def uscln(a, b):

    if (b == 0):

        return a

    return uscln(b, a % b)

def Add(x, y):

    max\_len = max(len(x), len(y))

    x = x.zfill(max\_len)

    y = y.zfill(max\_len)

    # initialize the result

    result = ''

    # initialize the carry

    carry = 0

    # Traverse the string

    for i in range(max\_len - 1, -1, -1):

        r = carry

        r += 1 if x[i] == '1' else 0

        r += 1 if y[i] == '1' else 0

        result = ('1' if r % 2 == 1 else '0') + result

        carry = 0 if r < 2 else 1     # Compute the carry.

    if carry != 0:

        result = '1' + result

    return result.zfill(max\_len)

def normaliseString(str1, str2):

    diff = abs((len(str1) - len(str2)))

    if diff != 0:

        if len(str1) < len(str2):

            str1 = ('0' \* diff) + str1

        else:

            str2 = ('0' \* diff) + str2

    return [str1, str2]

def Sub(str1, str2):

    if len(str1) == 0:

        return

    if len(str2) == 0:

        return

    str1, str2 = normaliseString(str1, str2)

    startIdx = 0

    endIdx = len(str1) - 1

    carry = [0] \* len(str1)

    result = ''

    while endIdx >= startIdx:

        x = int(str1[endIdx])

        y = int(str2[endIdx])

        sub = (carry[endIdx] + x) - y

        if sub == -1:

            result += '1'

            carry[endIdx-1] = -1

        elif sub == 1:

            result += '1'

        elif sub == 0:

            result += '0'

        else:

            raise Exception('Error')

        endIdx -= 1

    return result[::-1]

def Mul(binOne, binTwo):

    i = 0

    rem = 0

    sum = []

    bProd = 0

    while binOne != 0 or binTwo != 0:

        sum.insert(i, (((binOne % 10) + (binTwo % 10) + rem) % 2))

        rem = int(((binOne % 10) + (binTwo % 10) + rem) / 2)

        binOne = int(binOne/10)

        binTwo = int(binTwo/10)

        i = i+1

    if rem != 0:

        sum.insert(i, rem)

        i = i+1

    i = i-1

    while i >= 0:

        bProd = (bProd \* 10) + sum[i]

        i = i-1

    return bProd

a = int(input("So nguyen a: "))

b = int(input("So nguyen b: "))

print("chuyen doi so thap phan sang nhi phan")

a1 = bin(a)

b1 = bin(b)

a2 = format(a, "b")

b2 = format(b, "b")

# print(max(len(a1),len(b1)))

print("{0}\t->\t{1}\n{2}\t->\t{3}".format(a, a2, b, b2))

print("Ket qua cua 3 phep tinh:")

print("{0} + {1} = {2}".format(a, b, int(Add(a1, b1), 2)))

# print("{0}\t->\t{1}\n".format(int(binarySubstration(a1,b1),2),binarySubstration(a1,b1)))

print("{0} - {1} = {2}".format(a, b, int(Sub(a2, b2), 2)))

binMul = 0

factr = 1

binOne = int(a2)

binTwo = int(b2)

while binTwo != 0:

    digit = binTwo % 10

    if digit == 1:

        binOne = binOne \* factr

        binMul = Mul(binOne, binMul)

    else:

        binOne = binOne \* factr

    binTwo = int(binTwo/10)

    factr = 10

print("{0} \* {1} = {2}".format(a,b,int(str(binMul),2)))

print("Euclid cua {0} va {1} la {2}".format(a,b,uscln(a,b)))

print("--------------------------------------------------------------------")

conversion\_table = {0: '0', 1: '1', 2: '2', 3: '3',

                    4: '4', 5: '5', 6: '6', 7: '7',

                    8: '8', 9: '9', 10: 'A', 11: 'B',

                    12: 'C', 13: 'D', 14: 'E', 15: 'F'}

def dec2hex(decimal):

    hexadecimal = ''

    while(decimal > 0):

        remainder = decimal % 16

        hexadecimal = conversion\_table[remainder] + hexadecimal

        decimal = decimal // 16

    return hexadecimal

print("2 - Nhi Phan\n8 - Bat Phan\n10 - Thap Phan\n16 - Thap Luc Phan\n")

dec = int(input("So nguyen = "))

coso = int(input("Nhap he co so = "))

if(coso == 2):

    print("{0} -> {1} theo he co so {2}".format(dec, format(dec, "b"), coso))

elif(coso == 8):

    print("%d -> %o theo he co so %d" % (dec, dec, coso))

elif(coso == 10):

    print("{0} -> {1} theo he co so {2}".format(dec, dec, coso))

elif(coso == 16):

    print("{0} -> {1} theo he co so {2}".format(dec, dec2hex(dec), coso))

else:

    print("Sai input")

