In [4]:
```
#Problem 1: To write a function which inputs a number & prints the mult
iplication table

def mul_table(x,y):
    """
    To calculate the product of two numbers
    """
    for i in range(1,y+1):
        prod=x*i;
        print(x,"X",i,"=",prod);
    return
a=int(input("Enter the number to get the multiplication table for: "))
b=int(input("Enter the last multiplier for the given number: "))
print("The multiplication table for {} till {} is as follows:".format(a
,b));
mul_table(a,b);
```

```
Enter the number to get the multiplication table for: 5
Enter the last multiplier for the given number: 10
The multiplication table for 5 till 10 is as follows:
5 X 1 = 5
5 X 2 = 10
5 X 3 = 15
5 X 4 = 20
5 X 5 = 25
5 X 6 = 30
5 X 7 = 35
5 X 8 = 40
5 X 9 = 45
5 X 10 = 50
```

In [4]:
```
#Problem 2: To print twin primes less than 1000

print("Twin primes less than 1000 are:\n")
def twin_prime(a,b):
    """To check & evaluate if the difference between 2 consecutive odd
```

```
 prime numbers is "2" by using variable "j" & "if" statement"""
    for i in range(a,b):
        j=i+2;
        if(prime_check(j) and prime_check(i)):
            print("(",i,",",j,")",end='')
def prime_check(c):
    """To check if the number obtained as input from the if statemnt of
 twin_prime function is prime or not"""
    for k in range(2,c):
        if c%k==0:
            return False;
    return True;
twin_prime(2,1000);

# reference link: https://www.tutorialspoint.com/How-to-generate-prime-
twins-using-Python
# request you to suggest a way to print the entire thing in one single
 line
```

```
Twin primes less than 1000 are:

( 3 , 5 )( 5 , 7 )( 11 , 13 )( 17 , 19 )( 29 , 31 )( 41 , 43 )( 59 , 61
)( 71 , 73 )( 101 , 103 )( 107 , 109 )( 137 , 139 )( 149 , 151 )( 179 ,
181 )( 191 , 193 )( 197 , 199 )( 227 , 229 )( 239 , 241 )( 269 , 271 )(
281 , 283 )( 311 , 313 )( 347 , 349 )( 419 , 421 )( 431 , 433 )( 461 ,
463 )( 521 , 523 )( 569 , 571 )( 599 , 601 )( 617 , 619 )( 641 , 643 )(
659 , 661 )( 809 , 811 )( 821 , 823 )( 827 , 829 )( 857 , 859 )( 881 ,
883 )
```

In [3]:

```
#Problem3: prime factors

num=int(input("Enter the number to find the prime factors: "));

def p_f(num):
    """
    To check if the number's divisibility by starting off with 2(later
 increasing by one-fold) and producing the output in the form of a list
    """
    prime_fac=[];
    div=2;
    while num>1:
        if num%div==0:
            prime_fac.append(div);
            num/=div;
        else:
            div+=1;
    return prime_fac

p_f(num)

#reference link: https://anh.cs.luc.edu/331/code/factoring.py
```

```
Enter the number to find the prime factors: 56
```

Out[3]:
```
[2, 2, 2, 7]
```

In [17]:
```
#Problem 4: Program to compute Permutations & Combinations
def n_value(n):
    """
    To find n!
    """
    if n==1:
        return n
    else:
        return (n*n_value(n-1))

a=int(input("Enter the value of n:"))
print("n! is:",n_value(a))

def r_value(r):
    """
    To find r!
    """
    if r==1:
        return r;
    else:
        return(r*r_value(r-1))
b=int(input("enter r:"))
print("r! is:",r_value(b));
```

```
def n_diff_r(diff):
    """
    To find (n-r)!
    """
    if diff==1:
        return diff;
    else:
        return(diff*n_diff_r(diff-1))
d=a-b; #equivalent to (n-r)
print("(n-r)! is:",n_diff_r(d));

perm=(n_value(a))/(n_diff_r(d));
comb=perm/(r_value(b));

print("Permutation of {} things taken {} at a time is: {}".format(a,b,p
erm))
print("Combination of {} things taken {} at a time is: {}".format(a,b,c
omb))
```

```
Enter the value of n:5
n! is: 120
enter r:2
r! is: 2
(n-r)! is: 6
Permutation of 5 things taken 2 at a time is: 20.0
Combination of 5 things taken 2 at a time is: 10.0
```

In [10]:

```
#Problem 5: Covert a decimal number to binary number
num=int(input("Enter the decimal number: "))
def conversion(n):
    """ To print the reminder of the number when divided by 2 in revers
e order"""
    if n>1:
        conversion(n//2)
    print(n%2)
    return;
conversion(num)
```

```
#reference link: https://www.tutorialspoint.com/How-to-Convert-Decimal-to-Binary-Using-Recursion-in-Python
```

```
Enter the decimal number: 25
1
1
0
0
1
```

In [31]:

```
#Problem 6: To find sum of cube of individual digits of a number and to
 check if the number is armstrong & print it

num=int(input("Enter the number: "))

def PrintArmstrong(result):

    """ To print the final result if the number is armstrong or not"""

    if result==True:
        print("The given number is an armstrong number.")
    else:
        print("The given number is not an armstrong number.")

def isArmstrong(check):

    """ To check if the number is armstrong or not"""

    if check==num:
        PrintArmstrong(True);
    else:
        PrintArmstrong(False);

def cubesum(num):

    """ To calculate the sum of cubes of individual digits"""

    n=[int(i) for i in str(num)];                                # to conve
```

```
rt digits of an interger to a list
    tot=0;                                                      #to find t
otal which is assigned zero initially
    for j in n:
        tot+=j**3;
    print("Sum of cubes of {} is {}".format(num,tot))
    isArmstrong(tot)
    return
cubesum(num);
```

```
Enter the number: 407
Sum of cubes of 407 is 407
The given number is an armstrong number.
```

In [37]:

```
#Problem 7: To calculate product of digits
num=int(input("Enter the number: "))

def prodDigits(num):

    """ To calculate the product of individual digits"""

    n=[int(i) for i in str(num)];                               # to conve
rt digits of an interger to a list
    prod=1;                                                    #to find pr
oduct which is assigned 1 initially
    for j in n:
        prod*=j;
    print("Product of the digits of {} is: {}".format(num,prod));
    return;
prodDigits(num);
```

```
Enter the number: 555
Product of the digits of 555 is: 125
```

In [2]:

```
#Problem 8: To calculate MDR & MPersistence of a number

num=int(input("Enter the number to find MDR & MP: "));
```

```
def MPersistence(count):
    """To find the multiplicative persistence of a number"""
    print("MPersistence is:",count)

def prodDigits(num):
    """To find product of digits of the number"""
    count=0;
    count+=1;
    MPersistence(count)
    n=[int(i) for i in str(num)];
    prod=1;
    for j in n:
        prod*=j;
    return prod


def MDR(num):
    """To find multiplicative digital root"""
    while num>=10:
        num=prodDigits(num);
    return num;

print("The MDR is",MDR(num));
```

```
Enter the number to find MDR & MP: 77
MPersistence is: 1
MPersistence is: 1
MPersistence is: 1
MPersistence is: 1
The MDR is 8
```

In [6]:

```
#Problem 9: To find the sum of proper divisors of a number

num=int(input("Enter the number: "))

def sumPdivisors(num):
    """ To calculate sum of divisors"""
    tot=0;
    for i in range(1,num):
        if num%i==0:
            tot+=i;
        else:
            continue;
    return tot;

result=sumPdivisors(num);
print("Sum of all the divisors of {} is: {}".format(num,result))
```

```
Enter the number: 77
Sum of all the divisors of 77 is: 19
```

In [100]:
```
#Problem 10: To print all perfect numbers in the given range
ln=int(input("Enter the lower limit number of the range: "))
hn=int(input("Enter the upper limit number of the range: "))
print("The perfect numbers from {} to {} are:".format(ln,hn))

def sumPdivisors(num):
    """ To calculate sum of divisors"""
    tot=0;
    for i in range(1,num):
        if num%i==0:
            tot+=i;
        else:
            continue;
    return tot;

for num in range(ln,hn+1):
    result=sumPdivisors(num);
    if result==num:
        print(num)
```

```
Enter the lower limit number of the range: 1
Enter the upper limit number of the range: 500
The perfect numbers from 1 to 500 are:
6
28
496
```

In [ ]:
```
#Problem 11: To print pairs of amicable numbers in the given range
print("Amicable numbers between 1 & 1000 are:")
def sumPdivisors(a,b):
    """ To calculate sum of divisors for both the numbers and to return
 a boolean value"""
    tot1=0;
    tot2=0;
    for i in range(1,a):
        if a%i==0:
            tot1+=i;
        else:
            continue;
    for j in range(1,b):
        if b%j==0:
            tot2+=j;
        else:
            continue;
    if tot1==b and tot2==a:
        return True;
    else:
        return False;

for a in range(1,1000):
    for b in range(1,1000):
        if sumPdivisors(a,b)==True and a!=b:
            s=str(a)+" "+str(b)
            print(s.split())

#reference link: https://stackoverflow.com/questions/38094818/what-is-t
he-most-efficient-way-to-find-amicable-numbers-in-python
```

```
Amicable numbers between 1 & 1000 are:
['220', '284']
['284', '220']
```

In [2]:
```
#Problem 12: To filter odd numbers in the list for a given range

a=int(input("Enter the lower limit number of the range: "))
b=int(input("Enter the upper limit number of the range: "))
```

```
num_list=range(a,b+1);
print("List without filter=",list(num_list));
oddnum_list=filter(lambda x:(x%2==1),num_list);
print("Odd number list=",list(oddnum_list));
```

```
Enter the lower limit number of the range: 1
Enter the upper limit number of the range: 20
List without filter= [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 1
5, 16, 17, 18, 19, 20]
Odd number list= [1, 3, 5, 7, 9, 11, 13, 15, 17, 19]
```

In [3]:
```
#Problem 13: To find cube of elements in the list for a given range

a=int(input("Enter the lower limit number of the range: "))
b=int(input("Enter the upper limit number of the range: "))
num_list=range(a,b+1);
print("Original list=",list(num_list));
cube_list=map(lambda x:x**3,num_list)
print("Cube of all elements=",list(cube_list));
```

```
Enter the lower limit number of the range: 1
Enter the upper limit number of the range: 5
Original list= [1, 2, 3, 4, 5]
Cube of all elements= [1, 8, 27, 64, 125]
```

In [15]:
```
#Problem 14: To find cube of even elements in the list for a given rang
e

a=int(input("Enter the lower limit number of the range: "))
b=int(input("Enter the upper limit number of the range: "))
num_list=range(a,b+1);
print("Original List =",list(num_list));
evennum_list=filter(lambda x:x%2==0,num_list);
cube_list=map(lambda i:i**3,evennum_list)
print("Cube of even numbers=",list(cube_list));
```

```
Enter the lower limit number of the range: 1
Enter the upper limit number of the range: 10
```

```
Original List = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
Cube of even numbers= [8, 64, 216, 512, 1000]
```