**UIT 2402 – ADVANCED DATA STRUCTURES AND ALGORITHM ANALYSIS**

**EX 7a: Binomial Coefficient**

**Name:** B.VASUNDHARA

**Roll no:** 3122 21 5002 119

**ALGORITHM:**

* Divide and Conquer divides the problem of size C(n, k), in two sub problems, each of size C(n – 1, k – 1) and C(n – 1, k) respectively. Solution of larger problem is build by adding solution of these two sub problems.
* Structure of binomial coefficient problem using divide and conquer approach is described as :

C(n, k) = C(n-1, k-1) + C(n-1, k)

C(n, 0) = C(n, n) = 1

**Program Code:**

**1) Optimal Substructure:** follows the recursive structure

# A naive recursive Python implementation

def binomialCoeff(n, k):

    if k > n:

        return 0

    if k == 0 or k == n:

        return 1

    # Recursive Call

    return binomialCoeff(n-1, k-1) + binomialCoeff(n-1, k)

# Driver Program to test ht above function

n = 5

k = 2

print ("Value of C(%d,%d) is (%d)" % (n, k,

                                    binomialCoeff(n, k)))

**2) Overlapping Subproblems**

# Returns value of Binomial Coefficient C(n, k)

def binomialCoef(n, k):

    C = [[0 for x in range(k+1)] for x in range(n+1)]

    # Calculate value of Binomial

    # Coefficient in bottom up manner

    for i in range(n+1):

        for j in range(min(i, k)+1):

            # Base Cases

            if j == 0 or j == i:

                C[i][j] = 1

            # Calculate value using

            # previously stored values

            else:

                C[i][j] = C[i-1][j-1] + C[i-1][j]

    return C[n][k]

# Driver program to test above function

n = 5

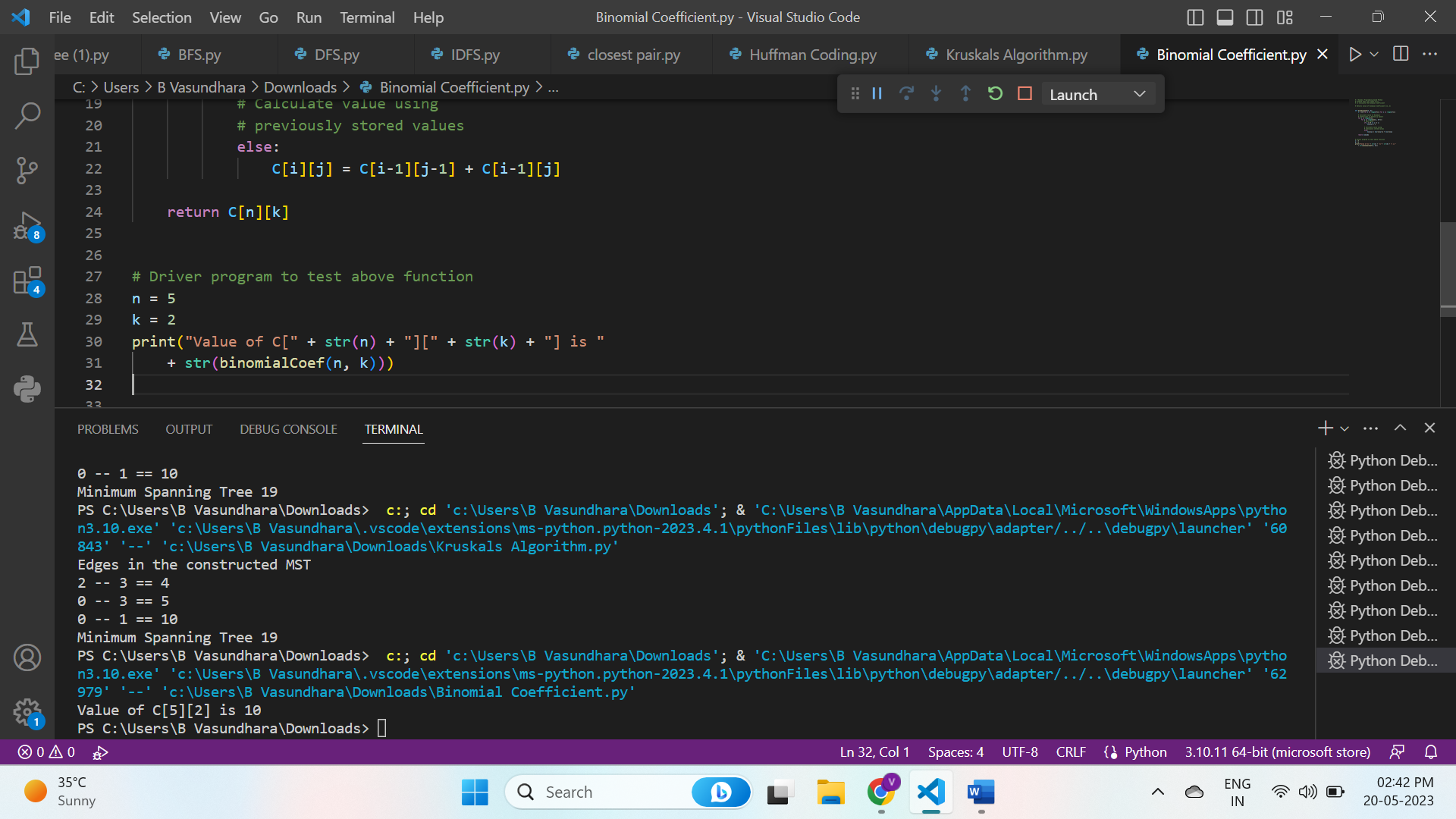
k = 2

print("Value of C[" + str(n) + "][" + str(k) + "] is "

    + str(binomialCoef(n, k)))

**OUTPUT:**

**1) Optimal Substructure:**



**2) Overlapping Subproblems**

A screenshot of a computer

Description automatically generated