In [1]:

```
#print  the multiplication table of that number
def table(num):
    for i in range(1,13):
        print(num,'X',i,'=', num*i)


table(13)
```

```
13 X 1 = 13
13 X 2 = 26
13 X 3 = 39
13 X 4 = 52
13 X 5 = 65
13 X 6 = 78
13 X 7 = 91
13 X 8 = 104
13 X 9 = 117
13 X 10 = 130
13 X 11 = 143
13 X 12 = 156
```

In [4]:

```
#Write a program to print twin primes less than 1000.
def primeornot(num):
   for i in range(2, num):
      if num % i == 0:
         return False
   return True

def twin_primes(start, end):
   for i in range(start, end):
      j = i + 2
      if(primeornot(i) and primeornot(j)):
         print(" {:d},{:d}".format(i, j))

twin_primes(1, 1000)
```

```
 1,3
 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 [5]:

```
# decimal number to binary number
def DecimalToBinary(num):
        if num > 1:
            DecimalToBinary(num // 2)
            print (num % 2)

DecimalToBinary(110)
```

```
1
0
1
1
1
0
```

In [14]:

```
#cubesum
num=int(input())
sum=0
while(num>0):
    sum=sum+(num%10)*(num%10)*(num%10)
    num=num//10
print('sum is',sum)
def isArmstrong(num):
    if sum==num:
        print('it is an armstrong no')
    else:
        print('its not')
isArmstrong(153)
```

```
153
sum is 153
it is an armstrong no
```

In [9]:

```
#product of digits
import numpy as np
def prodDigits(num):
    num1 = str(num)
    list_of_number = list(map(int, num1.strip()))
    return np.prod(list_of_number)

num= 12234
print(prodDigits(num))
```

```
48
```

In [10]:

```
#MDR
import numpy as np
def MDR(num):
        num1 = str(num)
        a = list(map(int, num1.strip()))
        return np.prod(a)
num=1234
while num>10:
    num=MDR(num)
print(num)
```

```
8
```

In [11]:

```
#sum of proper divisors of a number
def sumPdivisors(num):
    sum=0
    for i in range(1,num):
        if num%i==0:
            print(i)
            sum+=i
    return sum
sumPdivisors(36)
```

```
1
2
3
4
6
9
12
18
```

Out[11]:

```
55
```

In [15]:

```
#perfect nos in a range
lower=int(input('enter lowest range'))
upper=int(input('enter upper range'))

for i in range(lower,upper+1):
    sum=0
    for num in range(1, i):
        if(i % num == 0):
            sum = sum + num
            if(sum == i):
                print(i)
```

```
enter lowest range1
enter upper range1000
6
24
28
496
```

In [22]:

```
#amicable
def sumPdivisors(num):
    sum=0
    for i in range(1,num):
        if num%i==0:

            sum+=i
    return sum
sumPdivisors(36)
def isamicable(i,j):
    if sumPdivisors(i)==j and sumPdivisors(j)==i:
        print('they R amicable')
    else:
        print('they R not')
isamicable(220,284)
```

```
they R amicable
```

In [23]:

```
#filtering odd no using filter()
l1=range(1,100)
def oddno(l1):
    if (l1%2)!=0:
        return l1
odd_num=list(filter(oddno,l1))
print(odd_num)
```

```
[1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25, 27, 29, 31, 33, 35, 37, 3
9, 41, 43, 45, 47, 49, 51, 53, 55, 57, 59, 61, 63, 65, 67, 69, 71, 73, 75,
77, 79, 81, 83, 85, 87, 89, 91, 93, 95, 97, 99]
```

In [24]:

```
#map() to make a list whose elements are cube of elements
l2=range(1,10)
def cube(l2):
    return l2**3
cube_list=list(map(cube,l2))
print(cube_list)
```

```
[1, 8, 27, 64, 125, 216, 343, 512, 729]
```

In [25]:

```
#map() and filter() to make a list whose elements are cube of even number in a given li
st
num=range(1,50)
def even(num):
    if num%2==0:
        return num
def cube(num):
    return num**3
even_list=list(filter(even,num))
cube_list=list(map(cube,even_list))
print(cube_list)
```

```
[8, 64, 216, 512, 1000, 1728, 2744, 4096, 5832, 8000, 10648, 13824, 17576,
21952, 27000, 32768, 39304, 46656, 54872, 64000, 74088, 85184, 97336, 1105
92]
```

In [ ]:

```

```