**SUB:-** Python Practical

**SUB CODE:-**

**ROLL NO:-** 39

**NAME:-** Prajval Raju Bhale.

**Q2]. Programs for understanding the data types ,control flow statements ,blocks and loops**

num = int(input("enter integer value"))

print(num)

print(type(num))

for x in range(1,num):

if(x%2==0):

print(x ,"=Even number")

else:

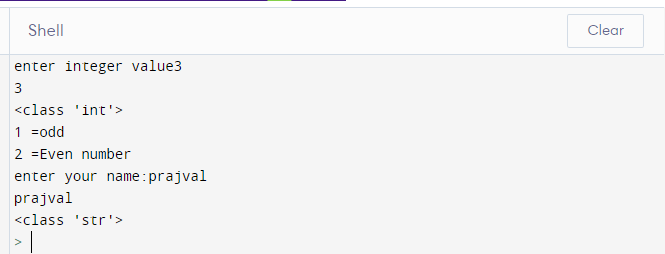
print(x , "=odd")

name = input("enter your name:")

print(name)

print(type(name))

**OUTPUT:-**



**Q3]. Programs for understanding functions, use of built in function ,user defined function.**

def factorial():

num = int(input("enter number to find factorial:"))

fact=1

for i in range(1,num+1):

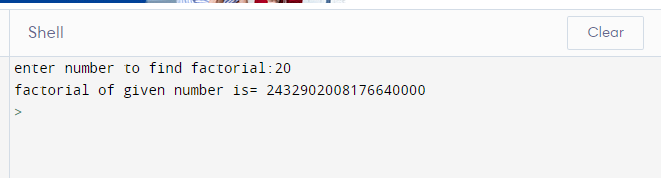
fact=fact \* i

print("factorial of given number is=",fact)

factorial()

#In above program factorial is user defined funtion and print funtion is in built function

**OUTPUT:**



**Q4]. Programs to use extisting modules,packages and creating modules , packages**.

def add():

num1 = int(input("enter num1="))

num2 = int(input("enter num1="))

sum = num1 + num2

print("Addition of two number is=", sum)

def mult():

num1 = int(input("enter num1="))

num2 = int(input("enter num1="))

sum = num1 \* num2

print("Multiplication of two number is=", sum)

program2nd:

import Q4

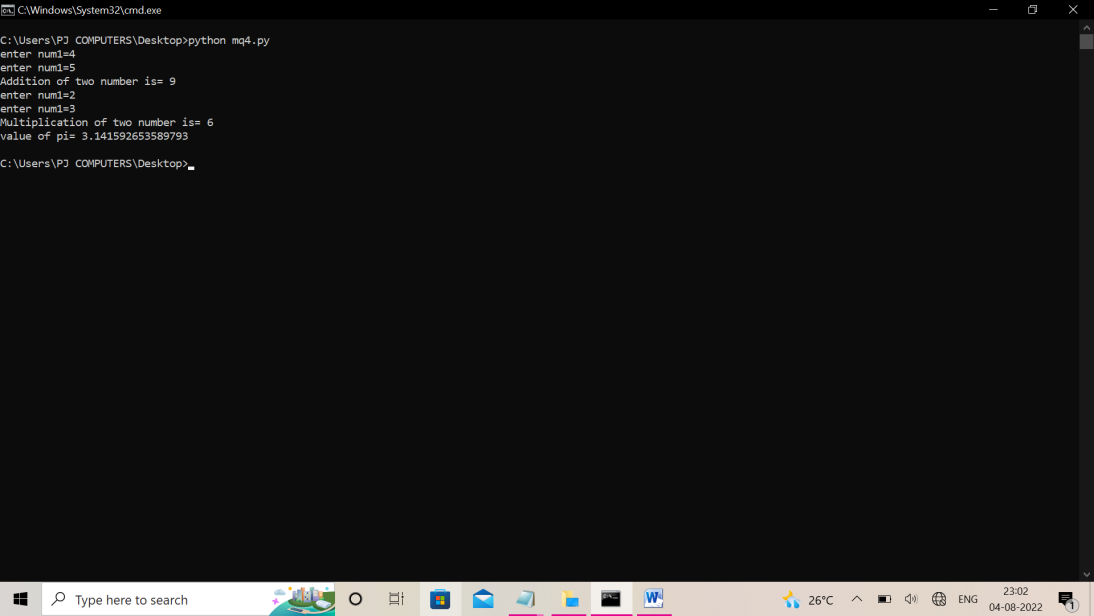
import math

Q4.add()

Q4.mult()

print("value of pi=",math.pi)

**OUTPUT:**



**Q5]. Programs for implementation of all object oriented concepts like class ,methods ,inheritance ,polymorphism etc.**

class India():

def capital(self):

print("Mumbai is the economic capital of India.")

def language(self):

print("Marathi is the most widely spoken language of Maharashtra.")

def type(self):

print("Mumbai is a developing City.")

class USA():

def capital(self):

print("Washington, D.C. is the capital of USA.")

def language(self):

print("English is the primary language of USA.")

def type(self):

print("USA is a developed country.")

obj\_ind = India()

obj\_usa = USA()

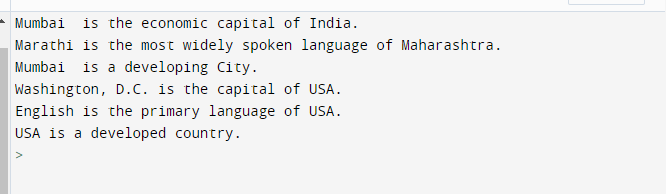
for country in (obj\_ind, obj\_usa):

country.capital()

country.language()

country.type()

**OUTPUT:-**



**Q6]. Programs for parsing of data ,validations like passwords ,email url etc.**

import re

regex = '^[a-z0-9]+[\.\_]?[a-z0-9]+[@]\w+[.]\w{2,3}$'

def check(email):

if(re.search(regex,email)):

print("Valid Email")

else:

print("Invalid Email")

if \_\_name\_\_ == '\_\_main\_\_' :

email = "prajval@009.org"

check(email)

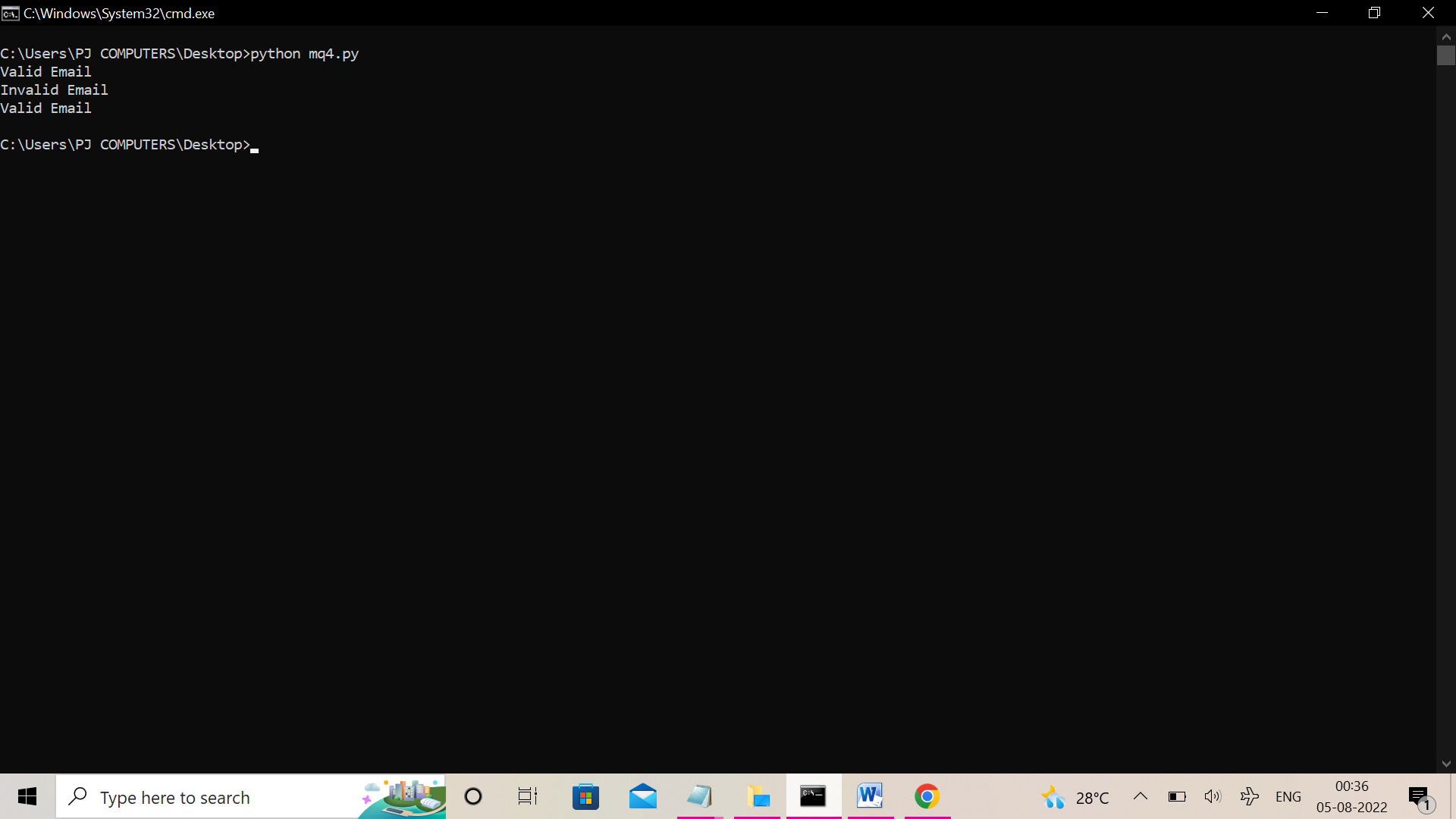
email = "prajwalbhale007@gmail.com"

check(email)

email = "abcd@gmail.com"

check(email)

**OUTPUT:**



**Q 7 - A]. Programs for pattern finding should be coverd.**

print("Print equilateral triangle Pyramid using asterisk symbol ")

# printing full Triangle pyramid using stars

size = 7

m = (2 \* size) - 2

for i in range(0, size):

for j in range(0, m):

print(end=" ")

# decrementing m after each loop

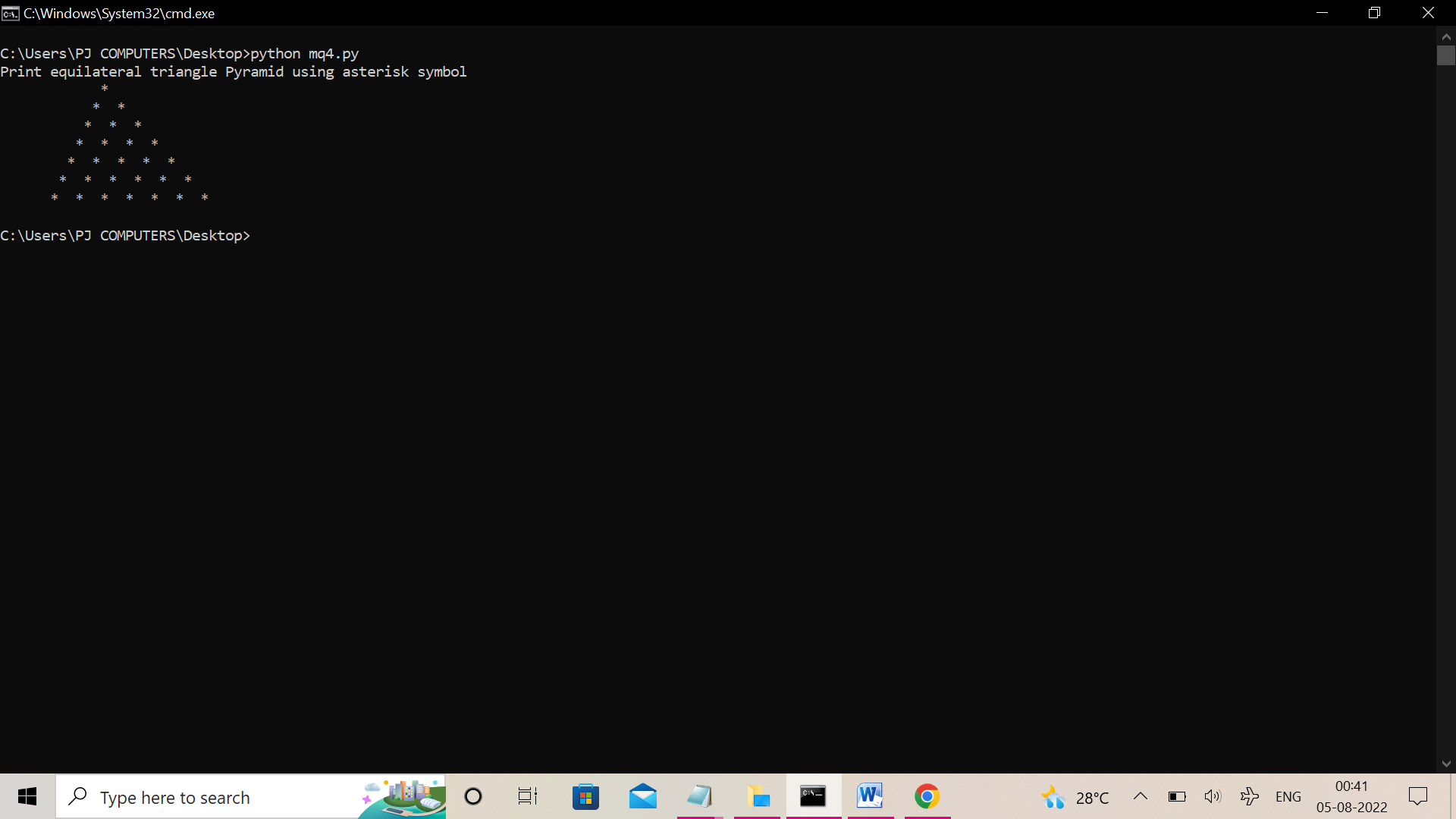
m = m - 1

for j in range(0, i + 1):

print("\* ", end=' ')

print(" ")

**OUTPUT:**



**Q7]. B**

rows = 5

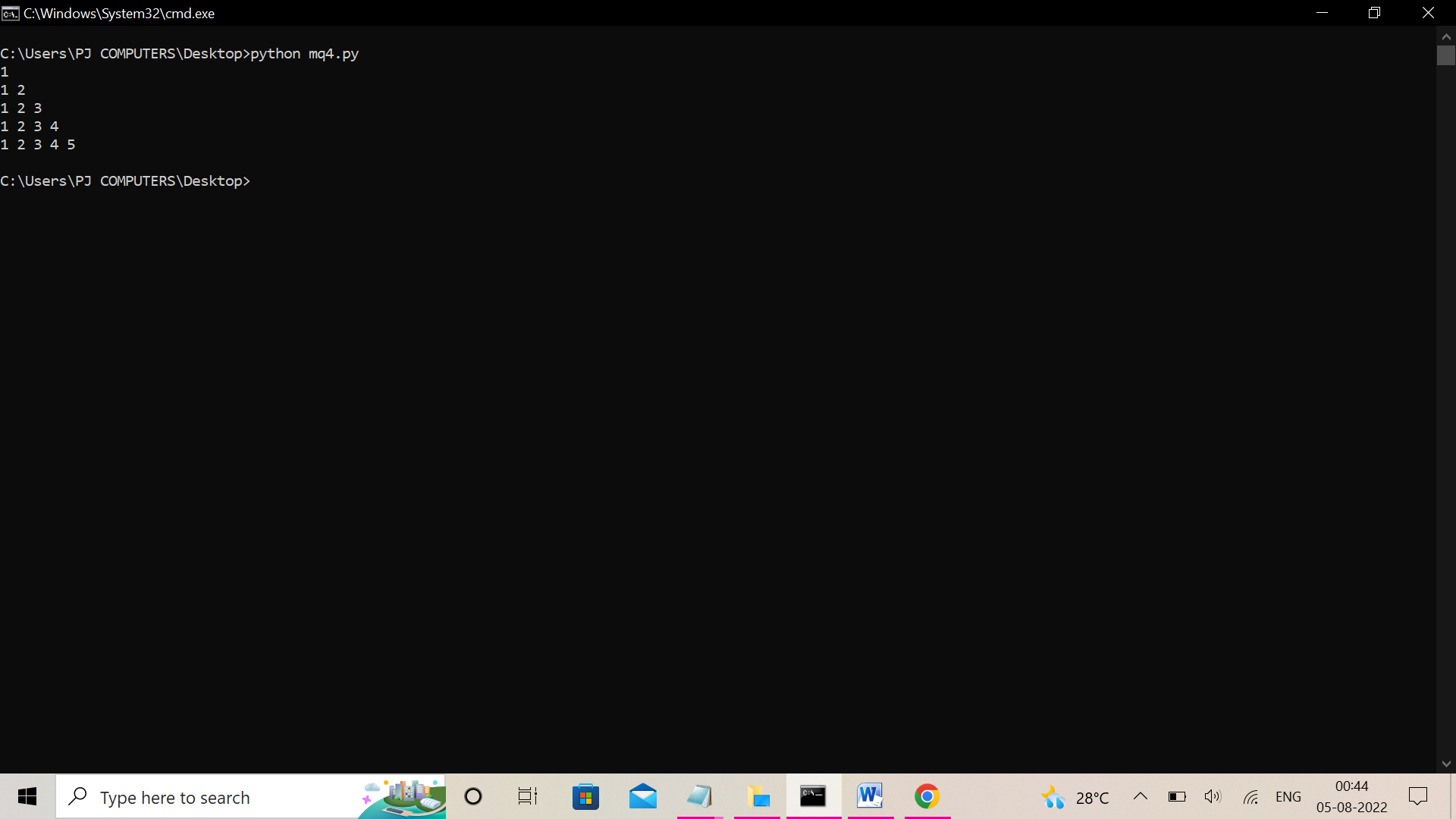
for i in range(1, rows + 1):

for j in range(1, i + 1):

print(j, end=' ')

print('')

**OUTPUT:**



**Q7]. C**

ascii\_number = 65

rows = 7

for i in range(0, rows):

for j in range(0, i + 1):

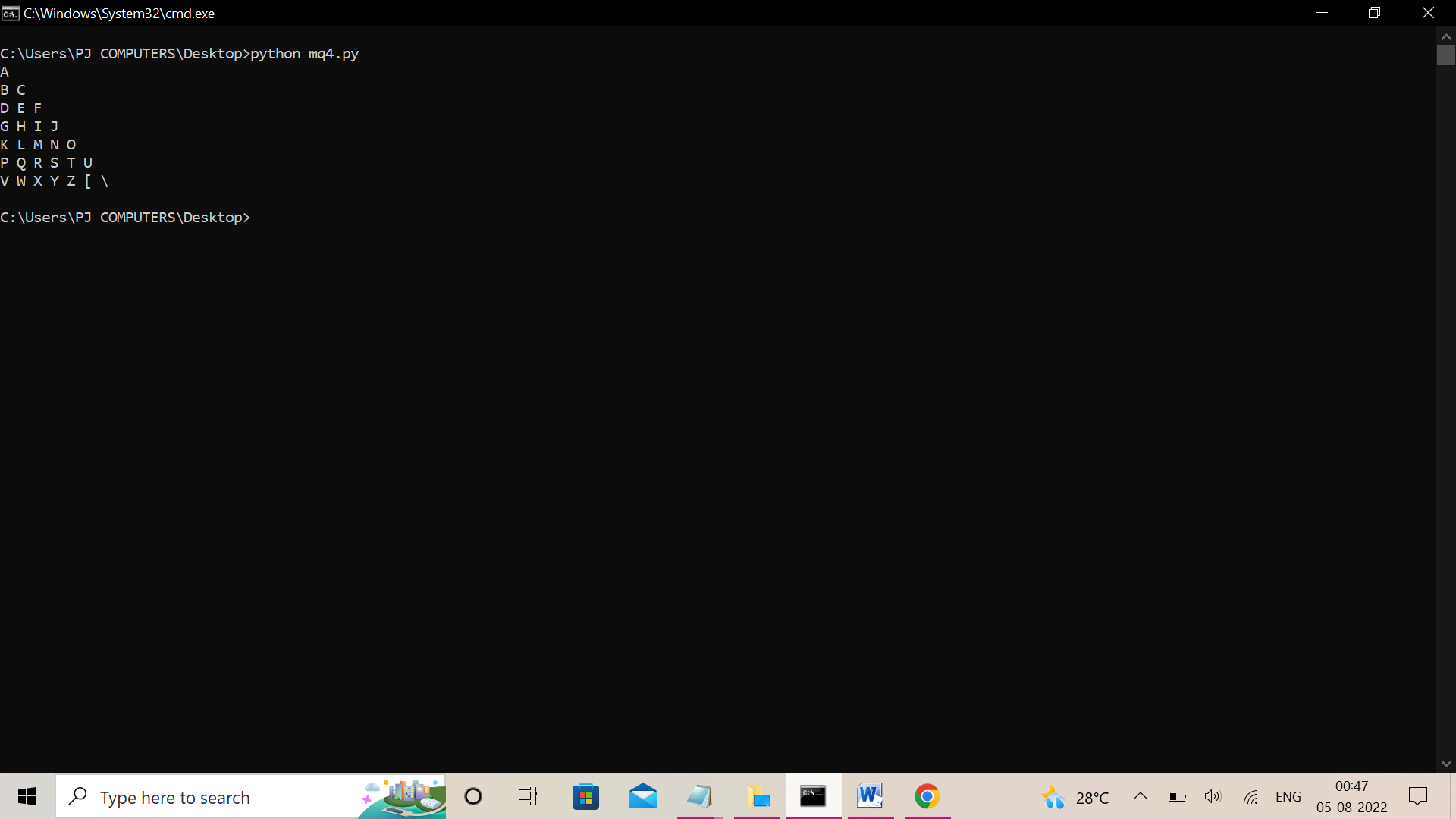
character = chr(ascii\_number)

print(character, end=' ')

ascii\_number += 1

print(" ")

**OUTPUT:**



**Q8].Programs covering all the aspects of exception handling , user defined exception, multithreading should be coverd**.

import threading

import sys

class MyThread(threading.Thread):

def someFunction(self):

print("Hello World")

def run(self):

self.someFunction()

def join(self):

threading.Thread.join(self)

def main():

t = MyThread()

t.start()

t.join()

if \_\_name\_\_ == '\_\_main\_\_':

main()

**OUTPUT:**

