1. **Write a function that inputs a number and prints the multiplication table of that number**

In [19]:

```
def table(num):
    for i in range(1,11):
        b=num*i
        print(num,"*",i,"=",b)
c = int(input("Please enter a number "))
table(c)
```

```
Please enter a number 20
20 * 1 = 20
20 * 2 = 40
20 * 3 = 60
20 * 4 = 80
20 * 5 = 100
20 * 6 = 120
20 * 7 = 140
20 * 8 = 160
20 * 9 = 180
20 * 10 = 200
```

In [ ]:

```
def table(num):
    lst = [*range(num,num*11,num)]

    print(lst)

b = int(input("please enter a number : "))
table(b)
```

```
please enter a number : 10
[10, 20, 30, 40, 50, 60, 70, 80, 90, 100]
```

2.Write a program to print twin primes less than 1000. If two consecutive odd numbers are both prime then they are known as twin primes

In [ ]:

```
def prime(num):
  #findind the prime numbers
    lst =[]
    for num in range (2,1001):
        for i in range(2,num):
            if (num%i) == 0:
                break
        else:
            lst.append(num)
    list1=[]
    list1 += lst
    list2 = []
    #finding the twin prime
    for i in range(0,len(list1) - 1):
        if(list1[i+1]-list1[i])== 2:
            list2.append((list1[i],list1[i+1]))

    return(list2)

prime(1000)
```

Out[ ]:

```
[(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)]
```

3.Write a program to find out the prime factors of a number. Example: prime factors of 56 - 2, 2, 2, 7

In [ ]:

```
def factor(num):
    #import pdb;
    lst = []
    while num%2==0:
        lst.append(2)
        #pdb.set_trace()
        num = num/2
    for i in range(3,10,2):
        while num%i==0:
            lst.append(i)
            num=num/i
            #pdb.set_trace()
    print(lst)
factor(56)
```

```
[2, 2, 2, 7]
```

4.Write a program to implement these formulae of permutations and combinations Number of permutations of n objects taken r at a time: p(n, r) = n! / (n-r)!. Number of combinations of n objects taken r at a time is: c(n, r) = n! / (r!*(n-r)!) = p(n,r) / r!

In [ ]:

```
def factorial(num):
  return 1 if (num==1)else (num*factorial(num-1))

def permutation(n,r):
  c = factorial(n)/factorial((n-r))
  return c


def combination(n,r):
  d=permutation(n,r)/factorial(r)
  return d
```

```
n=int(input("Enter total number of objects: "))
r=int(input("Enter number of objects that has to be selected: "))

print("Number of permutations of {0} objects taken {1} at a time: {2} ".format(n,r,permuta
tion(n,r)))
print("Number of combinations of {0} objects taken {1} at a time is: {2} ".format(n,r,comb
ination(n,r)))
```

```
Enter total number of objects: 10
Enter number of objects that has to be selected: 2
Number of permutations of 10 objects taken 2 at a time: 90.0
Number of combinations of 10 objects taken 2 at a time is: 45.0
```

5.Write a function that converts a decimal number to binary number

In [ ]:

```
def decimal_to_binary(n):
  a=[]
  while(n>1):
    a.append(n%2)
    n=n//2
  a.append(1)
  return a[::-1]
decimal_to_binary(17)
```

Out[ ]:

```
[1, 0, 0, 0, 1]
```

6.Write a function cubesum() that accepts an integer and returns the sum of the cubes of individual digits of that number. Use this function to make functions PrintArmstrong() and isArmstrong() to print Armstrong numbers and to find whether is an Armstrong number.

In [17]:

```
def cubesum(n):
  sum=0
  while n>0:
    sum+=(n%10)**3
    n=n//10
  return sum

def PrintArmstrong(n):
  for i in range(1,n):
    if (cubesum(i)==i):
      print(i)

def IsArmstrong(n):
  if n==cubesum(n):
    print("{} is an Armstrong numbers".format(n))
  else:
    print("{} is not an Armstrong numbers".format(n))

#z=int(input("Please enter a number : "))

PrintArmstrong(1000)
IsArmstrong(407)
IsArmstrong(385)
```

```
1
153
370
371
407
407 is an Armstrong numbers
385 is not an Armstrong numbers
```

7.Write a function prodDigits() that inputs a number and returns the product of digits of that number.

In [ ]:

```
def proddigit(n):
  prod=1
  while(n!=0):
    prod=prod*(n%10)
    n=n//10
  return prod
proddigit(96)
```

Out[ ]:

```
54
```

8.If all digits of a number n are multiplied by each other repeating with the product, the one digit number obtained at last is called the multiplicative digital root of n. The number of times digits need to be multiplied to reach one digit is called the multiplicative persistance of n. Example: 86 -> 48 -> 32 -> 6 (MDR 6, MPersistence 3) 341 -> 12->2 (MDR 2, MPersistence 2) Using the function prodDigits() of previous exercise write functions MDR() and MPersistence() that input a number and return its multiplicative digital root and multiplicative persistence respectively

In [ ]:

```
def mdr(n):
  i=0
  while(n>9):
    n = proddigit(n)
    i=i+1
  print("MDR---",n)
  print("MPersistence--",i)

mdr(341)
```

```
MDR--- 2
MPersistence-- 2
```

9.Write a function sumPdivisors() that finds the sum of proper divisors of a number. Proper divisors of a number are those numbers by which the number is divisible, except the number itself. For example proper divisors of 36 are 1, 2, 3, 4, 6, 9, 18

In [ ]:

```
def divisor(n):
  m=[]
  for i in range(1,n):
    if (n%i==0):
      m.append(i)
  return m
divisor(36)
```

Out[ ]:

```
[1, 2, 3, 4, 6, 9, 12, 18]
```

10.A number is called perfect if the sum of proper divisors of that number is equal to the number. For example 28 is perfect number, since 1+2+4+7+14=28. Write a program to print all the perfect numbers in a given range

In [ ]:

```
def divisor(n):
  m=[]
  for i in range(1,n):
    if (n%i==0):
      m.append(i)
  return sum(m)

def perfect_number(n):
  for i in range(1,n+1):
    if (divisor(i)==i):
        print(i)

perfect_number(100)
```

```
6
28
```

11.Two different numbers are called amicable numbers if the sum of the proper divisors of each is equal to the other number. For example 220 and 284 are amicable numbers. Sum of proper divisors of 220 = 1+2+4+5+10+11+20+22+44+55+110 = 284 Sum of proper divisors of 284 = 1+2+4+71+142 = 220 Write a function to print pairs of amicable numbers in a range

In [ ]:

```
def amicable(x,y):
  if  (sum(divisor(x))==y and sum(divisor(y))==x):
    print(x,"&",y,"are amicable numbers")
  else:
    print(x ,"&", y ,"are not amicable numbers")

amicable(28,220)
amicable(220,284)
```

```
28 & 220 are not amicable numbers
220 & 284 are amicable numbers
```

12.Write a program which can filter odd numbers in a list by using filter function

In [ ]:

```
def odd_number(num):
  if num%2!=0:
    return num
num=int(input("Please enter a number: "))
number_list=range(num)
print("The list of original numbers")
print(list(number_list))

odd_numbers=list(filter(odd_number,number_list))
print("The Odd numbers list is")
print(odd_numbers)
```

```
Please enter a number: 20
The list of original numbers
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19]
The Odd numbers list is
[1, 3, 5, 7, 9, 11, 13, 15, 17, 19]
```

13.Write a program which can map() to make a list whose elements are cube of elements in a given list

In [ ]:

```
def cube(num):
    return num**3
num=int(input("Please enter a number: "))
numbers=range(num)
print("The list of original numbers")
print(list(numbers))
cube_numbers=list(map(cube,numbers))
print("The list of cube numbers is")
print(cube_numbers)
```

```
Please enter a number: 20
The list of original numbers
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19]
The list of cube numbers is
[0, 1, 8, 27, 64, 125, 216, 343, 512, 729, 1000, 1331, 1728, 2197, 2744, 3375, 4096, 4913,
5832, 6859]
```

14.Write a program which can map() and filter() to make a list whose elements are cube of even number in a given

14.Write a program which can map() and filter() to make a list whose elements are cube of even number in a given list

In [ ]:

```
def even_number(num):
  if num%2==0:
    return num
num =int(input("Please enter a number: "))
numbers= range(num)
print("The list of original numbers")
print(list(numbers))

#cube_numbers=list(map(cube,numbers))
#print("The list of cube numbers is")
#print(cube_numbers
even_numbers=[]
b=map(cube,filter(even_number,numbers))
for i in b:
  even_numbers.append(i)
print("The list of cube of even numbers is: ")
print(even_numbers)
```

```
Please enter a number: 15
The list of original numbers
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14]
The list of cube of even numbers is:
[8, 64, 216, 512, 1000, 1728, 2744]
```