1) Function that inputs a number and prints multiplication table of that number

```
def Multiplication_Table(N):
    i=1
    for k in range(N,N*10+1,N):
        print("{} * {} = {}".format(N,i,k))
        i+=1


Multiplication_Table(10)
```

2) Program to print TwinPrimes < 1000

```
def TwinPrime():
        for k in range(1,999,2):
            print("({},{})".format(k,k+2))


TwinPrime()
```

3) Function to find the factors of number

```
def FactorsOfNumber(N):
    Num = N
    if Num > 1:
        for k in range(2,N+1):
         i=k
         while Num%i == 0:
             print(i)
             Num/=i
    else :
        print(N)

FactorsOfNumber(24)
```

4) Progmam to impliment Permutation and Combination formulae

```
def factorial(N):
    result = 1
    if N == 0:
        return result
    while(N > 1):
        result*=N
        N-=1
    return result

def Permutation_Combination(N,R):

    if type(N) == float or type(R) == float :
        print("Please enter valid input...")
        return

    permutation = factorial(N)/(factorial(N-R))

    combination = permutation/factorial(R)

    print("Number of permutation of N = {} and R = {} are {}".format(N,R,int(permutation))
    print("Number of combination of N = {} and R = {} are {}".format(N,R,int(combination))

Permutation_Combination(5,3)
```

5) Fuction that converts the Decimal number to Binary number

```
def DecimalToBinnary(N):
    count = 0
    binary = []

    if N < 0 :
        print("-",end="")
        N =N * -1

    number = int(N)
    fraction = N - number
    while number >= 1:
        binary.append(number%2)
        number = int(number/2)

    for k in range(len(binary)-1,-1,-1):
        print(binary[k],end="")

    print(".",end="")

    while fraction != 0 and count != 7:
        k = fraction * 2
```

```
        print(int(k),end = "")
        fraction = k - int(k)
        count+=1


DecimalToBinnary(11.625)
```

6) Armstrong Number

```
def Cubesum(N):
    result = 0
    while(N > 0):
        result+=(N%10)**3
        N = int(N/10)
    return result

def Powersum(N,k):
    result = 0
    while N > 0:
        result += (N%10)**k
        N = int(N/10)
    return result

def IsArmstrong(N):

    num = str(N)

    if len(num) == 3:
        result = Cubesum(N)
    else :
        result = Powersum(N,len(num))

    if N == result:
        print("True")
        print(N)
    else:
        print("False")



IsArmstrong(407)
```

7) Function that inputs number and return product of digits of that number

```
def ProductOfDigits(N):
    result = 1
    while(N > 0):
```

```
        result *= int (N % 10)
        N =int (N / 10)
    return result

print (ProductOfDigits(1234))
```

8) Write functions MDR() and MPersistence() that inputs the number and returns it's multiplicative digital root and multiplictive persistance respectively.

```
def ProductOfDigits(N):
    result = 1
    while(N > 0):
        result *= int (N % 10)
        N =int (N / 10)
    return result

def MDR(N):
    k=N
    while True:
        k = ProductOfDigits(k)
        if k <= 9 :
            return k

def MPersistance(N):
    k=N
    count=0
    while True:
        count+=1
        k = ProductOfDigits(k)
        if k <= 9 :
            return count

print(MDR(341))
print(MPersistance(341))
```

9) Function SumPdivisors() that finds the sum of proper divisors.

```
def sumPdivisors(N):
    result = 0
    divisor = 1

    while divisor < N:
        if N%divisor == 0:
            print(divisor,end=" ")
            result+=divisor
        divisor+=1
```

```
    print("\nSum = {}".format(result))

sumPdivisors(220)
```

10) Program to print all PERFECT NUMBER in given range as input to function

```
def sumPdivisors(N):
    result = 0
    divisor = 1

    while divisor < N:
        if N%divisor == 0:
            #print(divisor,end=" ")
            result+=divisor
        divisor+=1

    return result


def PerfectNumber(low,high):
    for k in range(low,high+1,1):
        if k == sumPdivisors(k):
            print(k,end=" ")

PerfectNumber(20,84)
```

11) Function to print pairs of Amicable numbers in a range

```
def sumPdivisors(N):
    result = 0
    divisor = 1

    while divisor < N:
        if N%divisor == 0:
            #print(divisor,end=" ")
            result+=divisor
        divisor+=1

    return result

def amicable_number(low,high):
    lst_sum = []
    for k in range(low,high+1):
        lst_sum.append([k,sumPdivisors(k)])
```

```
    for item1 in lst_sum:
        for item2 in lst_sum:
            if item1 != item2 and item1[0]==item2[1] and item1[1]==item2[0]:
                if item1[0] < item2[0]:
                    print("({},{})".format(item1[0],item2[0]))

amicable_number(200,300)
```

12) Write program which can filter odd number in list using filter().

```
def odd_number(Number):
    return Number%2 == 1


numbers = [int(num) for num in input().split( )]
odd_lst = list(filter(odd_number,numbers))
print(odd_lst)
```

13) Write program which can map() to make list cubes of elements in given list.

```
def PowerOfThree(Number):
    return Number**3

numbers = [int(num) for num in input().split( )]
cube_lst = list(map(PowerOfThree,numbers))
print(cube_lst)
```

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

```
def PowerOfThree(Number):
    return Number**3

def even_number(Number):
    return Number%2 == 0

numbers = [int(num) for num in input().split( )]
even_numbers = list(filter(even_number,numbers))
cubed_even_numbers = list(map(PowerOfThree,even_numbers))

print(cubed_even_numbers)
```

✓ 8s completed at 12:38 PM