Code for Problem 1:

```
print("--- Diptangshu Dey: 20CS8018 ---")

#input the table
r = int(input("Enter number of rows: "))
c = int(input("Enter number of columns: "))

table = []
print("\nTable Entry......")
for a in range(r):
    tmp = []
    print(f"Enter values of row {a}: ")
    tmp = list(map(int, input().split()))

    if len(tmp) != c:
        print("Incorrect number of elements")
        exit(1)

    table.append(tmp)

supply = []
print("\nEnter the supply values:")
supply = list(map(int, input().split()))
if len(supply) != r:
    print("Incorrect number of elements")
    exit(1)

demand = []
print("\nEnter the demand values:")
demand = list(map(int, input().split()))
if len(demand) != c:
    print("Incorrect number of elements")
    exit(1)

# initialise the iterators
i = 0
j = 0
res = 0

# checking for balanced table
def sumOf(ls):
    sum = 0
    for i in ls:
        sum += i
    return sum

if sumOf(supply) == sumOf(demand):
    print("\nThe table is balanced")

    #executing northwest corner rule
    while(i < len(table) and j < len(table[0])):
```

```
            if supply[i] < demand[j]:
                res += supply[i] * table[i][j]

                demand[j] -= supply[i]
                i += 1
            else:
                res += demand[j] * table[i][j]

                supply[i] -= demand[j]
                j += 1

        print(f"The basic feasible solution using Northwest-Corner Method is
    {res}")
    else:
        print("Table is not balanced")

    # Diptangshu Dey: 20CS8018
```

Output:

```
❯ python prob_1.py
--- Diptangshu Dey: 20CS8018 ---
Enter number of rows: 3
Enter number of columns: 4

Table Entry......
Enter values of row 0:
1 2 1 4
Enter values of row 1:
3 3 2 1
Enter values of row 2:
4 2 5 9

Enter the supply values:
30 50 20

Enter the demand values:
20 40 30 10

The table is balanced
The basic feasible solution using Northwest-Corner Method is 310
```

Code for Problem 2:

```
print("--- Diptangshu Dey: 20CS8018 ---")
#input the table
r = int(input("Enter number of rows: "))
c = int(input("Enter number of columns: "))

table = []
print("\nTable Entry......")
for a in range(r):
    tmp = []
    print(f"Enter values of row {a}: ")
    tmp = list(map(int, input().split()))

    if len(tmp) != c:
        print("Incorrect number of elements")
        exit(1)

    table.append(tmp)

supply = []
print("\nEnter the supply values:")
supply = list(map(int, input().split()))
if len(supply) != r:
    print("Incorrect number of elements")
    exit(1)

demand = []
print("\nEnter the demand values:")
demand = list(map(int, input().split()))
if len(demand) != c:
    print("Incorrect number of elements")
    exit(1)

# initialise the iterators
i = 0
j = 0
res = 0

# function definitions
def sumOf(ls):
    sum = 0
    for i in ls:
        sum += i
    return sum

def getMinElem(table, supply, demand):
    m = [99999999999999, 0, 0]
    for i in range(len(table)):
        if supply[i] == 0:
            continue
        else:
```

```
            for j in range(len(table[0])):
                if demand[j] == 0:
                    continue
                else:
                    if (table[i][j] < m[0]):
                        m = [table[i][j], i, j]
                    else:
                        pass
    return m


# checking for balanced table
if sumOf(supply) == sumOf(demand):
    print("\nTable is balanced")
    res = 0
    while(sumOf(supply) != 0 and sumOf(demand) != 0):
        m, i, j = getMinElem(table, supply, demand)
        if supply[i] < demand[j]:
            res += supply[i] * table[i][j]

            demand[j] -= supply[i]
            supply[i] = 0
        else:
            res += demand[j] * table[i][j]

            supply[i] -= demand[j]
            demand[j] = 0

    print(f"\nThe basic feasible solution using Least-Cost Method is
{res}")
else:
    print("table is not balanced")

# Diptangshu Dey: 20CS8018
```

Output:

```
❯ python prob_2.py
--- Diptangshu Dey: 20CS8018 ---
Enter number of rows: 3
Enter number of columns: 4

Table Entry......
Enter values of row 0:
21 16 25 13
Enter values of row 1:
17 18 14 23
Enter values of row 2:
32 27 18 41

Enter the supply values:
11 13 19

Enter the demand values:
6 10 12 15

Table is balanced

The basic feasible solution using Least-Cost Method is 922
```