**Question:**

**Write the Program for the following and give one example to prove your code is correct:**

1. **Stable Marriage Problem**
2. **Euclid’s Algorithm**
3. **Pigeonhole Principle**
4. **Multiplicative inverse**

**Solution:**

**Stable Marriage Problem**

**Code:**

**import collections**

**preferred\_order\_boy = {**

**'A': ['3', '2', '4', '1'],**

**'B': ['2', '1', '3', '4'],**

**'C': ['3', '1', '4', '2'],**

**'D': ['3', '1', '4', '2']**

**}**

**preferred\_order\_girl = {**

**'1': ['A', 'B', 'D', 'C'],**

**'2': ['C', 'A', 'D', 'B'],**

**'3': ['C', 'B', 'D', 'A'],**

**'4': ['B', 'A', 'C', 'D']**

**}**

**boys\_engage = []**

**girl\_engage = []**

**free\_boy = []**

**free\_girl = []**

**def init\_free\_boy():**

**for boy in preferred\_order\_boy.keys():**

**free\_boy.append(boy)**

**def init\_free\_girl():**

**for girl in preferred\_order\_girl.keys():**

**free\_girl.append(girl)**

**def start\_matching\_boy(boy):**

**for girl in preferred\_order\_boy[boy]:**

**taken\_match = [couple for couple in boys\_engage if girl in couple]**

**if (len(taken\_match) == 0):**

**boys\_engage.append([boy, girl])**

**free\_boy.remove(boy)**

**break**

**elif (len(taken\_match) > 0):**

**current\_guy = preferred\_order\_girl[girl].index(taken\_match[0][0])**

**potential\_guy = preferred\_order\_girl[girl].index(boy)**

**if (current\_guy < potential\_guy):**

**pass**

**else:**

**free\_boy.remove(boy)**

**free\_boy.append(taken\_match[0][0])**

**taken\_match[0][0] = boy**

**break**

**def start\_matching\_girl(girl):**

**for boy in preferred\_order\_girl[girl]:**

**match\_taken = [couple for couple in girl\_engage if boy in couple]**

**if (len(match\_taken) == 0):**

**girl\_engage.append([girl, boy])**

**free\_girl.remove(girl)**

**break**

**elif (len(match\_taken) > 0):**

**current\_girl = preferred\_order\_boy[boy].index(match\_taken[0][0])**

**potential\_girl = preferred\_order\_boy[boy].index(girl)**

**if (current\_girl < potential\_girl):**

**pass**

**else:**

**free\_girl.remove(girl)**

**free\_girl.append(match\_taken[0][0])**

**match\_taken[0][0] = girl**

**break**

**def stable\_matching\_boy():**

**while (len(free\_boy) > 0):**

**for boy in free\_boy:**

**start\_matching\_boy(boy)**

**def stable\_matching\_girl():**

**while (len(free\_girl) > 0):**

**for girl in free\_girl:**

**start\_matching\_girl(girl)**

**def main():**

**print("")**

**stable\_matching\_boy()**

**print(" Boy group is Proposing : ",boys\_engage)**

**print("")**

**stable\_matching\_girl()**

**print(" Girl group is Proposing : ",girl\_engage)**

**init\_free\_boy()**

**print(" Boys : ",free\_boy)**

**init\_free\_girl()**

**print(" Girls : ",free\_girl)**

**print("\n\n")**

**print(" Preferred Order for Boys : \n")**

**for x,y in preferred\_order\_boy.items():**

**print(x,y)**

**print("\n\n Preferred Order for Girls : \n")**

**for x,y in preferred\_order\_girl.items():**

**print(x,y)**

**main()**

**print("")**

**girl\_eng = []**

**for rlp in girl\_engage:**

**rlp.reverse()**

**girl\_eng.append(rlp)**

**girl\_eng.sort()**

**boys\_engage.sort()**

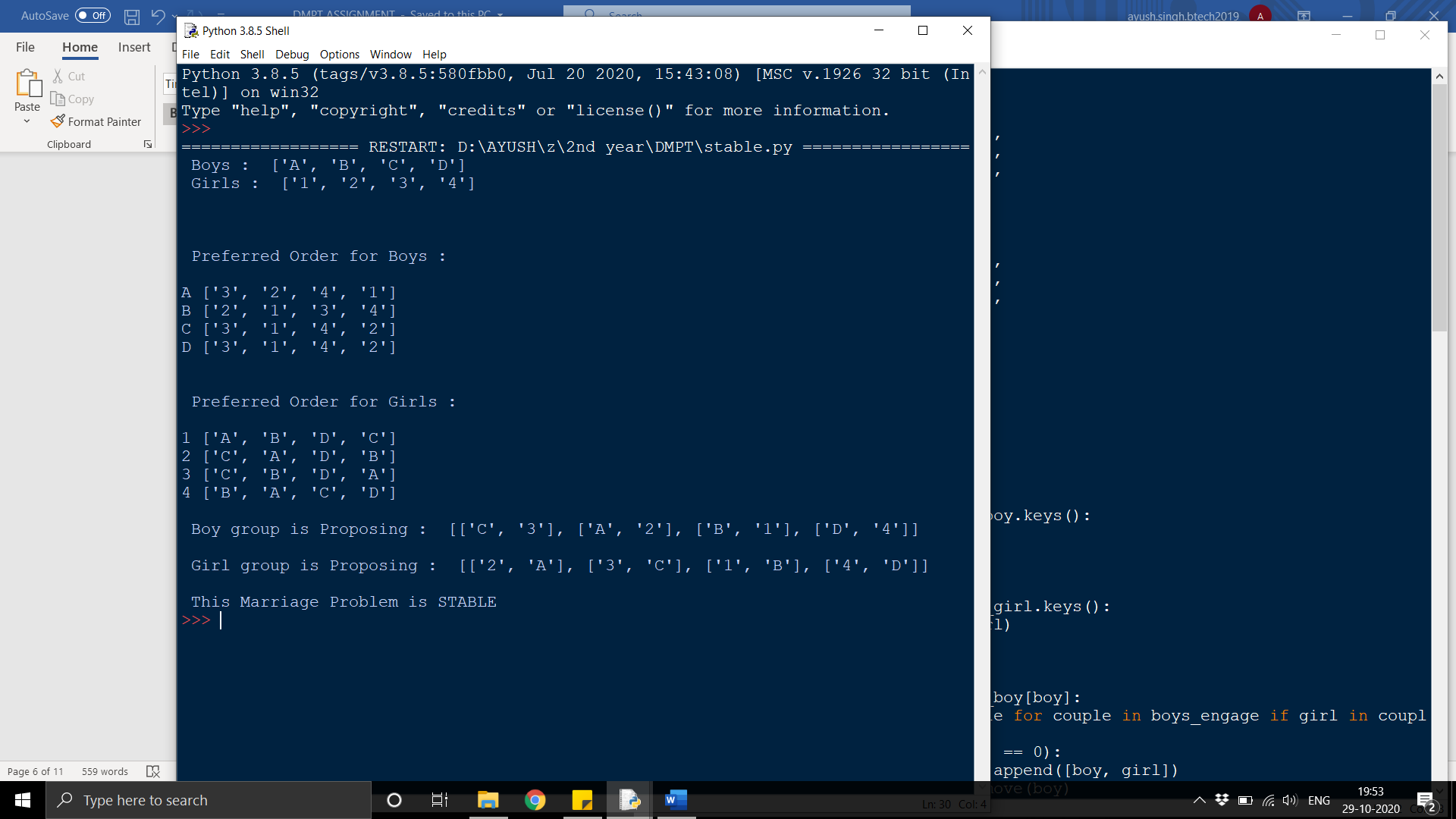
**if(boys\_engage == girl\_eng):**

**print(" This Marriage Problem is STABLE")**

**else:**

**print(" This Marriage Problem is UNSTABLE")**

**Output:**



**Euclid’s Algorithm**

**Code:**

**def euclid():**

**print(" EUCLID's ALGORITHM : \n")**

**num1,num2 = [int(x) for x in input(" Enter 2 numbers for G.C.D : ").split()]**

**m = num1**

**n = num2**

**while(n != 0):**

**div = m//n**

**r = m%n**

**print(" ",m,"=",n,"X",div,"+",r)**

**m = n**

**n = r**

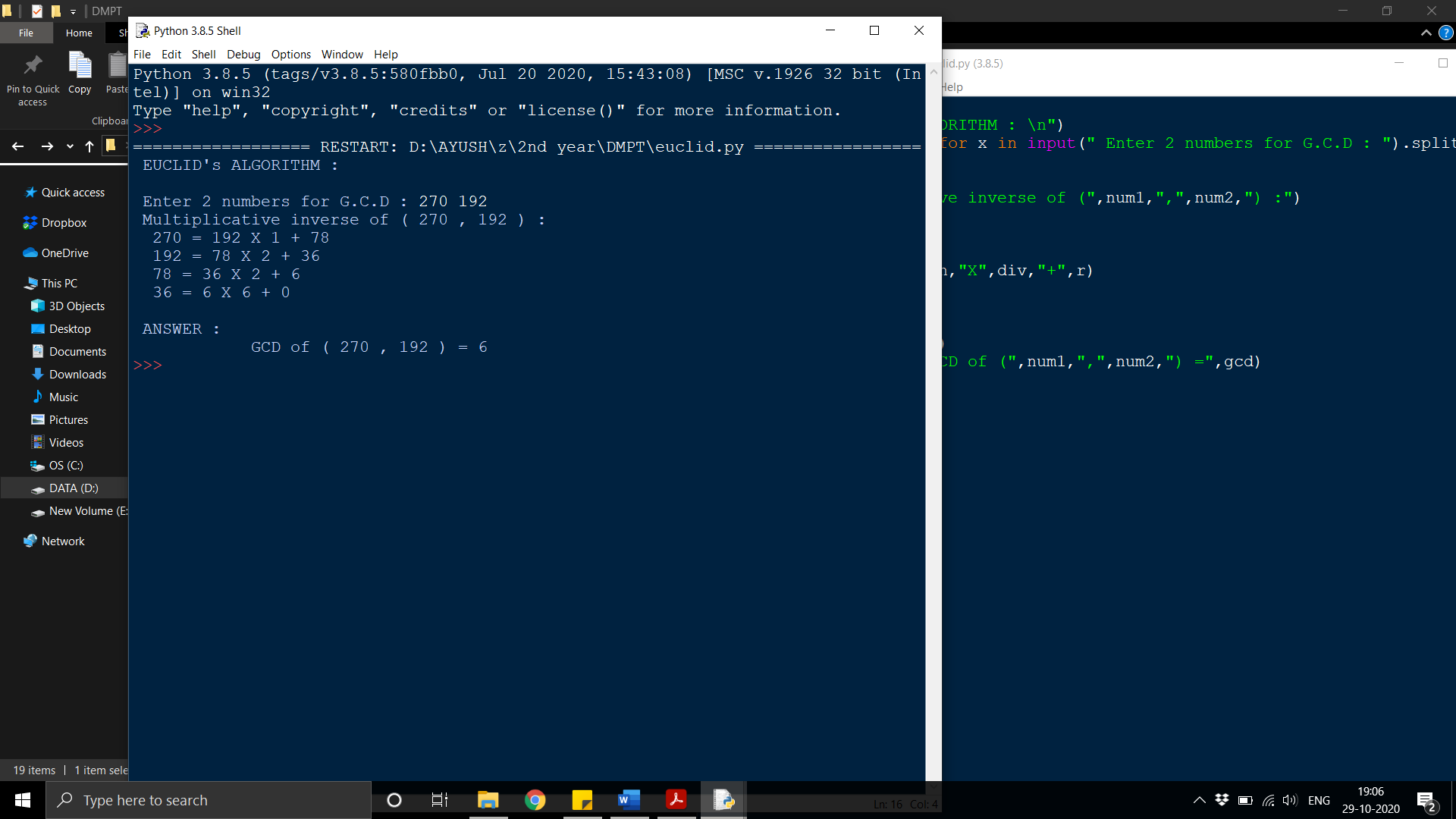
**gcd = m**

**print("\n ANSWER : ")**

**print(" GCD of (",num1,",",num2,") =",gcd)**

**euclid()**

**Output:**



**Pigeonhole Principle**

**Code:**

**def pigeonhole():**

**print(" PIGEONHOLE PRINCIPLE : \n")**

**n = int(input(" Enter total number of PIGEONS : "))**

**m = int(input(" Enter total number of PIGEONHOLES : "))**

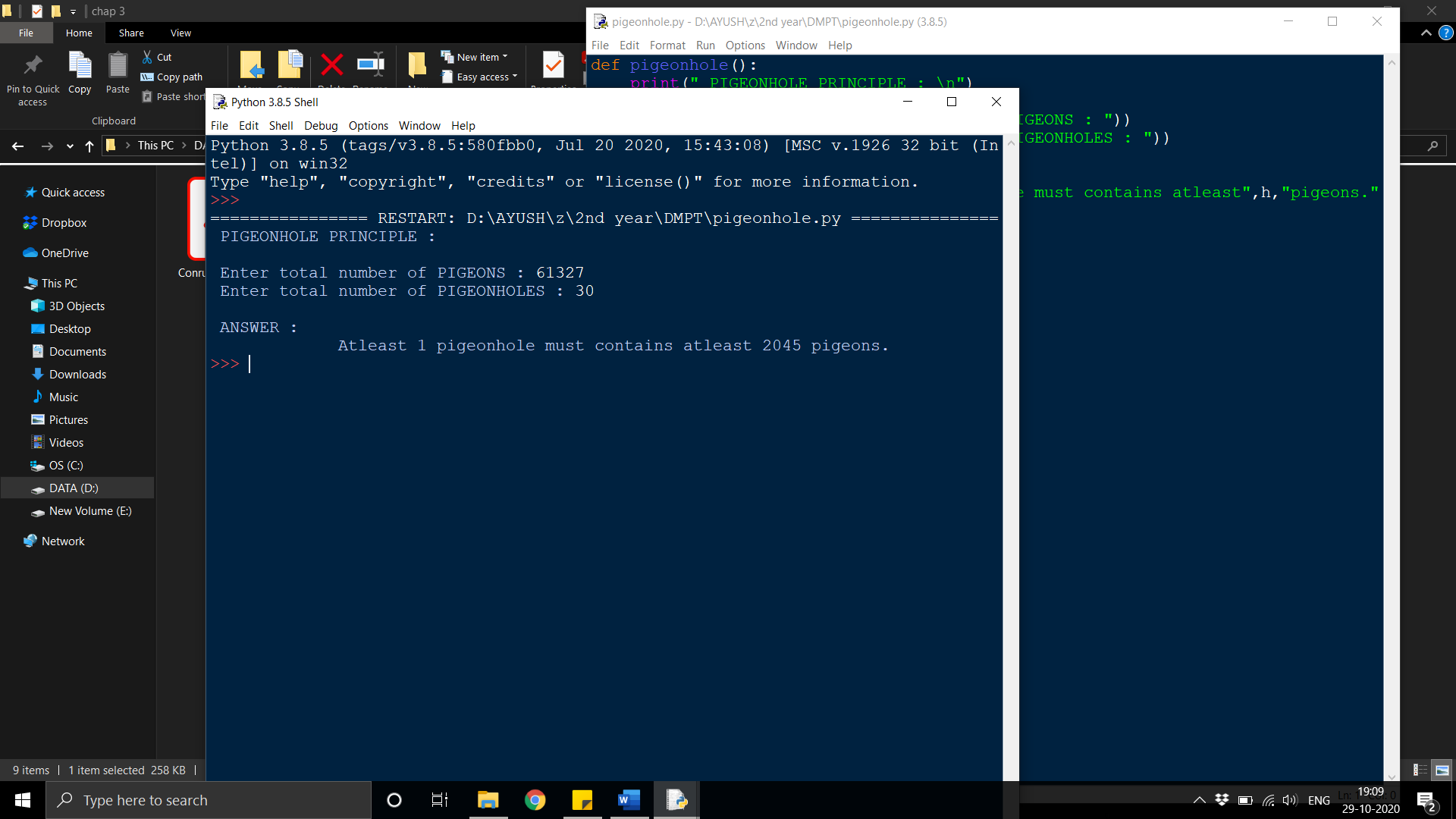
**h = ((n-1)//m)+1**

**print("\n ANSWER : ")**

**print(" Atleast 1 pigeonhole must contains atleast",h,"pigeons.")**

**pigeonhole()**

**Output:**



**Multiplicative Inverse**

**Code:**

**def inverse():**

**print(" MULTIPLICATIVE INVERSE : \n")**

**num1,num2 = [int(x) for x in input(" Enter 2 numbers for multiplicative inverse : ").split()]**

**m = num2**

**n = num1**

**print(" Multiplicative inverse of (",num1,",",num2,") :")**

**print("\n STEP 1 : GCD (",num1,",",num2,") ")**

**while(n != 0):**

**div = m//n**

**r = m%n**

**print(" ",m,"=",n,"X",div,"+",r)**

**m = n**

**n = r**

**gcd = m**

**if(gcd == 1):**

**print(" GCD of (",num1,",",num2,") =",gcd,". Therefore Multiplicative inverse possible\n")**

**print(" STEP 2 : Calculating Multiplicative inverse of ",num1,"(mod",num2,") \n")**

**p = num1**

**q = num2**

**p = p%q**

**for i in range(0,q):**

**if((p\*i)%q == 1):**

**print("\n ANSWER : ")**

**print(" Multiplicative inverse of ",num1,"(mod",num2,") =",i)**

**else:**

**print(" GCD of (",num1,",",num2,") =",gcd,". Therefore Multiplicative inverse not possible\n")**

**inverse()**

**Output:**

