: Sorry, no numbers below zero

SEARCH STACK OVERFLOW
```

```python
#Raise a TypeError if x is not an integer:
x = "hello"
if not type(x) is int:
  raise TypeError("Only integers are allowed")
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Input In [44], in <cell line: 4>()
      2 x = "hello"
      4 if not type(x) is int:
----> 5   raise TypeError("Only integers are allowed")

TypeError: Only integers are allowed

SEARCH STACK OVERFLOW
```

SIGNATURE: _____

PRACTICAL – IX

Date: 16-10-2023

AIM: Understanding Advance Python Programming

```
import pandas as pd
import requests
import numpy as np
from bs4 import BeautifulSoup
import warnings
warnings.filterwarnings('ignore')
df=pd.DataFrame()
requests.get('https://www.ambitionbox.com/list-of-companies').text
```

```
'<HTML><HEAD>\n<TITLE>Access Denied</TITLE>\n</HEAD><BODY>\n<H1>Access Denied</H1>\n \nYou don\'t have permission to access "http&#58;&#47;&#47;
6;ambitionbox&#46;com&#47;list&#45;of&#45;companies" on this server.<P>\nReference&#32;&#35;18&#46;d683c217&#46;1697447925&#46;13063083\n</BODY>
ML>\n'
```

```
headers={'User-agent':'Mozilla/5.0'}
webpage=requests.get('https://www.ambitionbox.com/list-of
companies',headers=headers)
webpage
```

```
<Response [200]>
```

```
soup=BeautifulSoup(webpage.content,'lxml')
soup
```

```
delete:before{content:"\e91c";color:#5e6b92}.icon-edit:before{content:"\e91f";color:#5e6b92}.icon-dots-
menu:before{content:"\e91b";color:#97a0be}.icon-bold:before{content:"\e920";color:#5e6b92}.icon-italic:before{content:"\e921";color:#5e6b92}.icon-
underline:before{content:"\e922";color:#5e6b92}.icon-link:before{content:"\e923";color:#5e6b92}.icon-ordered-
list:before{content:"\e924";color:#5e6b92}.icon-list:before{content:"\e925";color:#5e6b92}.icon-location:before{content:"\e927";color:#97a0be}.icon-
tag:before{content:"\e936";color:#fff}.icon-copy:before{content:"\e937";color:#fff}.icon-center-align:before{content:"\e961";color:#5e6b92}.icon-
code:before{content:"\e962";color:#5e6b92}.icon-left-align:before{content:"\e963";color:#5e6b92}.icon-info-
i:before{content:"\e964";color:#959595}.icon-Star .path1:before{content:"\e967";color:#fff}.icon-Star .path2:before{content:"\e968";margin-
left:-1em;color:#5e6b92}.icon-helpful:before{content:"\e965";color:#5b6ff2}.icon-helpful-solid:before{content:"\e966";color:#5b6ff2}.icon-helpful-
grey:before{content:"\e969";color:#5e6b92}.icon-comment:before{content:"\e91d";color:#5e6b92}.icon-chat:before{content:"\e96a";color:#fff}.shadow-
elevation-1{box-shadow:0 2px 6px -2px rgba(23,34,96,0.2);border:solid 1px #f5f7fd}.shadow-elevation-2{box-shadow:0 3px 8px 0
rgba(23,34,96,0.2);border:solid 1px #f5f7fd}.shadow-elevation-3{box-shadow:0 9px 16px 0 rgba(23,34,96,0.2);border:solid 1px #f5f7fd}.shadow-
elevation-4{box-shadow:2px 6px 0 rgba(0,106,194,0.2)}.shadow-bottomsheet{box-shadow:0 -4px 8px 0 rgba(23,34,96,0.2)}.shadow-left-curtain{box-
shadow:4px 0 8px 0 rgba(23,34,96,0.2)}.shadow-right-curtain{box-shadow:-4px 0 8px 0 rgba(23,34,96,0.2)}.shadow-section-bottom{box-shadow:inset 0
-1px 0 0 #e4e6ee}.shadow-section-top{box-shadow:inset 0 1px 0 0 #e4e6ee}.shadow-section-right{box-shadow:inset -1px 0 0 0
#e4e6ee}.container,.container-two-col{width:100%;max-width:64rem;margin:0 auto}.container.maxWidthFix,.maxWidthFix.container-two-col{max-
width:inherit;padding:0}@media (max-width: 1024px){.container,.container-two-col{margin:0;padding:0 1rem}}.container-two-col{display:flex}@media
```

```
soup.prettify()
```

```
'<!DOCTYPE html>\n<html data-n-head="%7B%22lang%22:%7B%22ssr%22:%22en%22%7D%7D" data-n-head-ssr="" lang="en">\n <head>\n  <meta charset="utf-8"/>\n
 meta content="width=device-width,initial-scale=1,minimum-scale=1" name="viewport"/>\n  <meta content="IE=edge" http-equiv="X-UA-Compatible"/>\n  <!--
 <script src="https://www.googletagmanager.com/gtag/js" async></script> -->\n  <script>\n   window.dataLayer=window.dataLayer||[],window.gtag=window.g
 ag||function(){window.dataLayer.push(arguments)},gtag("js",new Date)\n  </script>\n  <title>\n   List of companies in India | AmbitionBox\n  </title>
 \n  <meta content="2023 AmbitionBox" data-n-head="ssr" name="copyright"/>\n  <meta content="1 day" data-n-head="ssr" name="revisit-after"/>\n  <meta
 ontent="AmbitionBox" data-n-head="ssr" name="application-name"/>\n  <meta content="EN" data-n-head="ssr" name="content-language"/>\n  <meta content="
 62822053404-hphug4pkahqljh2tc96g35at47o4isv2.apps.googleusercontent.com" data-n-head="ssr" name="googl...'
```

```
soup.find_all('h2')[0].text
```

```
'\n\t\t\t\t\t\t\t\t\t\t\tTCS\n\t\t\t\t\t\t\t\t\t'
```

```
for i in soup.find_all('h2'):
    print(i.text.strip())
```

```
TCS
Accenture
Cognizant
Wipro
ICICI Bank
HDFC Bank
Infosys
Capgemini
Tech Mahindra
HCLTech
Genpact
Axis Bank
Concentrix Corporation
Amazon
Reliance Jio
IBM
Larsen & Toubro Limited
Reliance Retail
HDB Financial Services
Teleperformance
Companies by  Industry
Companies by  Locations
Companies by  Type
Companies by  Badges
```

```
soup.find_all('span',class_="companyCardWrapper__ratingValues")
```

```
[<span class="companyCardWrapper__ratingValues">Job Security, Work Life Balance, Company Culture</span>,
 <span class="companyCardWrapper__ratingValues">Promotions / Appraisal, Salary &amp; Benefits</span>,
 <span class="companyCardWrapper__ratingValues">Company Culture, Job Security, Skill Development / Learning</span>,
 <span class="companyCardWrapper__ratingValues">Skill Development / Learning, Job Security</span>,
 <span class="companyCardWrapper__ratingValues">Job Security, Skill Development / Learning</span>,
 <span class="companyCardWrapper__ratingValues">Job Security, Skill Development / Learning, Company Culture</span>,
 <span class="companyCardWrapper__ratingValues">Job Security, Skill Development / Learning</span>,
 <span class="companyCardWrapper__ratingValues">Job Security, Company Culture, Skill Development / Learning</span>,
 <span class="companyCardWrapper__ratingValues">Job Security, Work Life Balance, Company Culture</span>,
 <span class="companyCardWrapper__ratingValues">Job Security</span>,
 <span class="companyCardWrapper__ratingValues">Promotions / Appraisal</span>,
 <span class="companyCardWrapper__ratingValues">Job Security, Skill Development / Learning, Work Life Balance</span>,
 <span class="companyCardWrapper__ratingValues">Job Security</span>,
 <span class="companyCardWrapper__ratingValues">Job Security, Company Culture</span>,
 <span class="companyCardWrapper__ratingValues">Company Culture, Salary &amp; Benefits, Work Life Balance</span>,
 <span class="companyCardWrapper__ratingValues">Job Security, Skill Development / Learning</span>,
 <span class="companyCardWrapper__ratingValues">Job Security, Work Life Balance, Skill Development / Learning</span>,
 <span class="companyCardWrapper__ratingValues">Job Security, Skill Development / Learning</span>,
 <span class="companyCardWrapper__ratingValues">Skill Development / Learning, Job Security</span>,
 <span class="companyCardWrapper__ratingValues">Skill Development / Learning, Job Security, Company Culture</span>]
```

```
companies=[]
for i in soup.find_all('h2'):
      companies.append(i.text.strip())
companies
perks=[]
for j in soup.find_all('span',class_="companyCardWrapper_ratingValues"):
                        perks.append(j.text.strip())
perks
```

```
['Job Security, Work Life Balance, Company Culture',
 'Promotions / Appraisal, Salary & Benefits',
 'Company Culture, Job Security, Skill Development / Learning',
 'Skill Development / Learning, Job Security',
 'Job Security, Skill Development / Learning',
 'Job Security, Skill Development / Learning, Company Culture',
 'Job Security, Skill Development / Learning',
 'Job Security, Company Culture, Skill Development / Learning',
 'Job Security, Work Life Balance, Company Culture',
 'Job Security',
 'Promotions / Appraisal',
 'Job Security, Skill Development / Learning, Work Life Balance',
 'Job Security',
 'Job Security, Company Culture',
 'Company Culture, Salary & Benefits, Work Life Balance',
 'Job Security, Skill Development / Learning',
 'Job Security, Work Life Balance, Skill Development / Learning',
 'Job Security, Skill Development / Learning',
 'Skill Development / Learning, Job Security',
 'Skill Development / Learning, Job Security, Company Culture']
```

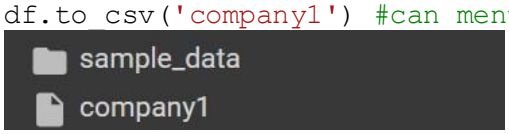
```
data={'company_name': companies,'perk':perks}
data
```

```
    'Tech Mahindra',
    'HCLTech',
    'Genpact',
    'Axis Bank',
    'Concentrix Corporation',
    'Amazon',
    'Reliance Jio',
    'IBM',
    'Larsen & Toubro Limited',
    'Reliance Retail',
    'HDB Financial Services',
    'Teleperformance',
    'Companies by  Industry',
    'Companies by  Locations',
    'Companies by  Type',
    'Companies by  Badges'],
   'perk': ['Job Security, Work Life Balance, Company Culture',
    'Promotions / Appraisal, Salary & Benefits',
    'Company Culture, Job Security, Skill Development / Learning',
    'Skill Development / Learning, Job Security',
    'Job Security, Skill Development / Learning',
    'Job Security, Skill Development / Learning, Company Culture',
    'Job Security, Skill Development / Learning',
    'Job Security, Company Culture, Skill Development / Learning',
    'Job Security, Work Life Balance, Company Culture',
    'Job Security',
    'Promotions / Appraisal',
    'Job Security, Skill Development / Learning, Work Life Balance',
    'Job Security',
    'Job Security, Company Culture',
    'Company Culture, Salary & Benefits, Work Life Balance',
```

```
newcompany=companies[0:20]
data={'company_name': newcompany,'perk':perks}
df = pd.DataFrame(data)
df
```

index	company_name	perk
0	TCS	Job Security, Work Life Balance, Company Culture
1	Accenture	Promotions / Appraisal, Salary & Benefits
2	Cognizant	Company Culture, Job Security, Skill Development / Learning
3	Wipro	Skill Development / Learning, Job Security
4	ICICI Bank	Job Security, Skill Development / Learning
5	HDFC Bank	Job Security, Skill Development / Learning, Company Culture
6	Infosys	Job Security, Skill Development / Learning
7	Capgemini	Job Security, Company Culture, Skill Development / Learning
8	Tech Mahindra	Job Security, Work Life Balance, Company Culture
9	HCLTech	Job Security
10	Genpact	Promotions / Appraisal
11	Axis Bank	Job Security, Skill Development / Learning, Work Life Balance
12	Concentrix Corporation	Job Security
13	Amazon	Job Security, Company Culture
14	Reliance Jio	Company Culture, Salary & Benefits, Work Life Balance
15	IBM	Job Security, Skill Development / Learning
16	Larsen & Toubro Limited	Job Security, Work Life Balance, Skill Development / Learning
17	Reliance Retail	Job Security, Skill Development / Learning
18	HDB Financial Services	Skill Development / Learning, Job Security
19	Teleperformance	Skill Development / Learning, Job Security, Company Culture

```
df.to_csv('company1') #can mention particular address/path
```



SIGNATURE: _____

PRACTICAL – X

Date: 25-09-2023

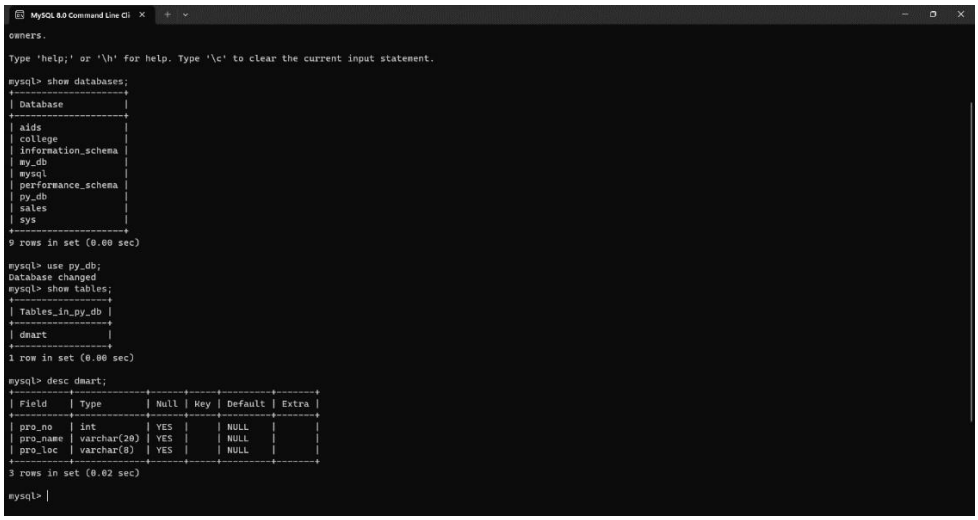
AIM: Perform CRUD Operation using Streamlit

PRE-REQUISTIC:

Visual Studio and along with it desired library for performing the same such as "streamlit","mysqlconnector"

Establishing Connection with MySQL and creating table

```
import mysql.connector as mycon
db=mycon.connect(host='localhost',user='root',password='root',database='py_db')
print(db)
db_curr=db.cursor()
db_curr.execute('create table dmart(pro_no int,pro_name varchar(20),pro_loc
varchar(8))')
```

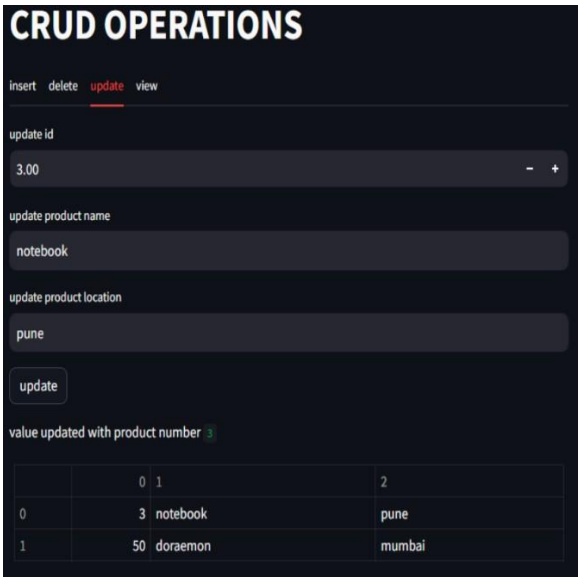


Creating GUI with the help of streamlit library in Python and performing CRUD Operation

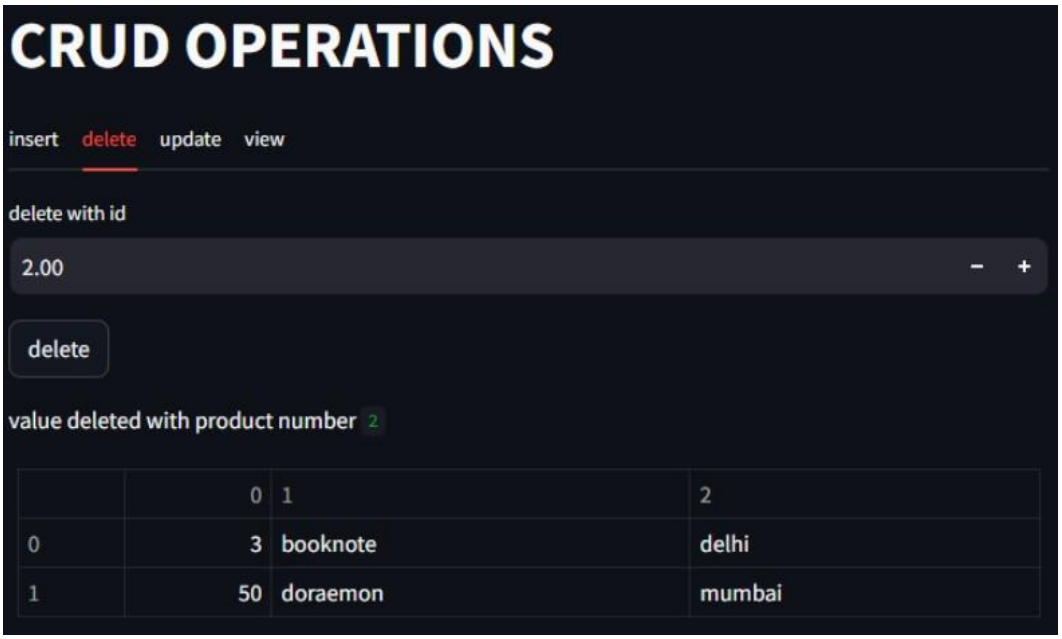
```
import streamlit as st
#Connection establish
import mysql.connector as mycon
db=mycon.connect(host='localhost',user='root',password='root',database=
'py_db')
db_curr=db.cursor()
#GUI
st.title('CRUD OPERATION')
tab1, tab2, tab3 = st.tabs(['insert','update','delete'])
with tab1:
    num=st.number_input("Enter Product No:")
    name=st.text_input("Enter Product Name:")
    loc=st.text_input("Enter Product Location:")
    if st.button("submit"):
        st.write("added")
        st.write(num,name,loc) #to print in streamlit
        sql ="insert into dmart(pro_no,pro_name,pro_loc) values
(%s,%s,%s)"
        val = (num,name,loc)
        db_curr.execute(sql,val)
        db.commit()
        db_curr.execute('Select * from dmart')
```

```
        st.table(db_curr.fetchall())
```


```
with tab2:
    number=int(st.number_input("Updt Product No:"))
    names=st.text_input("Updt Product Name:")
    locations=st.text_input("Updt Product Location:")
    if st.button("update"):
        st.write("added")
        st.write(number,names) #to print in streamlit
        sql ="update dmart set pro_name = %s where pro_no = %s"
        val = (names,number)
        db_curr.execute(sql,val)
        db.commit()
        db_curr.execute('Select * from dmart')
        st.table(db_curr.fetchall())
```



```
with tab3:
    numbers=int(st.number_input("dlt Product No:"))
    if st.button("delete"):
        st.write("deleted")
        st.write(numbers) #to print in streamlit
        sql =f"delete from dmart where pro_no = {numbers}"
        #val = (numbers)
        db_curr.execute(sql)
        db.commit()
        db_curr.execute('Select * from dmart')
        st.table(db_curr.fetchall())
```



SIGNATURE: _____

PRACTICAL – XI

Date: 18-09-2023

AIM: Understanding module Pandas

```
import pandas as pd
import seaborn as sns
import numpy as np
import matplotlib.pyplot as plt
employee=pd.read_csv(r"C:\Users\User20\Downloads\employees.csv",index_col=0)
fortune=pd.read_csv(r"C:\Users\User20\Downloads\Fortune_1000.csv",index_col=0)
mark=pd.read_csv(r"C:\Users\User20\Downloads\marks.csv",index_col=0)
mark.filter(['Name','Bio','Chem']) #create a subset
```

	Name	Bio	Chem
0	Fahad	29	33
1	Akash	56	66
2	Rohi	78	75
3	lalita	95	83
4	Sahhil	99	93

```
fortune.head()
```

company	rank	rank_change	revenue	profit	num. of employees	sector	city	state	newcomer	ceo_founder	ceo_woman	profitable	prev_rank
Walmart	1	0.0	572754.0	13673.0	2300000.0	Retailing	Bentonville	AR	no	no	no	yes	1.0
Amazon	2	0.0	469822.0	33364.0	1608000.0	Retailing	Seattle	WA	no	no	no	yes	2.0
Apple	3	0.0	365817.0	94680.0	154000.0	Technology	Cupertino	CA	no	no	no	yes	3.0
VS Health	4	0.0	292111.0	7910.0	258000.0	Health Care	Woonsocket	RI	no	no	yes	yes	4.0
nitedHealth Group	5	0.0	287597.0	17285.0	350000.0	Health Care	Minnetonka	MN	no	no	no	yes	5.0

```
fortune.shape
```

```
(1000, 17)
```

```
fortune.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 1000 entries, Walmart to DocuSign
Data columns (total 17 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   rank               1000 non-null   int64
 1   rank_change        1000 non-null   float64
 2   revenue            1000 non-null   float64
 3   profit             997 non-null    float64
 4   num. of employees  999 non-null    float64
 5   sector             1000 non-null   object
 6   city               1000 non-null   object
 7   state              1000 non-null   object
 8   newcomer           1000 non-null   object
 9   ceo_founder        1000 non-null   object
 10  ceo_woman          1000 non-null   object
 11  profitable         1000 non-null   object
 12  prev_rank          1000 non-null   object
 13  CEO                1000 non-null   object
 14  Website            1000 non-null   object
 15  Ticker             951 non-null    object
 16  Market Cap         969 non-null    object
dtypes: float64(4), int64(1), object(12)
memory usage: 140.6+ KB
```

```
sec=fortune.groupby('sector') #groupby only for categorical value
sec.size() #for aplha bases
```

```
sector
Aerospace & Defense              17
Apparel                          16
Business Services                52
Chemicals                        29
Energy                          100
Engineering & Construction       32
Financials                      166
Food & Drug Stores                9
Food, Beverages & Tobacco        34
Health Care                      77
Hotels, Restaurants & Leisure    28
Household Products               23
Industrials                      50
Materials                        46
Media                            28
Motor Vehicles & Parts           20
Retailing                        77
Technology                      121
Telecommunications                9
Transportation                   35
Wholesalers                      31
dtype: int64
```

```
sec.size().sort_values() #sort by numerical values
```

```
sector
Telecommunications               9
Food & Drug Stores               9
Apparel                         16
Aerospace & Defense             17
Motor Vehicles & Parts          20
Household Products              23
Media                           28
Hotels, Restaurants & Leisure   28
Chemicals                       29
Wholesalers                     31
Engineering & Construction      32
Food, Beverages & Tobacco       34
Transportation                  35
Materials                       46
Industrials                     50
Business Services               52
Health Care                     77
Retailing                       77
Energy                         100
Technology                     121
Financials                     166
dtype: int64
```

```
sec.first() #which one with it got encourted with the sector first
sec.last() #which one with it got encourted with the sector last

df1=pd.read_excel(r"C:\Users\User20\Downloads\df1.xlsx",index_col=0)
df1
```

	FirstName	LastName
ID		
1001	Shaikh	Fahad
1002	Yadav	Anil
1003	Waghmore	Shweta
1004	Kalaniya	Zaid

```
df1=pd.read_excel(r"C:\Users\User20\Downloads\df1.xlsx",index_col=0,sheet_name='df
1')
df2=pd.read_excel(r"C:\Users\User20\Downloads\df1.xlsx",index_col=0,sheet_name='df
2')
df1
```

	FirstName	LastName
ID		
1001	Shaikh	Fahad
1002	Yadav	Anil
1003	Waghmore	Shweta
1004	Kalaniya	Zaid

```
df2
```

	FirstName	LastName
ID		
1001	Shaikh	Fahad
1002	Yadav	Anil
1006	Desoza	Zen
1007	Vaiz	Bob
1008	Magic	Mike

```
innerjoin=df1.merge(df2,how='inner')
print(innerjoin)
```

	FirstName	LastName
0	Shaikh	Fahad
1	Yadav	Anil

```
innerjoin=pd.merge(df1,df2,on='ID',how='inner') #leftjoin
innerjoin
```

	FirstName_x	LastName_x	FirstName_y	LastName_y
ID				
1001	Shaikh	Fahad	Shaikh	Fahad
1002	Yadav	Anil	Yadav	Anil

```
leftjoin=pd.merge(df1,df2,on='ID',how='left')
leftjoin
```

	FirstName_x	LastName_x	FirstName_y	LastName_y
ID				
1001	Shaikh	Fahad	Shaikh	Fahad
1002	Yadav	Anil	Yadav	Anil
1003	Waghmore	Shweta	NaN	NaN
1004	Kalaniya	Zaid	NaN	NaN

```
rightjoin=pd.merge(df1,df2,on='ID',how='right')
rightjoin
```

	FirstName_x	LastName_x	FirstName_y	LastName_y
ID				
1001	Shaikh	Fahad	Shaikh	Fahad
1002	Yadav	Anil	Yadav	Anil
1006	NaN	NaN	Desoza	Zen
1007	NaN	NaN	Vaiz	Bob
1008	NaN	NaN	Magic	Mike

```
outerjoin=df1.merge(df2,how='outer')
print(outerjoin)
```

	FirstName	LastName
0	Shaikh	Fahad
1	Yadav	Anil
2	Waghmore	Shweta
3	Kalaniya	Zaid
4	Desoza	Zen
5	Vaiz	Bob
6	Magic	Mike

```
#pivot table
fortune.pivot_table(index='sector',columns='city',aggfunc='count')
```

	CEO										...	state			
city	Abbott Park	Akron	Allentown	Ames	Andover	Ankeny	Ann Arbor	Anoka	Arden Hills	Arlington	...	Wilmington	Windsor	Winona	Winston-Salem
sector															
Aerospace & Defense	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN
Apparel	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	1.0
Business Services	NaN	NaN	NaN	NaN	NaN	NaN	NaN	1.0	NaN	1.0	...	NaN	NaN	NaN	NaN
Chemicals	NaN	NaN	1.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	3.0	NaN	NaN	NaN
Energy	NaN	1.0	2.0	1.0	NaN	NaN	NaN	NaN	NaN	1.0	...	NaN	NaN	NaN	NaN
Engineering & Construction	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	1.0	...	NaN	NaN	NaN	NaN
Financials	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	1.0	...	2.0	NaN	NaN	NaN
Food & Drug Stores	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN
Food, Beverages & Tobacco	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	1.0	NaN	...	NaN	NaN	NaN	NaN
Health Care	1.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	2.0	NaN	NaN	NaN
Hotels, Restaurants & Leisure	NaN	NaN	NaN	NaN	NaN	NaN	1.0	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN

SIGNATURE: _____

PRACTICAL – XII

Date: 18-09-2023

AIM: Understanding module Numpy

```
import numpy as np
arr=np.array([1,2,3,4,5])
print(arr)
print(type(arr))
```

```
[1 2 3 4 5]
<class 'numpy.ndarray'>
```

```
arr=np.array(12)
print(arr)
print(type(arr))
np.ndim(arr)
```

```
12
<class 'numpy.ndarray'>
0
```

```
arr=np.array([1,3])
print(arr)
np.ndim(arr) #reflects the dimension
```

```
[1 3]
1
```

```
arr=np.array([[1,2],[2,4]])
print(arr)
np.ndim(arr)
```

```
[[1 2]
 [2 4]]
2
```

```
arr=np.array([[[1,2]],[[2,30]],[[3,4]]])
print(arr)
np.ndim(arr)
arr.shape
```

```
[[[ 1  2]]

 [[ 2 30]]

 [[ 3  4]]]
(3, 1, 2)
```

```
# basic array characteristics
import numpy as np
# Creating array object
arr = np.array( [[ 1, 2, 3],[ 4, 2, 5]] )
print("Array is of type: ", type(arr))
print("No. of dimensions: ", arr.ndim)
print("Shape of array: ", arr.shape) # Printing shape of array
print("Size of array: ", arr.size) # Printing size
print("Array stores elements of type: ", arr.dtype)
```

```
Array is of type:  <class 'numpy.ndarray'>
No. of dimensions:  2
Shape of array:  (2, 3)
Size of array:  6
Array stores elements of type:  int64
```

```
arr = np.array([1, 2, 3, 4], ndmin=5)
print(arr)
print('number of dimensions :', arr.ndim)
```

```
[[[[[1 2 3 4]]]]]
number of dimensions : 5
```

```
array=np.arange(20)
array
```

```
array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16,
       17, 18, 19])
```

```
array=np.arange(20).reshape(4,5)
array
```

```
array([[ 0,  1,  2,  3,  4],
       [ 5,  6,  7,  8,  9],
       [10, 11, 12, 13, 14],
       [15, 16, 17, 18, 19]])
```

```
arr = np.array([1, 2, 3, 4])
print(arr[1])
```

```
2
```

```
arr = np.array([[1,2,3,4,5], [6,7,8,9,10]])
print('2nd element on 1st row: ', arr[0, 1])
```

```
2nd element on 1st row:  2
```

```
arr = np.array([[[1, 2, 3], [4, 5, 6]], [[7, 8, 9], [10, 11, 12]]])
print(arr[0, 1, 2])
```

```
6
```

```
arr = np.array([1, 2, 3, 4, 5, 4, 4])
x = np.where(arr == 4)
print(x)
```

```
(array([3, 5, 6]),)
```

```
arr = np.array([3, 2, 0, 1])
print(np.sort(arr))
```

```
[0 1 2 3]
```

SIGNATURE: _____

PRACTICAL – XIII

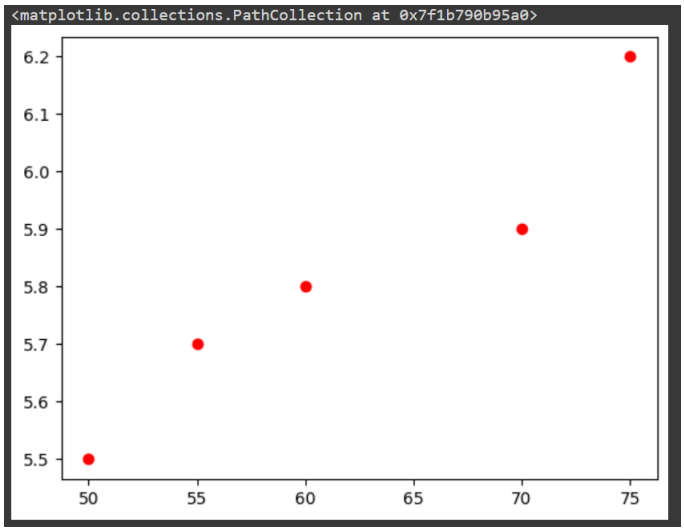
Date: 27-09-2023

AIM: Understanding Visualization

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
data = {'Sr.No': [1,2,3,4,5],'Gender': ['m', 'f',
'f','m','m'],'Weight':[50,55,60,70,75],'Height':[5.5,5.7,5.8,5.9,6.2]}
df = pd.DataFrame(data)
df
```

	Sr.No	Gender	Weight	Height
0	1	m	50	5.5
1	2	f	55	5.7
2	3	f	60	5.8
3	4	m	70	5.9
4	5	m	75	6.2

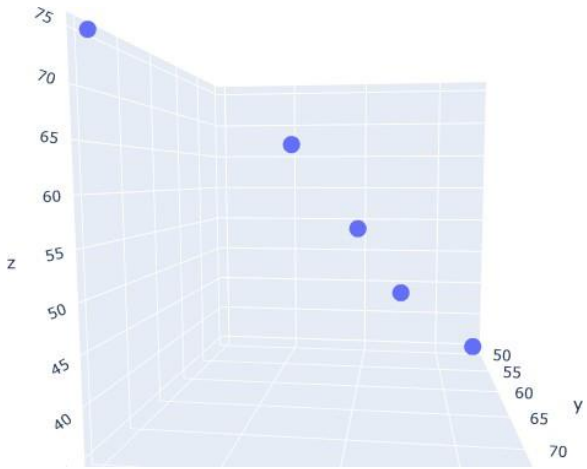
```
from matplotlib import pyplot as plt #no need as we have imported above
plt.scatter(df.iloc[:,-2],df['Height'],c='r') #we can use color='red' #iloc is we
used for 'x' #second is for y axis
```




```
df['BMI']=[35,45,55,65,75] #to add new coloumn without any function such as
append()
df
```

	Sr.No	Gender	Weight	Height	BMI
0	1	m	50	5.5	35
1	2	f	55	5.7	45
2	3	f	60	5.8	55
3	4	m	70	5.9	65
4	5	m	75	6.2	75

```
import plotly.express as px #3D matplot lib
px.scatter_3d(x=df['Height'],y=df['Weight'],z=df['BMI'],)
```



SIGNATURE: _____