In [1]:
```
#1.Program that prints table of a number
n = int(input("enter the number "))
for i in range(1,11):
    print(n,"*",i,"=",n*i)
```

```
enter the number 8
8 * 1 = 8
8 * 2 = 16
8 * 3 = 24
8 * 4 = 32
8 * 5 = 40
8 * 6 = 48
8 * 7 = 56
8 * 8 = 64
8 * 9 = 72
8 * 10 = 80
```

In [2]:
```
#2.Program to print twin primes less than 1000
n = int(input("enter the number "))
l = []
for i in range(2, n):
    f = 0
    for j in range(2,i):
        if i%j == 0:
            f += 1
    if f == 0:
        l.append(i)
#print(l)
t = []
for i in l:
    for j in l:
        if j-i == 2:
            t.append((i, j))
            #t.append(j)
print(t)
```

```
enter the number 100
[(3, 5), (5, 7), (11, 13), (17, 19), (29, 31), (41, 43), (59, 61), (71,
73)]
```

In [3]:
```python
#3.Program to find prime factors of a number
n = int(input("enter the number "))
m = []
l = []
for i in range(2, n+1):
    if n % i == 0:
        m.append(i)
#print(m)
def isprime(m):
    for i in m:
        f = 0
        for j in range(2,i):
            if i%j == 0:
                f += 1
        if f == 0:
            l.append(i)
    return l
a = isprime(m)
print(a)
```

```
enter the number 58
[2, 29]
```

In [4]:
```python
#4.Program on permutations and combinations
fact = lambda x : 1 if x == 1 else x * fact(x-1)
def task(m):
    for i in range(1,m+1):
        fact(i)
    return(fact(i))
n, r = map(int, input("enter the values of n & r").split())
if n < r:
    print("n can't be less than r")
else:
    print("Permutations of "+str(n)+" objects taken "+str(r)+" at a tim
e is", task(n)/task(n-r))
```

```
    print("Combinations of "+str(n)+" objects taken "+str(r)+" at a tim
e is", task(n)/(task(n-r)*task(r)))
```

```
enter the values of n & r15 10
Permutations of 15 objects taken 10 at a time is 10897286400.0
Combinations of 15 objects taken 10 at a time is 3003.0
```

In [5]:
```
#5.Program to convert decimal number to binary number
"""Used this link as reference for concept: http://cs.furman.edu/digita
ldomain/more/ch6/dec_frac_to_bin.htm"""
def float_bin(number, places = 3):
    whole, dec = str(number).split(".")
    whole = int(whole)
    dec = int(dec)
    res = bin(whole).lstrip("0b")+"."
    for x in range(places):
        whole, dec = str((decimal_converter(dec))*2).split(".")
        dec = int(dec)
        res += whole
    return res
def decimal_converter(num):
    while num>1:
        num /= 10
    return num
n = input("enter the floating number ")
p = int(input("enter the number of decimal places of result "))
print(float_bin(n, places  = p))
```

```
enter the floating number 10.625
enter the number of decimal places of result 3
1010.101
```

In [6]:
```
#6.Program to find cube of given digits and isArmstrong(), PrintArmstro
ng()
m = int(input("enter a number "))
n = list(map(int, str(m)))
def cubesum(m):
    return (sum([i**3 for i in m]))
def isArmstrong(a):
    if cubesum(a) == m:
        print("Yes it is armstrong number")
    else:
        print("It is not armstrong number")
isArmstrong(n)
def printArmstrong(b):
    if cubesum(b) == m:
        print("Armstrong number is", m)
    else:
        pass
printArmstrong(n)
```

```
enter a number 158
It is not armstrong number
```

In [7]:
```
#7.Program that returns product of digits of a number
n = input("enter a number ")
def prodDigits(a):
    l = list(a)
    p=1
    for i in l:
        p=p*int(i)
    return p
print(prodDigits(n))
```

```
enter a number 128
16
```

In [8]:
```
#8.Program on MDR() and MPersistence using proDigits()
x = input("enter a number ")
def MDR(q):
    while int(q)>9:
        s = prodDigits(str(q))
        q=s
    return s
print("MDR is ",MDR(x))
def MPersistence(q):
    a=0
    while int(q)>9:
        s = prodDigits(str(q))
        q=s
        a+=1
    return a
print("MPersistence is ", MPersistence(x))
```

```
enter a number 341
MDR is  2
MPersistence is  2
```

In [9]:
```
#9.Sum of proper divisors of a given number
def sumPdivisors(n=int(input("enter the number "))):
    l = []
    for i in range(1,n):
        if n%i == 0:
            l.append(i)
    return sum(l)
print(sumPdivisors())
```

```
enter the number 143
25
```

In [10]:
```
#10.Program to print perfect number
for i in range(2, int(input("enter the range "))):
    s = sumPdivisors(i)
    if s == i:
        print(i)
```

```
enter the range 300
6
28
```

In [11]:
```
#11.Program to print amicable numbers in a range
d = dict()
z, c = [], []
for i in range(2, int(input("enter the range "))):
    s = sumPdivisors(i)
    d[i]=s
#print(d)
```

```
for i, j in d.items():
    for a, b in d.items():
        if i == b and j == a:
            if i!= a:
                z.append(sorted([i,a]))
for i in z:
    if i not in c:
        c.append(i)
print(c)
```

```
enter the range 300
[[220, 284]]
```

In [12]:
```
#12.Program to find odd numbers in a list using filter function
print(list(filter(lambda x : x%2==1, [i for i in range(int(input("enter
 the range ")))])))
```

```
enter the range 100
[1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25, 27, 29, 31, 33, 35, 37,
39, 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 [13]:
```
#13.Program to print cube elements of a given list
print(list(map(lambda x : x**3, [i for i in range(10)])))
```

```
[0, 1, 8, 27, 64, 125, 216, 343, 512, 729]
```

In [14]:
```
#14.Program using map() and filter()
print(list(map(lambda x : x**3, list(filter(lambda x: x%2==0, [i for i
 in range(100)])))))
```

```
[0, 8, 64, 216, 512, 1000, 1728, 2744, 4096, 5832, 8000, 10648, 13824,
17576, 21952, 27000, 32768, 39304, 46656, 54872, 64000, 74088, 85184, 9
7336, 110592, 125000, 140608, 157464, 175616, 195112, 216000, 238328, 2
62144, 287496, 314432, 343000, 373248, 405224, 438976, 474552, 512000,
551368, 592704, 636056, 681472, 729000, 778688, 830584, 884736, 941192]
```

In [15]:
```
#7.Alternative program for 7
```

```python
from functools import reduce
n = map(int, input("enter a number "))
def prodDigits(a):
    return (reduce(lambda x, y: int(x)*int(y), a)) #Learnt from the lec
ture provided.
p = prodDigits(n)
print(p)
```

```
enter a number 25
10
```

In [ ]: