**PRACTICAL – 1 (1.1)**

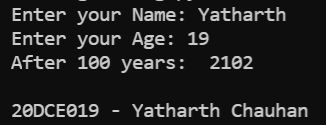
**AIM:**

|  |
| --- |
| **Create a program that asks the user to enter their name and their age. Printout a message addressed to them that tells them the year that they will turn 100 years old.** |

**PROGRAM:**

|  |
| --- |
| import datetime  name = input("Enter your Name: ")  age = int(input("Enter your Age: "))  date = datetime.datetime.now()  temp = 100 - age  update = temp + date.year  print(update)  print("\n20DCE019 - Yatharth Chauhan") |

**OUTPUT:**

****

**CONCLUSION:** In this practical we learned about the usage of datetime library, input and print function.

**PRACTICAL – 1 (1.2)**

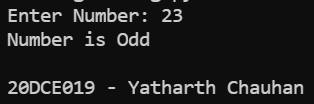
**AIM:**

|  |
| --- |
| **Write a program to ask the user for a number. Depending on whether the number is even or odd, print out an appropriate message to the user. Hint: how does an even / odd number react differently when divided by 2?** |

**PROGRAM:**

|  |
| --- |
| n = int(input("Enter Number: "))  if(n % 2 == 0):  print("Number is Even")  else:  print("Number is Odd")  print("\n20DCE019 - Yatharth Chauhan") |

**OUTPUT:**

****

**CONCLUSION:** In this practical we learned about the usage of ‘if’ statement.

**PRACTICAL – 2 (2.1)**

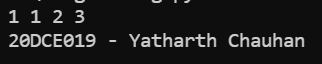
**AIM:**

|  |  |
| --- | --- |
| |  | | --- | | **Write a program as mentioned below:**  **Take a list, a = [1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89], and write a program that print out all the elements of the list that are less than 5.** | |

**PROGRAM:**

|  |
| --- |
| list = [1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89]  for i in range(len(list)):  if(list[i] < 5):    print(list[i], end=" ")  print("\n20DCE019 - Yatharth Chauhan") |

**OUTPUT:**

****

**CONCLUSION:** In this practical we learned about the usage of list and for loop.

**PRACTICAL – 2 (2.2)**

**AIM:**

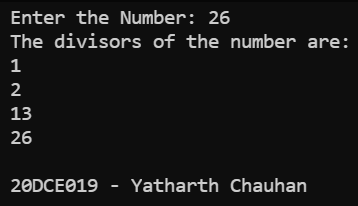
|  |
| --- |
| **Create a program that asks the user for a number and then prints out a list of all the divisors of that number. (If you don’t know what a divisor is, it is a number that divides evenly into another number. For example, 13 is a divisor of 26 because 26 / 13 has no remainder.)** |

|  |
| --- |
|  |

**PROGRAM:**

|  |
| --- |
| n = int(input("Enter the Number: "))  print("The divisors of the number are:")  for i in range(1, n + 1):  if(n % i == 0):  print(i)  print("\n20DCE019 - Yatharth Chauhan") |

**OUTPUT:**

****

**CONCLUSION:** In this practical we learned about the usage of ‘if’, ’for’ loop and the usage of lists.

**PRACTICAL – 3 (3.1)**

**AIM:**

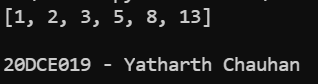
**Take two lists, a = [1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89], b = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13] and write a program that returns a list that contains only the elements that are common between the lists (without duplicates). Make sure your program works on two lists of different sizes.**

|  |
| --- |
|  |

**PROGRAM:**

|  |
| --- |
| a = [1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89]  b = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13]  common = []  a = set(a)  b = set(b)  for i in a:  if i in b:  common.append(i)  print(common)  print("\n20DCE019 - Yatharth Chauhan") |

**OUTPUT:**

****

**CONCLUSION:** In this practical we learned about the usage of set.

**PRACTICAL – (3.2)**

**AIM:**

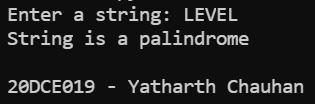
**Write a program by asking the user for a string and print out whether this string is a palindrome or not. (A palindrome is a string that reads the same forwards and backwards.)**

|  |
| --- |
|  |

**PROGRAM:**

|  |
| --- |
| string = input(("Enter a string: "))  if(string == string[::-1]):  print("String is a palindrome")  else:  print("String is not a palindrome")  print("\n20DCE019 - Yatharth Chauhan") |

**OUTPUT:**

****

**CONCLUSION:** In this practical we learned about the palindrome.

**PRACTICAL – 4(4.1)**

**AIM:**

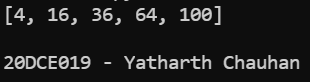
**Write one line of Python that takes list a = [1, 4, 9, 16, 25, 36, 49, 64, 81, 100] and makes a new list that has only the even elements of this list in it.**

|  |
| --- |
|  |

**PROGRAM:**

|  |
| --- |
| list = [1, 4, 9, 16, 25, 36, 49, 64, 81, 100]  ans = [i for i in list if i % 2 == 0]  print(ans)  print("\n20DCE019 - Yatharth Chauhan") |

**OUTPUT:**

****

**CONCLUSION:** In this practical we learned the usage of list in python.

**PRACTICAL – 4(4.2)**

**AIM:**

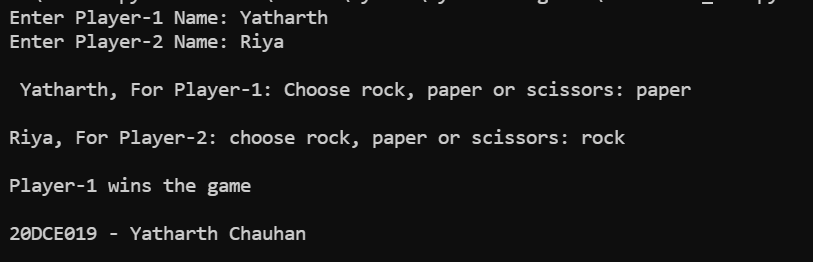
**Write a program to make a two-player Rock-Paper-Scissors game. (Hint: Ask for player plays (using input), compare them, print out a message of congratulations to the winner, and ask if the players want to start a new game) Rules: Rock beats scissors, Scissors beats paper, Paper beats rock**

|  |
| --- |
|  |

**PROGRAM:**

|  |
| --- |
| player1 = input("Enter Player-1 Name: ")  player2 = input("Enter Player-2 Name: ")  p1 = input(  "\n %s, For Player-1: Choose rock, paper or scissors: " % player1)  p2 = input(  "\n %s, For Player-2: choose rock, paper or scissors: s" % player2)  def compare(u1, u2):  if u1 == u2:  return("It's a tie")  elif u1 == 'rock':  if u2 == 'scissors':  return("\nPlayer-1 wins the game")  else:  return("\nPlayer-2 wins the game")  elif u1 == 'scissors':  if u2 == 'paper':  return("\nPlayer-1 win the game")  else:  return("\nPlayer-2 wins the game")  elif u1 == 'paper':  if u2 == 'rock':  return("\nPlayer-1 wins the game")  else:  return("\nPlayer-2 win the game")  else:  return("\nInvalid input")  print(compare(p1, p2))  print("\n20DCE019 - Yatharth Chauhan") |

**OUTPUT:**

****

**CONCLUSION:** In this practical we learned the usage of ‘if’’, ‘else’ and ‘elif ‘statement.

**PRACTICAL – 5(5.1)**

**AIM:**

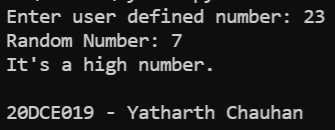
**Write a program to generate a random number between 1 and 9 (including 1 and 9). Ask the user to guess the number, then tell them whether they guessed too low, too high, or exactly right. (Hint: remember to use the user input lessons from the very first practical)**

|  |
| --- |
|  |

**PROGRAM:**

|  |
| --- |
| from random import randrange  a = int(input("Enter user defined number: "))  rd = int(input("Random Number: "))  if(a > rd):  print("It's a high number.")  elif(a < rd):  print("It's a low number.")  else:  print("Exactly Same")  print("\n20DCE019 - Yatharth Chauhan") |

**OUTPUT:**

****

**CONCLUSION:** In this practical we learned about random library file

**PRACTICAL – 5(5.2)**

**AIM:**

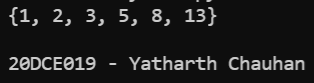
**Write a program to generate a random number between 1 and 9 (including 1 and 9). Ask the user to guess the number, then tell them whether they guessed too low, too high, or exactly right. (Hint: remember to use the user input lessons from the very first practical)**

|  |
| --- |
|  |

**PROGRAM:**

|  |
| --- |
| a = [1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89]  a = (set(a))  b = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13]  b = (set(b))  ans = a & b  print(str(ans))  print("\n20DCE019 - Yatharth Chauhan") |

**OUTPUT:**

****

**CONCLUSION:** In this practical we learned about random library file

**PRACTICAL – 6(6.1)**

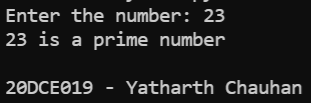
**AIM:**

**Write a program using a function to check whether the number is prime or not. (A prime number is a number that has no divisors.)**

**PROGRAM:**

|  |
| --- |
| num = int(input("Enter the number: "))  if num > 1:  for i in range(2, int(num/2)+1):  if (num % i) == 0:  print(num, "is not a prime number")  break  else:  print(num, "is a prime number")  else:  print(num, "is not a prime number")  print("\n20DCE019 - Yatharth Chauhan") |

**OUTPUT:**

****

**CONCLUSION:** In this practical we learned about the range operator

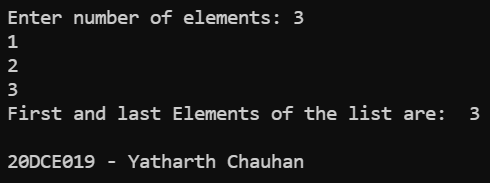
**PRACTICAL – 6(6.2)**

**AIM: Write a program that takes a list of numbers (for example, a = [5, 10, 15, 20, 25]) and makes a new list of only the first and last elements of the given list. For practice, write this code inside a function.**

**PROGRAM:**

|  |
| --- |
| def first\_last(a):  newlist = []  newlist.append(a[0])  newlist.append(a[-1])  print("First and last Elements of the list are: ", a[-1])  return newlist  a = []  num = int(input("Enter number of elements: "))  for i in range(0, num):  element = int(input())  a.append(element)  first\_last(a)  print("\n20DCE019 - Yatharth Chauhan") |

**OUTPUT:**

****

**CONCLUSION:** In this practical we learned about the use of append operator.

**PRACTICAL – 7(7.1)**

**AIM:**

**Write a program that asks the user how many Fibonacci numbers to generate and then generates them. Take 2 1,2 this opportunity to think about how you can use functions. Make sure to ask the user to enter the number of**

**numbers in the sequence to generate. (Hint: The Fibonacci sequence is a sequence of numbers where the next**

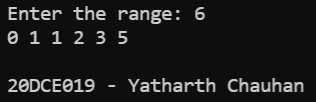
**number in the sequence is the sum of the previous two numbers in the sequence. The sequence looks like this: 1,**

**1, 2, 3, 5, 8, 13, ...)**

**PROGRAM:**

|  |
| --- |
| def FibonacciNum(n):  n1 = 0  n2 = 1  if (n < 1):  return  print(n1, end=" ")  for i in range(1, n):  print(n2, end=" ")  sum = n1 + n2  n1 = n2  n2 = sum  FibonacciNum(int(input("Enter the range: ")))  print("\n\n20DCE019 - Yatharth Chauhan") |

**OUTPUT:**

****

**CONCLUSION:** In this practical we learned about the usage of functions.

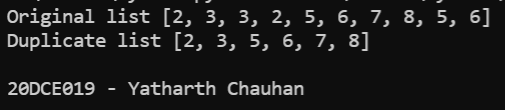
**PRACTICAL – 7(7.2)**

**AIM: Write a program (function!) that takes a list and returns a new list that contains all the elements of the first 1,2 list minus all the duplicates.**

**PROGRAM:**

|  |
| --- |
| def dedupe\_v1(x):  y = []  for i in x:  if i not in y:  y.append(i)  return y  a = [2, 3, 3, 2, 5, 6, 7, 8, 5, 6]  print("Original list", a)  print("Duplicate list", dedupe\_v1(a))  print("\n20DCE019 - Yatharth Chauhan") |

**OUTPUT:**

****

**CONCLUSION:** In this practical we learned about the use of def keyword.

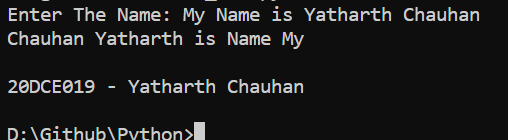
**PRACTICAL – (8.1)**

**AIM: Write a program (using functions!) that asks the user for a long string function. Containing multiple words. Print back to the user the same string, except with the words in backwards order. For example, say I type the string: My name is Michele Then I would see the string: Michele is name My shown back to me.**

**PROGRAM:**

|  |
| --- |
| def reverse(string):  return ' '.join(string.split(' ')[::-1])  print(reverse(input('Enter The Name: ')))  print("\n20DCE019 - Yatharth Chauhan") |

**OUTPUT:**

****

**CONCLUSION:** In this practical, we learned to Reverse a sentence without actually reversing the word.

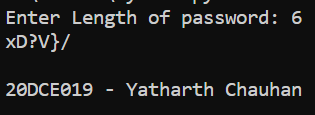
**PRACTICAL – (8.2)**

**AIM: Write a password generator in Python. Be creative with how you generate passwords - strong passwords have a mix of lowercase letters, uppercase letters, numbers, and symbols. The passwords should be random, generating a new password every time the user asks for a new password. Include your run time code in a main method.**

**PROGRAM:**

|  |
| --- |
| import string  import random  a = int(input("Enter Length of password: "))  all = string.ascii\_lowercase+string.ascii\_uppercase + \  string.digits+string.punctuation  def passwordGen(size):  return ''.join(random.choice(all) for \_ in range(size))  print(passwordGen(a))  print("\n20DCE019 - Yatharth Chauhan") |

**OUTPUT:**

****

**CONCLUSION:** In this practical, we learned about How to use Random Library to Generate Password.

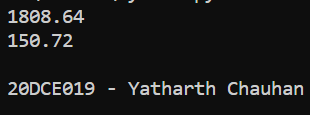
**PRACTICAL – (9.1)**

**AIM: Write a Python class named Circle constructed by a radius and two methods which will compute the area and the perimeter of a circle.**

**PROGRAM:**

|  |
| --- |
| pi = 3.14  class Circle():  def \_\_init\_\_(self, r):  self. radius = r  def area(self):  return self.radius\*\*2\*pi  def peri(self):  return self.radius\*pi\*2  NewCircle = Circle(24)  print(NewCircle.area())  print(NewCircle.peri())  print("\n20DCE019 - Yatharth Chauhan") |

**OUTPUT:**

****

**CONCLUSION:** In this practical, we learned to use and define function in python program.

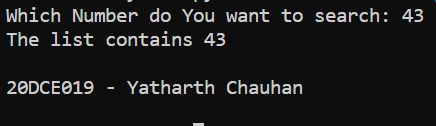
**PRACTICAL – (9.2)**

**AIM: Write a function that takes an ordered list of numbers (a list where the elements are in order from smallest to largest) and another number. The function decides whether or not the given number is inside the list and returns (then prints) an appropriate Boolean.**

**PROGRAM:**

|  |
| --- |
| a = [1, 3, 5, 30, 42, 43, 500]  q = int(input('Which Number do You want to search: '))  flag = 0  while a != []:  mid = int(len(a)/2)  if q == a[mid]:  print('The list contains', q)  flag = 1  break  elif q > a[mid]:  del a[:mid+1]  elif q < a[mid]:  del a[mid:]  if flag == 0:  print('The list does not contain ', q)  print("\n20DCE019 - Yatharth Chauhan") |

**OUTPUT:**

****

**CONCLUSION:** In this Practical, we learned about binary Search execution in python.

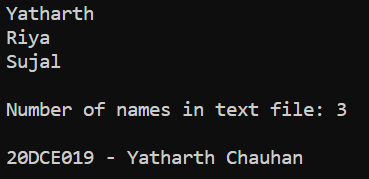
**PRACTICAL – (10.1)**

**AIM: Given a .txt file that has a list of a bunch of names, count how many of each name there are in the file, and print out the results to the screen.**

**PROGRAM:**

|  |
| --- |
| f = open("P\_10.txt", "rt")  data = f.read()  words = data.split()  print("Number of names in text file: ", len(words))  print("\n20DCE019 - Yatharth Chauhan") |

**OUTPUT:**

****

**CONCLUSION:** In this practical, we learned how to count the number of words written in any particular file.

**PRACTICAL – (10.2)**

**AIM: Write a program to implement different Data structures using Python.**

**• Linked List**

**• Stack**

**• Queue**

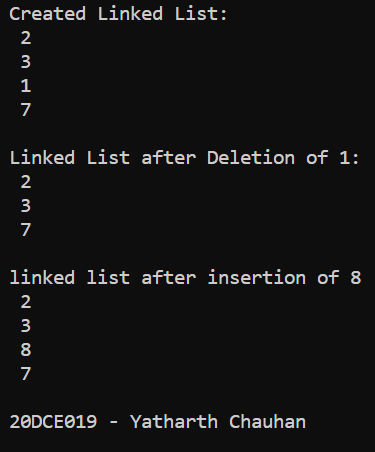
**• Binary Tree**

**PROGRAM:**

1. **Linked List**

|  |
| --- |
| class Node:  def \_\_init\_\_(self, data):  self.data = data  self.next = None  class LinkedList:  def \_\_init\_\_(self):  self.head = None  def push(self, new\_data):  new\_node = Node(new\_data)  new\_node.next = self.head  self.head = new\_node  def deleteNode(self, key):  temp = self.head  if (temp is not None):  if (temp.data == key):  self.head = temp.next  temp = None  return  while(temp is not None):  if temp.data == key:  break  prev = temp  temp = temp.next  if(temp == None):  return  prev.next = temp.next  temp = None  def printList(self):  temp = self.head  while(temp):  print(" %d" % (temp.data)),  temp = temp.next  def insertAfter(self, prev\_node, new\_data):  if prev\_node is None:  print("The given previous node must inLinkedList.")  return  new\_node = Node(new\_data)  new\_node.next = prev\_node.next  prev\_node.next = new\_node  llist = LinkedList()  llist.push(7)  llist.push(1)  llist.push(3)  llist.push(2)  print("Created Linked List: ")  llist.printList()  llist.deleteNode(1)  print("\nLinked List after Deletion of 1:")  llist.printList()  print("\nlinked list after insertion of 8")  llist.insertAfter(llist.head.next, 8)  llist.printList()  print("\n20DCE019 - Yatharth Chauhan") |

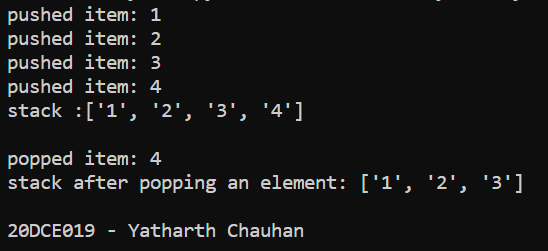
**OUTPUT:**

****

1. **Stack**

|  |
| --- |
| def create\_stack():  stack = []  return stack  def check\_empty(stack):  return len(stack) == 0  def push(stack, item):  stack.append(item)  print("pushed item: " + item)  def pop(stack):  if (check\_empty(stack)):  return "stack is empty"  return stack.pop()  stack = create\_stack()  push(stack, str(1))  push(stack, str(2))  push(stack, str(3))  push(stack, str(4))  print("stack :" + str(stack))  print("\npopped item: " + pop(stack))  print("stack after popping an element: " + str(stack))  print("\n20DCE019 - Yatharth Chauhan") |

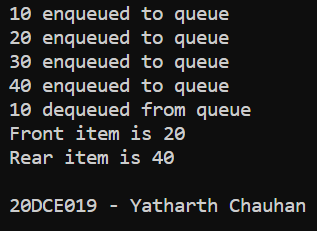
**OUTPUT:**

****

1. **Queue**

|  |
| --- |
| class Queue:  def \_\_init\_\_(self, capacity):  self.front = self.size = 0  self.rear = capacity - 1  self.Q = [None]\*capacity  self.capacity = capacity  def isFull(self):  return self.size == self.capacity  def isEmpty(self):  return self.size == 0  def EnQueue(self, item):  if self.isFull():  print("Full")  return  self.rear = (self.rear + 1) % (self.capacity)  self.Q[self.rear] = item  self.size = self.size + 1  print("% s enqueued to queue" % str(item))  def DeQueue(self):  if self.isEmpty():  print("Empty")  return  print("% s dequeued from queue" % str(self.Q[self.front]))  self.front = (self.front + 1) % (self.capacity)  self.size = self.size - 1  def que\_front(self):  if self.isEmpty():  print("Queue is empty")  print("Front item is", self.Q[self.front])  def que\_rear(self):  if self.isEmpty():  print("Queue is empty")  print("Rear item is", self.Q[self.rear])  if \_\_name\_\_ == '\_\_main\_\_':  queue = Queue(30)  queue.EnQueue(10)  queue.EnQueue(20)  queue.EnQueue(30)  queue.EnQueue(40)  queue.DeQueue()  queue.que\_front()  queue.que\_rear()  print("\n20DCE019 - Yatharth Chauhan") |

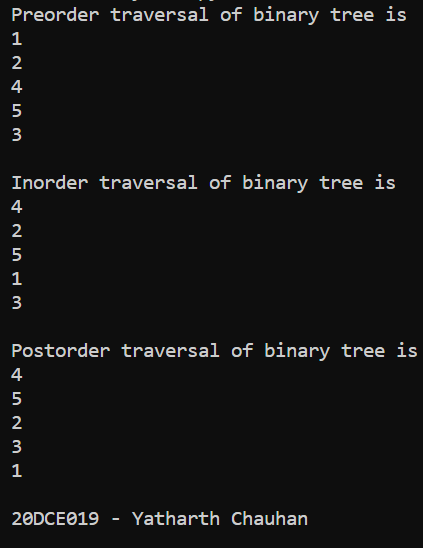
**OUTPUT:**

****

1. **Binary Tree**

|  |
| --- |
| class Node:  def \_\_init\_\_(self, key):  self.left = None  self.right = None  self.val = key  def printInorder(root):  if root:  printInorder(root.left)  print(root.val),  printInorder(root.right)  def printPostorder(root):  if root:  printPostorder(root.left)  printPostorder(root.right)  print(root.val),  def printPreorder(root):  if root:  print(root.val),  printPreorder(root.left)  printPreorder(root.right)  root = Node(1)  root.left = Node(2)  root.right = Node(3)  root.left.left = Node(4)  root.left.right = Node(5)  print("Preorder traversal of binary tree is")  printPreorder(root)  print("\nInorder traversal of binary tree is")  printInorder(root)  print("\nPostorder traversal of binary tree is")  printPostorder(root)  print("\n20DCE019 - Yatharth Chauhan") |

**OUTPUT:**

****

**CONCLUSION:** In this practical we learnt some basic data structures implementation in python for example linked list, stack, queue and binary tree.

**PRACTICAL – (11)**

**AIM: Develop programs to understand the working of exception handling with the user guessing a number until he/she gets it right.**

**Program:**

import sys

randomList = ['a', 0, 2]

for entry in randomList:

try:

print("The Entry is", entry)

f = 1/int(entry)

break

except:

print("Ooops!", sys.exc\_info()[0], " occured")

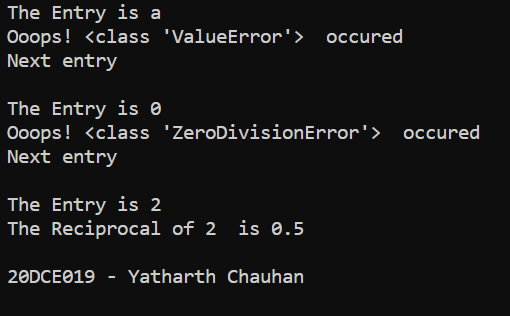
print("Next entry")

print()

print("The Reciprocal of", entry, " is", f)

print("\n20DCE019 - Yatharth Chauhan")

**OUTPUT:**

****

**CONCLUSION:** In this practical, we learned to handle exception in python program.

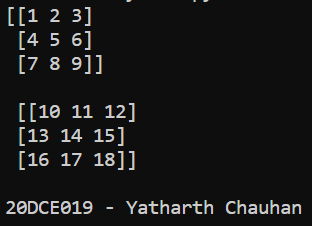
**PRACTICAL – 12.1**

**Aim:** Create two 2D Numpy arrays with random numbers and concatenate them using the Numpy library. After Concatenation, reshape the resulting Numpy array such that the number of rows and columns is reversed.

**PROGRAM:**

|  |
| --- |
| import numpy as np  # creating two 2D arrays  arr1 = np.arange(1, 10).reshape(3, 3)  arr2 = np.arange(10, 19).reshape(3, 3)  print(arr1)  print("\n", arr2)  # concating operations (axis 1 for column, 0 for row)  np.concatenate((arr1, arr2), axis=1)  print("\n20DCE019 - Yatharth Chauhan") |

**OUTPUT:**

****

**CONCLUSION:** In this practical, we learned to use Numpy Arrays and concatenate those together.

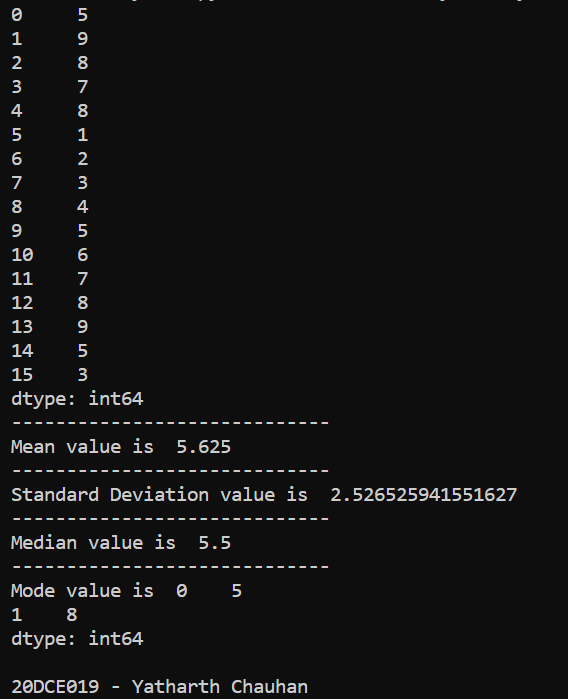
**PRACTICAL – (12.2)**

**AIM: Create a Pandas series from a Python List. Find out the mean, median, mode, range and standard deviation of the series.**

**PROGRAM:**

|  |
| --- |
| import pandas as pd  s = pd.Series(data=[5, 9, 8, 7, 8, 1, 2, 3, 4, 5, 6, 7, 8, 9, 5, 3])  print(s)  print("-----------------------------")  print("Mean value is ", s.mean())  print("-----------------------------")  print("Standard Deviation value is ", s.std())  print("-----------------------------")  print("Median value is ", s.median())  print("-----------------------------")  print("Mode value is ", s.mode())  print("\n20DCE019 - Yatharth Chauhan") |

**OUTPUT:**

****

**CONCLUSION:** In this Practical, we learned about Panda series from Python List to perform mathematical operations.