**ANSWERS FOR ASSIGNMENT-1**

**1A.**

def number\_game():

    user\_turn = True

    max\_num = 0

    while max\_num < 20:

        if user\_turn:

            user\_input = input("Enter 1, 2, or 3 digits starting from {}:".format(max\_num+1))

            user\_nums = list(map(int, user\_input.split()))

            if len(user\_nums) < 1 or len(user\_nums) > 3:

                print("Invalid input. Please enter 1, 2, or 3 digits.")

                continue

            if user\_nums[0] != max\_num + 1:

                print("Invalid input. Please start from the next number in sequence.")

                continue

            max\_num = user\_nums[-1]

            user\_turn = False

        else:

            computer\_nums = list(range(max\_num + 1, max\_num + 4))

            max\_num = computer\_nums[-1]

            print("Computer plays: {}".format(computer\_nums))

            user\_turn = True

    if user\_turn:

        print("Computer wins!")

    else:

        print("User wins!")

number\_game()

**2A**

def ncr(n, r):

    if r > n - r:

        r = n - r

    res = 1

    for i in range(r):

        res \*= (n - i)

        res /= (i + 1)

    return int(res)

def print\_pascal\_triangle(n):

    for i in range(n):

        for j in range(n - i):

            print(" ", end="")

        for k in range(i + 1):

            print(ncr(i, k), end=" ")

        print()

# Input the number of rows

n = int(input("Enter the number of rows: "))

print\_pascal\_triangle(n)

**3A**

def print\_repeated\_elements(nums):

    freq\_dict = {}

    for num in nums:

        if num in freq\_dict:

            freq\_dict[num] += 1

        else:

            freq\_dict[num] = 1

    for num, freq in freq\_dict.items():

        print(f"Element {num} has come {freq} times")

nums = [1,2,3,4,5,2,3,3,4,4,5,4,5,5,6,6,2,3]

print\_repeated\_elements(nums)

**4A**

def read\_matrix\_from\_file(filename):

    with open(filename, 'r') as file:

    return matrix

def add\_matrices(A, B):

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

    return result

def print\_matrix(matrix):

    for row in matrix:

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

# Read matrices A and B from file

A = read\_matrix\_from\_file("matrixA.txt")

B = read\_matrix\_from\_file("matrixB.txt")

# Perform matrix addition

result = add\_matrices(A, B)

# Print the result

print("Matrix A:")

print\_matrix(A)

print("Matrix B:")

print\_matrix(B)

print("Result of A + B:")

print\_matrix(result)

**5A**

class Fraction:

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

        self.numerator = numerator

        self.denominator = denominator

    def \_\_add\_\_(self, other):

        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}"

frac1 = Fraction(1, 4) #1,4 are examples

frac2 = Fraction(1, 8) #1,8 are examples

frac3 = frac1 + frac2

print(frac3)