**Question 1:**

Number game between user and computer. The user starts by entering either 1 or 2 or 3 digits starting from 1 sequentially. The computer can return either 1 or 2 or 3 next digits in sequence, starting from the max number played by the user. User enters the next 1 or 2 or 3 next digits in sequence, starting from the max number played by the computer. Whoever reaches 20 first wins the game.

Note:

- the numbers should be in sequence starting from 1.

- minimum number user or computer should pick is at least 1 digit in sequence

- maximum number user or computer can pick only 3 digits in sequence

**Example 1:**

Player: 1 2

Computer played: [3, 4]

Player: 5 6 7

Computer played: [8, 9]

Player: 10

Computer played: [11, 12, 13]

Player: 14 15

Computer played: [16, 17, 18]

Player: 19 20

Player Wins!!!

**Example 2:**

Player: 1

Computer played: [2, 3]

Player: 4 5

Computer played: [6, 7, 8]

Player: 9 10

Computer played: [11]

Player: 12

Computer played: [13]

Player: 14 15

Computer played: [16]

Player: 17 18

Computer played: [19, 20]

Computer Wins!!!

**Code.py**

import random

def computer\_turn(last\_number):

 # Computer picks between 1 and 3 sequential numbers

 pick = random.randint(1, 3)

next\_numbers = list(range(last\_number + 1, min(last\_number + pick + 1, 21)))

 return next\_numbers

def player\_turn(last\_number):

 while True:

 try:

player\_input = list(map(int, input("Player's turn: ").split()))

 if all(x == last\_number + i + 1 for i, x in enumerate(player\_input)) and 1 <= len(player\_input) <= 3:

 return player\_input

 else:

print("Invalid input. You must enter 1 to 3 consecutive numbers starting from", last\_number + 1)

 except ValueError:

print("Invalid input. Please enter numbers.")

def play\_game():

print("Welcome to the number game! The first to reach or exceed 20 wins.")

last\_number = 0

 while last\_number< 20:

 # Player's turn

player\_numbers = player\_turn(last\_number)

last\_number = player\_numbers[-1]

print(f"Player played: {player\_numbers}")

 if last\_number>= 20:

print("Player Wins!!!")

 break

 # Computer's turn

computer\_numbers = computer\_turn(last\_number)

last\_number = computer\_numbers[-1]

print(f"Computer played: {computer\_numbers}")

 if last\_number>= 20:

print("Computer Wins!!!")

 break

play\_game()

**Question 2:**

Develop a function called ncr(n,r) which computes r-combinations of n-distinct object . use this function to print pascal triangle, where number of rows is the input

**Code.py**

def ncr(n, r):

 from math import factorial

 return factorial(n) // (factorial(r) \* factorial(n - r))

def pascal\_triangle(rows):

 for i in range(rows):

 row = [ncr(i, j) for j in range(i + 1)]

print(' '.join(map(str, row)))

# Example usage

rows = int(input("Enter number of rows for Pascal's Triangle: "))

pascal\_triangle(rows)

**Question 3:**

Read a list of n numbers during runtime. Write a Python program to print the repeated elements with frequency count in a list.

Example :

Input:- [ 2,1,2,3,4,5,1,3,6,2,3,4]

Output:-

Element 2 has come 3 times

Element 1 has come 2 times

Element 3 has come 2 times

Element 4 has come 2 times

Element 1 has come 1 times

Element 6 has come 1 times

**Code.py**

from collections import Counter

def count\_repeated\_elements(nums):

 counter = Counter(nums)

 for element, count in counter.items():

print(f"Element {element} has come {count} times")

# Example usage

nums = list(map(int, input("Enter numbers separated by space: ").split()))

count\_repeated\_elements(nums)

**Question 4:-**

Develop a python code to read matric A of order 2X2 and Matrix B of order 2X2 from a file and perform the addition of Matrices A & B and Print the results.

**Code.py**

def read\_matrix(filename):

 with open(filename, 'r') as file:

 matrix = [list(map(int, line.split())) for line in file.readlines()]

 return matrix

def add\_matrices(A, B):

 return [[A[i][j] + B[i][j] for j in range(len(A[0]))] for i in range(len(A))]

def print\_matrix(matrix):

 for row in matrix:

print(' '.join(map(str, row)))

# Example usage (assuming matrices A and B are in 'matrices.txt')

A = read\_matrix('matrixA.txt') # Reads Matrix A from file

B = read\_matrix('matrixB.txt') # Reads Matrix B from file

result = add\_matrices(A, B)

print("Result of Matrix Addition:")

print\_matrix(result)

**Question 5:-**

Write a program that overloads the + operator so that it can add two objects of the class Fraction.

Fraction can be considered of the for P/Q where P is the numerator and Q is the denominator

**Code.py**

import math

class Fraction:

 def \_\_init\_\_(self, numerator, denominator):

self.numerator = numerator

self.denominator = denominator

 def \_\_add\_\_(self, other):

 # Finding Least Common Denominator (LCM)

lcm\_denom = (self.denominator \* other.denominator) // math.gcd(self.denominator, other.denominator)

 # Adjusting numerators

 adjusted\_num1 = self.numerator \* (lcm\_denom // self.denominator)

 adjusted\_num2 = other.numerator \* (lcm\_denom // other.denominator)

 # Adding the numerators

new\_num = adjusted\_num1 + adjusted\_num2

 # Returning a new Fraction object

 return Fraction(new\_num, lcm\_denom)

 def \_\_str\_\_(self):

 return f"{self.numerator}/{self.denominator}"

# Function to get user input for a fraction

def get\_fraction\_input():

 numerator = int(input("Enter numerator: "))

 denominator = int(input("Enter denominator: "))

 return Fraction(numerator, denominator)

# Example usage with user input

print("Enter the first fraction:")

frac1 = get\_fraction\_input()

print("Enter the second fraction:")

frac2 = get\_fraction\_input()

result = frac1 + frac2

print("Sum of fractions:", result)