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:

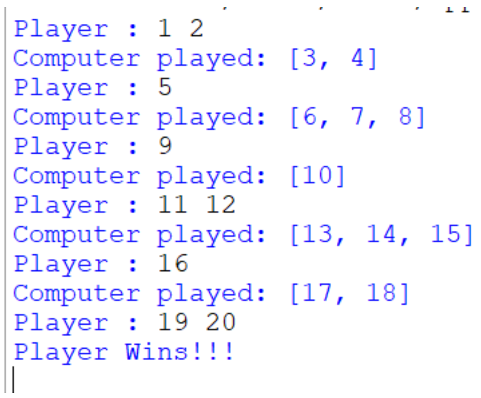
- the numbers should be in sequence starting from 1.

- 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

```
import random


def gettinginput(maxnum):
    while True:
        userinput = input("Player : ")
        numbers = list(map(int, userinput.split()))
        if all(num in range(maxnum, maxnum + 4) for num in numbers) and 1 <= len(numbers) <= 3:
            return numbers
        print("Invalid input. Please try again.")


def computerinput(maxnum):
    numtoplay = random.randint(1, 3)
    numbers = list(range(maxnum, maxnum + numtoplay))
    print(f"Computer played: {numbers}")
    return numbers


def main():
    maxnum = 1


    while maxnum <= 20:
        usernum = gettinginput(maxnum)
    maxnum += len(usernum)

    if maxnum >= 20:
      print("Player Wins!!!")
      break

    computernum = computerinput(maxnum)
    maxnum += len(computernum)

    if maxnum >= 20:
      print("Computer Wins!!!")
      break

main()
```

OUTPUT:



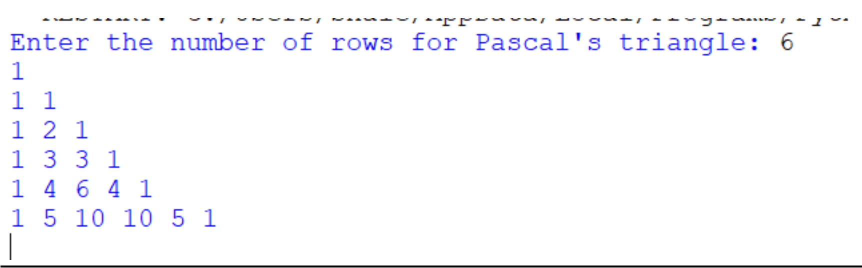
```
Player : 1 2
Computer played: [3, 4]
Player : 5
Computer played: [6, 7, 8]
Player : 9
Computer played: [10]
Player : 11 12
Computer played: [13, 14, 15]
Player : 16
Computer played: [17, 18]
Player : 19 20
Player Wins!!!
```

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

```
def ncr(n, r):

    if r > n or r < 0:

        return 0

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


def factorial(n):

    if n == 0 or n == 1:

        return 1

    result = 1

    for i in range(2, n + 1):

        result *= i

    return result


def print_pascal_triangle(rows):

    for i in range(rows):

        for j in range(i + 1):

            print(ncr(i, j), end=' ')

        print()


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

print_pascal_triangle(rows)
```

OUTPUT:



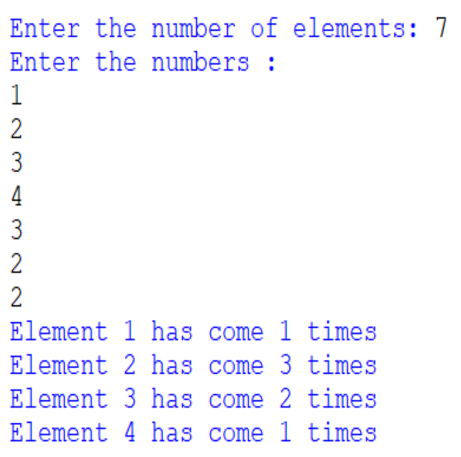
```
Enter the number of rows for Pascal's triangle: 6
1
1 1
1 2 1
1 3 3 1
1 4 6 4 1
1 5 10 10 5 1
```

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.

```
def count_frequencies(numbers):
    frequency = {}
    for num in numbers:
        if num in frequency:
            frequency[num] += 1
        else:
            frequency[num] = 1
    return frequency
def print_repeated_elements(frequency):
    for num, count in frequency.items():
        print(f"Element {num} has come {count} times")
n = int(input("Enter the number of elements: "))
numbers = []
print("Enter the numbers :")
for _ in range(n):
    number = int(input())
    numbers.append(number)
frequency = count_frequencies(numbers)
print_repeated_elements(frequency)
```

OUTPUT:



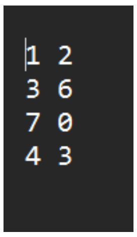
```
Enter the number of elements: 7
Enter the numbers :
1
2
3
4
3
2
2
Element 1 has come 1 times
Element 2 has come 3 times
Element 3 has come 2 times
Element 4 has come 1 times
```

QUESTION 4:

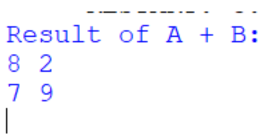
Develop a python code to read matric A of order 2X2 and Matrix B of order 2X2 from a file and perform the addition of Matrices A & B and Print the results

```
with open("matrices,txt", 'r') as file:
    lines = file.readlines()
    A = [[int(num) for num in lines[0].strip().split()],
         [int(num) for num in lines[1].strip().split()]]
    B = [[int(num) for num in lines[2].strip().split()],
        [int(num) for num in lines[3].strip().split()]]

def add_matrices(A, B):
    return [[A[0][0] + B[0][0], A[0][1] + B[0][1]],
            [A[1][0] + B[1][0], A[1][1] + B[1][1]]]

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

filename = 'matrices.txt'
A, B = read_matrices(filename)
result = add_matrices(A, B)
print("Result of A + B:")
print_matrix(result)
```

OUTPUT:



```
1 2
3 6
7 0
4 3
```



```
Result of A + B:
8 2
7 9
```

Matrices.txt

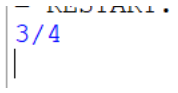
QUESTION 5:

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

```
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):
        def gcd(a, b):
            while b:
                a, b = b, a % b
            return abs(a)

        common_divisor = gcd(self.numerator, self.denominator)
        self.numerator //= common_divisor
        self.denominator //= common_divisor
        if self.denominator < 0:
            self.numerator = -self.numerator
            self.denominator = -self.denominator

    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)


f1 = Fraction(1, 2)

f2 = Fraction(1, 4)

result = f1 + f2

print(f"{result.numerator}/{result.denominator}")
```

OUTPUT:



```
= RESTART:
3/4
```