```
# 1.write a program to print a table of a desired given number.

given_number = int(input())
Table = [i*given_number for i in range(1,11)]
print(Table)
```

```
5
[5, 10, 15, 20, 25, 30, 35, 40, 45, 50]
```

```
# 2. write a program to print twin primes less than 1000.
import math
def isprime(a):
    for i in range (2,int(math.sqrt(a)+1)):
        if a%i == 0:
            return False
            break
    else:
        return True

def twinprime(x,y):
    for i in range(x,y):
        j = i+2
        if isprime(i) and isprime(j):
            print("{}, and {} are twin primes".format(i,j))


twinprime(2,10)
```

```
3, and 5 are twin primes
5, and 7 are twin primes
```

```
# 3.write a program to find out the prime factors of a number!

def prime_factors(a):
    z = []
    div = 2
    while(div <= (a)):
        if a%div == 0:
            z.append(div)
            a = a/div
        else:
            div += 1
    return z

print(prime_factors(88))
```

```
[2, 2, 2, 11]
```

```
# 4. write a program to print decimal to binary.
```

```
def decimal_binary(a):
    z = []
    while (a >= 1):
        if a%2 == 0:
            z.append(0)
        else:
            z.append(1)
        a = a//2
    z.reverse()
    return z

print(decimal_binary(300))
```

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

```
# 5. write a program to find the permutation and combination.

from itertools import permutations
from itertools import combinations

def permutation(a):
    perm = permutations(list(a))
    for i in (perm):
        print(i)

def combination(a,b):
    comb = combinations(list(a),b)
    for i in (comb):
        print(i)

s = [1,2,3]
permutation(s)
t = [1,2,3]
u = 2
combination(t,u)
```

```
(1, 2, 3)
(1, 3, 2)
(2, 1, 3)
(2, 3, 1)
(3, 1, 2)
(3, 2, 1)
(1, 2)
(1, 3)
(2, 3)
```

```
# 6.function cubesum and then find armstrong number.

def cubesum(num):
    sum = 0
    temp = num
    while(temp > 0):
        a = temp % 10
        sum  += (a*a*a)
        temp = temp //10
    return sum

def isArmstrong(num):
    if (cubesum(num) == num):
        print("Its an armstrong number")
    else:
        print("No its not an armstrong number")


def printArmstrong(minimum,maximum):
    for i in range(minimum,maximum+1):
        if (cubesum(i) == i):
            print(i)

print(cubesum(158))
isArmstrong(153)
printArmstrong(1,500)
```

```
638
Its an armstrong number
1
153
370
371
407
```

```
# 7.write a function named prodDigits, which enters the number and
return the product of the digits of that number.

def prodDigits(num):
    prod = 1
    temp = num
    while(temp >= 1):
        a = temp%10
        prod *=a
        temp //=10
    return prod

prodDigits(2349)
```

```
216
```

```
#8. Write a function MDR() and Mpersistence()


def prodDigits(num):
    prod = 1
    temp = num
    while(temp >= 1):
        a = temp%10
        prod *=a
        temp //=10
    return prod

def MDR(num1):
    while(num1 > 10):
        z = prodDigits(num1)
        num1 = z
        print(num1)

def Mpersistent(num2):
    count = 0
    while(num2 > 10):
        z = prodDigits(num2)
        num2 = z
        count += 1
    print(count)


a =51
print(MDR(a))
Mpersistent(a)
```

```
5
None
1
```

```
# 9.Write a function to find the sum of proper divisors.
def sumPdivisors(num):
    sum = 0
    for i in range(1,num):
        if (num%i == 0):
            sum += i
    print(sum)
a = 36
print(sumPdivisors(a))
```

```
55
None
```

```
# 10. write a function for perfect numbers in a given range.


def perfect_number(num):
    sum = 0
    for i in range(1,num):
        if (num%i == 0):
            sum += i
    if sum == num:
        return (num)

x = int(input())
y = int(input())

for a in range(x,y):
    if (perfect_number(a)):
        print(a)
```

```
1
500
6
28
496
```

```
# 11. write a function for amicable numbers in a given range.

def sumPdivisors(num):
    sum = 0
    for i in range(1,num):
        if (num%i == 0):
            sum += i
    return sum

def amicable(x,y):
    if (sumPdivisors(x) == y and sumPdivisors(y) == x):
        return(x,y)


def inrange(m,n):  # time complexity hell! O(n**2) try using
hashtable(2 pass or 1 pass)
    for x in range(m,n):
        for y in range(m+1,n):
            if (sumPdivisors(x) == y and sumPdivisors(y) == x and (x !
= y)):
                print(x,y)


inrange(1,1000)
```

```
220 284
284 220
```

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

def odd_numbers(a):
    if a%2 == 1:
        return a

m = filter(odd_numbers,[1,2,3,4,5,6,7,8,9])
print(list(m))
```

```
[1, 3, 5, 7, 9]
```

```
# 13. write a program which can map to make a list whose elements are
cube of elements in a given list

def square(a):
    lst = (a**2)*a
    return lst

x = map(square,[2,3,4])
print(list(x))
```

```
[8, 27, 64]
```

```
# 14. write a program which can map and filter to make  a list whose
elements are cube of even number in a given list.

def ofeven(e):
    if e%2 == 0:
        return e

def cubes(a):
    c = (a**2)*a
    return c

cubesofeven = map(cubes,(filter(ofeven,[1,2,3,4,5,6,7,8,9])))
print(list(cubesofeven))
```

```
[8, 64, 216, 512]
```