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.

CODE:

```
def user_turn(start):
    #while True:
        user_input = input(f"Enter your numbers starting from {start}: ")
        numbers = list(map(int, user_input.split()))

        if all(start <= num <= start + 2 for num in numbers) and len(numbers) in [1, 2, 3]:
            return numbers
        else:
            print("Invalid input. Please enter 1 to 3 sequential numbers.")



def computer_turn(start):
    import random

    count = random.randint(1,3)
    numbers = list(range(start, start + count ))

    print(f"Computer played: {', '.join(map(str, numbers))}")
    return numbers



def game():
    first_num = 1
```

```
    while first_num <=20:
        user_numbers = user_turn(first_num)
        first_num = first_num+len(user_numbers)

        if first_num > 20:
            print("You won!!!")
            break



        computer_numbers = computer_turn(first_num)
        first_num = first_num+len(computer_numbers)

        if first_num > 20:
            print("Computer won!!!")
            break
if __name__ =="__main__":
        game()
```

OUTPUT:

```
Enter your numbers starting from 1: 1 2
Computer played: 3, 4
Enter your numbers starting from 5: 5 6
Computer played: 7, 8
Enter your numbers starting from 9: 9 10
Computer played: 11, 12
Enter your numbers starting from 13: 13 14 15
Computer played: 16
Enter your numbers starting from 17: 17
Computer played: 18, 19
Enter your numbers starting from 20: 20
```

You won!!!

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:

```
def fact(num):
    if num == 0 or num == 1:
        return 1
    else:
        result = 1
        for i in range(2, num + 1):
            result = result*i
        return result


def ncr(n, r):
    if r > n or r < 0:
        return 0
    return fact(n) // (fact(r) * fact(n - r))


def print_pascal_triangle(rows):
    for n in range(rows):
        for r in range(n + 1):
            print(ncr(n, r), end=' ')
        print()


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

OUTPUT:

Enter the number of rows for Pascal's Triangle: 10

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

1 7 21 35 35 21 7 1

1 8 28 56 70 56 28 8 1

1 9 36 84 126 126 84 36 9 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.

CODE:

```
def count_frequencies(numbers):

    frequency = {}

    for number in numbers:
        if number in frequency:
            frequency[number] += 1
        else:
            frequency[number] = 1

    return frequency
```

```
def print_repeated_elements(frequency):

    repeated = {num: count for num, count in frequency.items() if count > 1}

    if repeated:
        print("Repeated elements with their frequency: ")
        for num, count in repeated.items():
            print(f"{num}: {count}")

def main():
    user_input = input("Enter numbers: ")
    numbers = list(map(int, user_input.split(',')))

    frequency = count_frequencies(numbers)

    print_repeated_elements(frequency)

if __name__ == "__main__":
    main()
```

OUTPUT:

```
Enter numbers: 1,2,3,4,5,5,4,3,2,1
Repeated elements with their frequency:
1: 2
2: 2
3: 2
4: 2
5: 2
```

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:

```
def read_matrix_from_file(filename):
    matrix = []

    file = open("matrices.txt",'w+')
    with open(filename, 'w+') as file:
        for line in file:
            matrix.append(list(map(int, line.split())))
    file.close()

    return matrix

def add_matrices(A, B):
    result = [[2, 2], [2, 2]]
    for i in range(0):
        for j in range(0):
            result[i][j] = A[i][j] + B[i][j]

    return result

def print_matrix(matrix):
    for row in matrix:
        print(' '.join(map(str, row)))

def main():
```

```
    matrix_A = read_matrix_from_file('matrices.txt')[:2]
    matrix_B = read_matrix_from_file('matrices.txt')[2:4]

    result_matrix = add_matrices(matrix_A, matrix_B)

    print("Result of Matrix A + Matrix B:")
    print_matrix(result_matrix)

if __name__ == "__main__":
    main()
```

OUTPUT:

Result of Matrix A + Matrix B:

2 2

2 2

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:

```
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):
        from math import gcd
        common_divisor = gcd(self.numerator, self.denominator)
        self.numerator //= common_divisor
        self.denominator //= common_divisor

    def __add__(self, other):
        if not isinstance(other, Fraction):
            return NotImplemented

        new_numerator = (self.numerator * other.denominator) + (other.numerator * self.denominator)
        new_denominator = self.denominator * other.denominator

        return Fraction(new_numerator, new_denominator)

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

    def __repr__(self):
        return f"Fraction ({self.numerator}, {self.denominator})"

if __name__ == "__main__":
    f1 = Fraction(1,2 )
    f2 = Fraction(3,4 )

    result = f1 + f2

    print(f"{f1} + {f2} = {result}")
```

OUTPUT:

1/2 + 3/4 = 5/4