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!!!

Solution:

```
import random


def get_user_move(current_number):
    """Prompts the user for their move and ensures it’s valid."""
    while True:
        try:
            user_input = input("Your turn (enter 1 to 3 consecutive numbers): ")
            user_numbers = list(map(int, user_input.split()))


            # Validate that user picked between 1 to 3 numbers and they're sequential
            if 1 <= len(user_numbers) <= 3 and all(user_numbers[i] == current_number + i + 1 for i in
range(len(user_numbers))):
                return user_numbers
            else:
                print("Invalid input! Pick 1, 2, or 3 consecutive numbers in sequence.")
        except ValueError:
            print("Invalid input! Enter numbers separated by spaces.")


def get_computer_move(current_number):
    """Computer picks between 1 to 3 numbers in sequence."""
    # Randomly decide how many numbers to pick (1 to 3) without exceeding 20
    num_picks = random.randint(1, min(3, 20 - current_number))
```

```
    computer_numbers = [current_number + i + 1 for i in range(num_picks)]
    print("Computer played:", computer_numbers)
    return computer_numbers

def play_game():
    print("Welcome to the Number Game! Reach 20 to win.")
    current_number = 0

    while current_number < 20:
        # User's turn
        user_move = get_user_move(current_number)
        current_number = user_move[-1]  # Update the current number to the user's last pick
        if current_number >= 20:
            print("You reached 20. You win!")
            return

        # Computer's turn
        computer_move = get_computer_move(current_number)
        current_number = computer_move[-1]  # Update the current number to the computer's last pick
        if current_number >= 20:
            print("Computer reached 20. Computer wins!")
            return

# Start the game
play_game()
```

Output:

```
Welcome to the Number Game! Reach 20 to win.
```

Your turn (enter 1 to 3 consecutive numbers): 1

Computer played: [2, 3]

Your turn (enter 1 to 3 consecutive numbers): 4

Computer played: [5, 6]

Your turn (enter 1 to 3 consecutive numbers): 7

Computer played: [8, 9, 10]

Your turn (enter 1 to 3 consecutive numbers): 11

Computer played: [12]

Your turn (enter 1 to 3 consecutive numbers): 13

Computer played: [14, 15, 16]

Your turn (enter 1 to 3 consecutive numbers): 17

Computer played: [18, 19]

Your turn (enter 1 to 3 consecutive numbers): 20

You reached 20. You win!

==========Second computer wins========

Welcome to the Number Game! Reach 20 to win.

Your turn (enter 1 to 3 consecutive numbers): 1

Computer played: [2]

Your turn (enter 1 to 3 consecutive numbers):

Invalid input! Pick 1, 2, or 3 consecutive numbers in sequence.

Your turn (enter 1 to 3 consecutive numbers): 3

Computer played: [4]

Your turn (enter 1 to 3 consecutive numbers): 5

Computer played: [6, 7]

Your turn (enter 1 to 3 consecutive numbers): 8

Computer played: [9, 10, 11]

Your turn (enter 1 to 3 consecutive numbers): 12

Computer played: [13, 14, 15]

Your turn (enter 1 to 3 consecutive numbers): 16

Computer played: [17]

Your turn (enter 1 to 3 consecutive numbers): 18 19

Computer played: [20]

Computer reached 20. Computer wins!

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

Solution:

```
def factorial(num):
    """Helper function to calculate the factorial of a number."""
    if num == 0 or num == 1:
        return 1
    result = 1
    for i in range(2, num + 1):
        result *= i
    return result

def ncr(n, r):
    """Computes the combination nCr (n choose r)."""
    return factorial(n) // (factorial(r) * factorial(n - r))

def print_pascals_triangle(rows):
    """Prints Pascal's Triangle up to the given number of rows."""
    for n in range(rows):
        # Print spaces to center-align the triangle
        print(" " * (rows - n), end="")
        # Compute each entry in row n
        for r in range(n + 1):
            print(ncr(n, r), end=" ")
        print()  # Move to the next line after each row

# Example usage
num_rows = int(input("Enter the number of rows for Pascal's Triangle: "))
print_pascals_triangle(num_rows)
```

Output:

Enter the number of rows for Pascal's Triangle: 7

```
      1
     1 1
    1 2 1
   1 3 3 1
  1 4 6 4 1
 1 5 10 10 5 1
1 6 15 20 15 6 1
```

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

Solution:

```
from collections import Counter


# Input list from user
n = int(input("Enter the number of elements in the list: "))
input_list = []
print("Enter the elements:")
for _ in range(n):
    element = int(input())
    input_list.append(element)
```

```
# Count frequency of each element
frequency = Counter(input_list)

# Display the repeated elements with their counts
print("Repeated elements with their frequency:")
for element, count in frequency.items():
    print(f"Element {element} has come {count} times")
```

Output:

```
Enter the number of elements in the list: 8
Enter the elements:
1
2
3
2
4
5
3
2
Repeated elements with their frequency:
Element 1 has come 1 times
Element 2 has come 3 times
Element 3 has come 2 times
Element 4 has come 1 times
Element 5 has come 1 times
```

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.

Solution:

#create matrices.txt file with below matrix

Matrix A

1 2

3 4

Matrix B

5 6

7 8

#RUN THE BELOW CODE

```
def read_matrix(file, start_line):
    """Reads a 2x2 matrix from the file starting from a specific line."""
    matrix = []
    for i in range(start_line, start_line + 2):
        row = list(map(int, file[i].strip().split()))
        matrix.append(row)
    return matrix


def add_matrices(matrix_a, matrix_b):
    """Adds two 2x2 matrices and returns the resulting matrix."""
    result = []
    for i in range(2):
        row = []
        for j in range(2):
            row.append(matrix_a[i][j] + matrix_b[i][j])
        result.append(row)
```

```
    return result

# Read the matrices from the file
with open("matrices.txt", "r") as file:
    lines = file.readlines()
    matrix_a = read_matrix(lines, 0)  # Read Matrix A starting at line 0
    matrix_b = read_matrix(lines, 3)  # Read Matrix B starting at line 3

# Perform matrix addition
result_matrix = add_matrices(matrix_a, matrix_b)

# Print the result
print("Resultant Matrix (A + B):")
for row in result_matrix:
    print(row)
```

Output:

Resultant Matrix (A + B):

[6, 8]

[10, 12]

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

Solution:

```
from math import gcd

class Fraction:
    def __init__(self, numerator, denominator):
        if denominator == 0:
```

```
            raise ValueError("Denominator cannot be zero")
        self.numerator = numerator
        self.denominator = denominator
        self.simplify()

    def simplify(self):
        common_divisor = gcd(self.numerator, self.denominator)
        self.numerator //= common_divisor
        self.denominator //= common_divisor

    def __add__(self, other):
        if isinstance(other, Fraction):
            new_numerator = (self.numerator * other.denominator) + (other.numerator *
self.denominator)
            new_denominator = self.denominator * other.denominator
            return Fraction(new_numerator, new_denominator)
        return NotImplemented

    def __str__(self):
        return f"{self.numerator}/{self.denominator}"

# Example usage
fraction1 = Fraction(3, 2)
fraction2 = Fraction(3, 1)

result = fraction1 + fraction2
print("Sum:", result)
```

Output:

Sum: 9/2