**1.Create a list and perform the following methods append ,copy ,clear , count ,extend ,index ,insert, pop ,remove, reverse,sort, min,max**

**Output:**

**['Mathematics', 'chemistry', 1997, 2000, 20544]**

**Output:**

**['Mathemat ics', 'chemistry', 10087, 1997, 2000, 20544]**

**Output:**

**[1, 2, 3, 2, 3, 4, 5]**

**[2, 3, 4, 5, 1, 2, 3, 2, 3, 4, 5]**

**Output:**

**15**

**Output:**

**4**

**List = [1, 2, 3, 1, 2, 1, 2, 3, 2, 1]**

**print(len(List))**

**Output:**

**10**

**List = [1, 2, 3, 1, 2, 1, 2, 3, 2, 1] print(List.index(2))**

**Output:1**

**numbers = [5, 2, 8, 1, 9] print(min(numbers))**

**Output:1**

**numbers = [5, 2, 8, 1, 9] print(max(numbers))**

**Output:9**

**Output:**

**[5.33, 4.445, 3, 2.5, 2.3, 1.054]**

**Output:2.5**

**Output:2.3**

**Output:[4.445, 3 , 5.33, 1.054, 2.5]**

**Output:[2.3, 4.445, 5.33, 1.054, 2.5]**

**2.Write a python program to add two numbers.**

# This program adds two numbers num1 = 1.5 num2 = 6.3

# Add two numbers sum = num1 + num2

# Display the sum

print('The sum of {0} and {1} is {2}'.format(num1, num2, sum))

## Output:

The sum of 1.5 and 6.3 is 7.8

## Source Code:

# Store input numbers

num1 = input('Enter first number: ') num2 = input('Enter second number: ') # Add two numbers

sum = float(num1) + float(num2) # Display the sum

print('The sum of {0} and {1} is {2}'.format(num1, num2, sum))

## Output:

Enter first number: 1.5 Enter second number: 6.3

The sum of 1.5 and 6.3 is 7.8

**4.** **Write a program to double a given number and add two numbers using lambda ()?**

**(lambda x : x \* 2)(3) sum = lambda a,b: a+b num1 = 5**

**num2 = 3**

**result = sum(num1,num2)**

**print(f"Sum of {num1} and {num2} is {result}.")**

## Output:

**6**

**Sum of 5 and 3 is 8.**

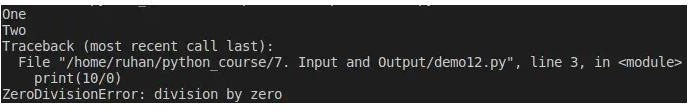
**5.Demonstrate a python code to implement abnormal termination**?

print('One')

print('Two') print(10/0) print('Four')

print('Five')

## Output:



6. **Write a python program to get python version**

**import sys print("Python version") print (sys.version) print("Version info.") print (sys.version\_info)**

## Output:

**Python version 3.5.2**

**(default, Sep 10 2016, 08:21:44) [GCC 5.4.0 20160609]**

**Version info.**

**sys.version\_info(major=3, minor=5, micro=2, releaselevel='final', serial=0)**

**7.Write a python program to print date, time for today and now.**

**from datetime import datetime**

**# datetime object containing current date and time now = datetime.now()**

**print("now =", now)**

**# dd/mm/YY H:M:S**

**dt\_string = now.strftime("%d/%m/%Y %H:%M:%S") print("date and time =", dt\_string)**

## Output:

10:09:20

date and time = 27/12/2022

now = 2022 - 12 - 27 10:09:20.430322

**8.Write a python Program to display welcome to MRCET by using classes and objects**

**class Welcome:**

**def init (self, name):**

**self.name = name def greet(self):**

**print("Welcome to " + self.name) w1 = Welcome ("MRCET")**

**w1.greet()**

## Output:

**Welcome to MRCET**

**9.Using a numpy module create an array and check the following: 1. Type of array 2. Axes of array 3. Shape of array 4. Type of elements in array**

**import numpy as np**

**# Create an array (for example, a 2x3 array) my\_array = np.array([[1, 2, 3], [4, 5, 6]]) # 1. Type of array**

**print("Type of array:", type(my\_array))**

**# 2. Axes of array**

**print("Number of axes (dimensions) in the array:", my\_array.ndim)**

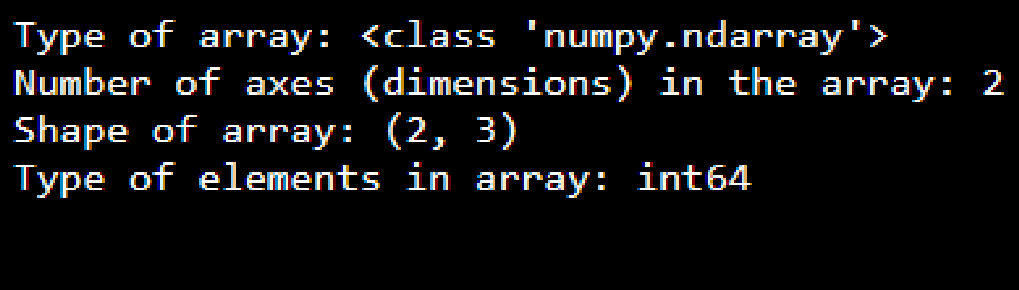
**# 3. Shape of array**

**print("Shape of array:", my\_array.shape)**

**# 4. Type of elements in array**

**print("Type of elements in array:", my\_array.dtype)**

## Output:



**10.Write a python program to concatenate the dataframes with two different objects.**

**import pandas as pd**

**# First DataFrame**

**df1 = pd.DataFrame({'id': ['A01', 'A02', 'A03', 'A04'],**

**'Name': ['ABC', 'PQR', 'DEF', 'GHI']})**

**# Second DataFrame**

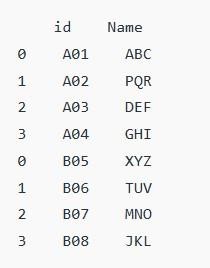
**df2 = pd.DataFrame({'id': ['B05', 'B06', 'B07', 'B08'],**

**'Name': ['XYZ', 'TUV', 'MNO', 'JKL']})**

**frames = [df1, df2]**

**result = pd.concat(frames) display(result)**

## Output:

****

**11. Write a python code to set background color and pic and draw a circle using turtle module**

**import turtle**

**# Set background color turtle.bgcolor("lightblue")**

**# Display an image (replace 'path\_to\_your\_image' with the actual path) screen = turtle.Screen()**

**screen.bgpic("C:/Users/ANITA-WIN10PRO/Pictures/Screenshots/Screenshot.png") # Make sure to replace with your image path**

**# Create a turtle pen = turtle.Turtle()**

**# Draw a circle pen.color("red") pen.begin\_fill() pen.circle(100) pen.end\_fill()**

**# Hide the turtle pen.hideturtle()**

**# Keep the window open until user closes it turtle.mainloop()**

## Output:



**12.Write a Python program to find first n prime numbers**

**nprime=int(input("Enter How many prime numbers you want to print:")) n = 0**

**i = 0**

**while n < nprime:**

**i += 1**

**count = 1**

**for j in range(2, i):**

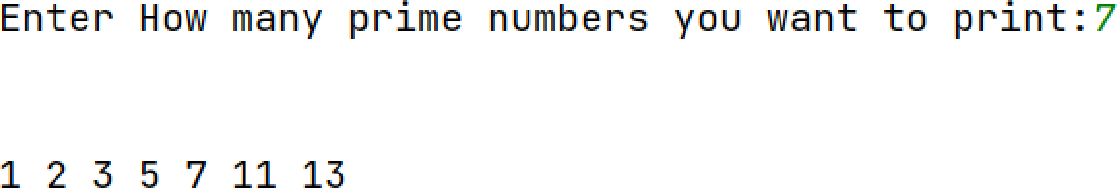
**if i % j == 0: count = 0 break**

**if count == 1:**

**print(i, end=' ')**

**n += 1**

## Output:

****

**13.Write a Python program to multiply matrices.**

**import numpy as np**

**# take a 3x3 matrix A = [[12, 7, 3],**

**[4, 5, 6],**

**[7, 8, 9]]**

**# take a 3x4 matrix B = [[5, 8, 1, 2],**

**[6, 7, 3, 0],**

**[4, 5, 9, 1]]**

**# result will be 3x4 result= [[0,0,0,0],**

**[0,0,0,0],**

**[0,0,0,0]]**

**result = np.dot(A,B) for r in result:**

**print(r)**

**Output:**

**[114, 160, 60, 27]**

**[74, 97, 73, 14]**

**[119, 157, 112, 23]**

**14.Write a Python program to find the exponentiation of a number**

**def exponentiate(base, exponent): result = base \*\* exponent return result**

**# Input the base and exponent**

**base = float(input("Enter the base: ")) exponent = float(input("Enter the exponent: "))**

**result = exponentiate(base, exponent)**

**print(f"{base} raised to the power of {exponent} is: {result}")**

## Output:

**Enter the base: 5 Enter the exponent: 3**

**5.0 raised to the power of 3.0 is: 125.0**

**15.Using a for loop, write a program that prints out the decimal equivalents of ½+1/3+1/4+. . . , 1/10**

**decimal\_equivalents = [ ]**

**for denominator in range(2, 11): fraction = 1 / denominator**

**decimal\_equivalents.append(fraction)**

**print("Decimal equivalents:")**

**for i, fraction in enumerate(decimal\_equivalents, start=2): print(f"1/{i} =", fraction)**

## Output:

**Decimal equivalents: 1/2 = 0.5**

**1/3 = 0.3333333333333333**

**1/4 = 0.25**

**1/5 = 0.2**

**1/6 = 0.16666666666666666**

**1/7 = 0.14285714285714285**

**1/8 = 0.125**

**1/9 = 0.1111111111111111**

**1/10 = 0.1**