In [2]:
```
# 1.Program to print the multiplication table of the entered integer

import sys #imported for exception handling

#function to print the multiplication table
def multiply(par):
    for i in range(10):
        print("{0} * {2} = {1}".format(par,par*(i+1),(i+1)))

#Below snippet takes user input and checks if the input is valid intege
r and then calls the function
try:
    Input=int(input('Enter an integer:'))
    multiply(Input)
except ValueError:
    print("The entered input is not an integer. Please try again")
```

```
Enter an integer:e
The entered input is not an integer. Please try again
```

In [64]:
```
# 2.Program to print twin primes below 1000

#Function to determine if a number is odd or not
def odd(temp):
    return False if (temp%2==0) else True

#Function to determine if a number is prime or not
#Returns True if the number is prime and false it isn't
def prime(temp):
    if temp>1:
        flag=0
        for i in range(2,temp):
            if temp%i==0:
                flag=flag+1
            else:
                flag
```

```
        return False if flag>1 else True
    else:
        return True

num_list = list(range(1,1000)) #Creates a list with number from 1 to 10
00
#print(num_list)

sort_list=sorted(list(filter(odd,num_list)),reverse=False) # Filters od
d numbers and sorts the list in ascending order
#print(sort_list[1])

for indx,val in enumerate(sort_list):
    #Boolean condition to check if consecutive odd numbers are prime or
 not
    if prime(sort_list[indx]) and prime(sort_list[indx+1]):
        print(sort_list[indx],sort_list[indx+1])
```

```
1 3
3 5
5 7
7 9
9 11
11 13
17 19
23 25
29 31
41 43
47 49
59 61
71 73
101 103
107 109
137 139
149 151
167 169
179 181
191 193
197 199
227 229
```

```
239 241
269 271
281 283
311 313
347 349
359 361
419 421
431 433
461 463
521 523
569 571
599 601
617 619
641 643
659 661
809 811
821 823
827 829
839 841
857 859
881 883
```

In [63]:

```python
# 3.Print prime factors of a number

#Function to find the factors of n
def factors(temp,Input):
    List=[]
    for i in range(temp,Input+1):
        while Input%i == 0:
            List.append(i)
            Input=Input//i
    return List

Input=int(input('Enter a number:')) #User input
result=factors(2,Input)
print(result)
```

```
Enter a number:315
[3, 3, 5, 7]
```

In [91]:
```
# 4.Program to print permutations and combinations of the given numbers

def factorial(num):
    result=1
    for i in range(num,1,-1):
        result=result*i
        num=num-1
    return result

n=int(input('Enter number of objects:')) #User input
r=int(input('Enter number of selections:')) #User input
npr=int(factorial(n)/factorial(n-r))

print('No. of Permutations of {} objects taken {} at a time:{}'.format(
n,r,npr))
print('No. of Combinations of {} objects taken {} at a time:'.format(n,
r),int((npr)/factorial(r)))
```

```
Enter number of objects:4
Enter number of selections:3
No. of Permutations of 4 objects taken 3 at a time:24
No. of Combinations of 4 objects taken 3 at a time: 4
```

In [107]:
```
# 5.Program to convert decimal to binary

#function to convert decimal to binary list
def dtob(num):
    binary=[]
    while num>1:
        binary.append(int(num%2))
        num=num/2
    binary.reverse() #reverses the elements of the list
    return binary

num=int(input('Enter a number:')) #User input
List=dtob(num)
```

```
#Convert list to string
print(''.join(str(i) for i in List))
```

```
Enter a number:44
101100
```

In [12]:

```
# 6.Program to calculate cubesum and find armstrong number

#function to calculate the cubesum of the entered number
def cubesum(val):
    result=0
    o_val=val
    while val>0:
        result=(val%10)**3+result
        val=int(val/10)
    isArmstrong(o_val,result) #function call for isArmstrong
    PrintArmstrong(o_val,result) #function call for PrintArmstrong
    return result

#function to check if the entered value and the cubesum result are equa
l
def isArmstrong(actual_val,cubesum_val):
    if actual_val == cubesum_val:
        print('The number {} is an armstrong number'.format(actual_val
))
    else:
        print('The number {} is not an armstrong number'.format(actual_
val))

#function to print armstrong number
def PrintArmstrong(actual_val,cubesum_val):
    print('The cubesum of the number {} is {}'.format(actual_val,cubesu
m_val))

Input=int(input('Enter a number:'))
cubesum(Input)
```

```
Enter a number:371
The number 371 is an armstrong number
```

```
The cubesum of the number 371 is 371
```

Out[12]:
```
371
```

In [10]:
```python
# 7.program to calculate product of entered digits

def prodDigits(val):
    result=1
    while val>0:
        result=(result*(val%10))
        val=int(val/10)
    return result

val=int(input('Enter a number:'))
print(prodDigits(val))
```

```
Enter a number:543
60
```

In [9]:
```python
# 8.Programt to find the MDR and MPersistence

#function to calculate the product of digits of a number
def prodDigits(val):
    result=1
    while val>0:
        result=(result*(val%10))
        val=int(val/10)
    return result

#function to loop MDR calculation
def MDR(val):
    while val>9:
        result=prodDigits(val)
        val=result
    return result

#function to calculate the number of iterations
def MPersistence(val):
    counter=0
```

```
    while val>9:
        result=prodDigits(val)
        val=result
        counter+=1
    return counter


Input=int(input('Enter a number:')) #user input
print('MDR:{}'.format(MDR(Input)))
print('MPersistence:{}'.format(MPersistence(Input)))
```

```
Enter a number:341
MDR:2
MPersistence:2
```

In [32]:

```
# 9. Program to print sum of proper divisors of a number

#function calculates the sum of proper divisors of the entered value an
d returns the result
def sumPdivisors(val):
    result=0
    for i in range(1,val):
        if val%i == 0:
            result+=i
    return result

Input=int(input('Enter a number:')) #user input
print('Sum of all proper divisors of {} is:'.format(Input),sumPdivisors
(Input))
```

```
Enter a number:36
Sum of all proper divisors of 36 is: 55
```

In [34]:

```
# 10.Program to print all perfect number in a range

#function prints all the values that are perfect numbers
def sumPdivisors(start,end):
    for i in range(start,end):
        result=0
```

```
        for j in range(1,i):
            if i%j == 0:
                result+=j
        if i==result:
            print(i)

#User input
start=int(input('Enter start of the range:'))
end=int(input('Enter end of the range:'))

print('Below are the perfect numbers between {} and {}:'.format(start,e
nd))
sumPdivisors(start,end) #function call
```

```
Enter start of the range:1
Enter end of the range:1000
Below are the perfect numbers between 1 and 1000:
6
28
496
```

In [60]:

```
# 11.Program to print amicable numbers in a range

#function creates a dictionary of all the values that are not perfect n
umbers
def sumPdivisors(start,end):
    dic={}
    for i in range(start,end):
        result=0
        for j in range(1,i):
            if i%j == 0:
                result+=j
        if i!=result:
            dic[i]=result
    return dic

#function creates a copy of the above created dictionary and compares b
oth the dictionaries for matching values
def amicable(dic):
```

```
    dic1=dic.copy()
    for k1,v1 in dic.items():
        for k2,v2 in dic1.items():
            if k1==v2 and k2==v1:
                print(k1,v1)
                dic1[k1]=0

#user input for start and end ranges
start=int(input('Enter start of the range:'))
end=int(input('Enter end of the range:'))

dic=sumPdivisors(start,end) #stores all the numbers that are not perfec
t in a dictionary, 'dic'
amicable(dic) #function call
```

```
Enter start of the range:1
Enter end of the range:1000
220 284
```

In [70]:

```
# 12.program to find odd numbers in a list using filter function

#Function to determine if a number is odd or not
def odd(temp):
    return False if (temp%2==0) else True

Input=[int(i) for i in input('Enter comma seperated numbers as input').
split(',')]
print(list(filter(odd,Input)))
```

```
Enter comma seperated number as input1,2,23,23,24,5,345,346,457,567,563
[1, 23, 23, 5, 345, 457, 567, 563]
```

In [71]:

```
# 13.program to print the cubes of elements in a given list

def cube(val):
    return val**3

Input=[int(i) for i in input('Enter comma seperated numbers as input').
```

```
split(',')]
print(list(map(cube,Input)))
```

```
Enter comma seperated number as input1,2,3,4,5,6,7,8
[1, 8, 27, 64, 125, 216, 343, 512]
```

In [74]:
```
# 14.Program to print the cubes of even numbers in a given list

def cube(val):
    return val**3

def even(val):
    return True if val%2==0 else False

Input=[int(i) for i in input('Enter comma seperated numbers as input').
split(',')]
print(list(map(cube,filter(even,Input))))
```

```
Enter comma seperated numbers as input1,2,3,4,5,6,7,8,9,10
[8, 64, 216, 512, 1000]
```

In [ ]:
```

```