f = (c * 9.5) + 32

leap year

```
if (year % 4 == 0 and year % 100 != 0) or
    (year % 400 == 0)
print ("leap year")
```

prime

```
num = int (input)
if num <= 1 :
    not prime
elif num <= 3
    prime
else :
  if num % 2 == 0
     or num % 3 == 0 :
     print not prime
  else
     prime
```

prime interval

```
low = int
high = int
print( . . . . )
for i in range (low, upp+1)
  :
    if n > 1 :
    for i in range (2, n)
      if (n % i ) == 0
        break
    else
      print (n)
```

Armstrong

```
num = int(input)
a = num
sum = 0
while num > 0:
    digit = num % 10
    sum += digit ** len(str(a))
    num //= 10
if sum == a
    print(f"{a} is armstrong)
    else:
        print(f"{a} not armstrong)
```

Interval

```
low = int(input(""))
upp = int(input)
for num in range(low, upp+1)
        a = num
        sum = 0
        temp_num = num
    while temp_num > 0:
        digit = temp_num > 0:
        sum += digit ** num_digits
        temp_num //= 10
    if sum == a:    print(num)
```

fibonacci series

```
num = int(input)

prev = 0

curr = 1

if num <= 0
    print("cannot be -ve or zero")

elif num == 1
    print(f"1")

else:
    print("fibonacci sequence :")
    for i in range(num):
        print(prev)
        prev, curr = curr, prev + curr
```

Sum of n natural nos.

```
num = int(input)

if num < 0:
    print("cannot be -ve")

else
    sum = 0
    while (num > 0):
        sum += num
        num -= 1
    print(f"The sum is {sum}")
```

Swapping

```
a = -10
b = 5
a = a + b
b = a - b
a = a - b
print("Swapping done : a =", a, "b = ", b)
```

random no.

```
import random
rand_list = []
for i in range(0, 10):
    n = random.randint(1,50)
    rand_list.append(n)
print(rand_list)
```

km to miles

```
def km_to_mile(km):
    miles = km * 0.621371
    return miles

km = float(input)
miles = km_to_mile(km)
print(f"Result : {km} km is
        equal to {miles} miles.")
```

cel to farhenite

```
def cel_to_fah(celcius
fah = (celcius * 9.5) + 32
return fah

celcius = float(input)
celcius = cel_to_fah(celcius)
print(f"Result")
```

Calendar

```
import cale
yy = int (
mm = int
print (
```

+ve, -ve.

```
def che
if nu
pri
if n
p
```

prime.

```
def pri
if
for
```

Calendar

```
import calendar
yy = int (input)
mm = int (input)
print (calendar.month(yy, mm))
```

+ve, -ve, zero.

```
def check_num (num):
    if num > 0:
        print("+ve")
    if num < 0:
        print(-ve)
    else:
        print (zero)
num = int (input)
num = check_num (num)
```

prime.

```
def prime_check (a):
    if a>1:
        for j in range (2, int (a/2) + 1):
            if (a%j) == 0:
                print ("not prime")
                break
        else:
            print ("prime")
    else:
        print ("not prime")
a = int (input)   prime_check(a)
```

leapyear

```
year = int(input)
if (year % 4 == 0 and
    year % 100 != 0) or
    (year % 400 == 0):
    print ("leap year")
else:
    print ("not leap year")
```

odd even

```
def check_oddeven(num):
    if num % 2 == 0:
        print("even")
    else:
        print ("odd")
num = int (input)
num = check_oddeven (num)
```

prime_in_interval

```
low_val = int(input)
upp_val = int(input)
print ("prime nos in range:")
for n in range (low_val,
                upp_val+1):
    if n > 1:
        for i in rang(2, n):
            if (n % i) == 0:
                break
        else
            print (n)
```

Factorial

```
num = int (input)
fact = 1
if num < 0 :
    print (cannot se -ve)
elif num == 0 :
    print (1)
else :
    for i in range (1, num + 1)
        fact = fact * 1
    print (f"{fact}")
```

Multiplication table

```
def mul_table (num):
    for i in range (1, 11):
        print (f"{num} * {i} = {num * i}")
num = int (input)
mul_table (num)
```

fibonacci

```
def fib (n) :
    fib_series = [0, 1]
    for i in range (2, n):
        fib_series.append (fib_series [i-1] + fib_series [i-2])
    return fib_series
n = int (input)
if n <= 0 :
    print (cannot se -ve)
else
    result = fib (n)
    print (f" series : {result} ")
```

fibonacci

```
num =
prev
curr
if
eli
e
```

Sum

```
num
if
```

Area of Δ

```
def cal_area (a,b,c):
    s=(a+b+c)/2
    area = s * (s-a) * (s-b) * (s-c)) ** 0.5
    return area

a = input
b = input
c = input
area = cal_area(a,b,c)
print ("The area of triangle is %0.2f %. area)
```

Quadratic

```
import cmath
def f_roots (a,b,c)
    d = b**c - 4*a*c
    rt = cmath.sqrt(abs(d))
    roots = [(-b +rt) / (2*a), (-b -rt)/(2*a)]
if d >0: print(f"Real and diff roots : {roots}")
    elif d == 0: print(f"Real and same roots: {roots[0]}")
    else : print(f"complex roots : {roots}")

a,b,c = map (float ,input('Enter a,b, c, by using space:')
        .split() )
if a==0 : print (a cannot be zero)
else: find_roots (a,b,c)
```

datatypes

int - 10
float - 20.5
boolean - True
complex - "1j"
string - "PRESIDENCY. AB"
list - ["AA", "BB", "CC"]
tuple - ("AA", "BB", "CC")
dict - {"name": "Rahul", "age": 30}
bytes - b"Hello"
set - {"AA", "BB", "CC"}
frozenset - frozenset({"AA", "BB", "CC"})

```
print(x)
print(type(x))
```

Arithmetic

```
def operations(num1, num2):
    add = float(num1) + float(num2)
    sub = float(num1) + float(num2)

    return add, sum

num1 = input('Enter first number: ')
num2 = input('Enter second number: ')

add, sub = operations(num1, num2)

print(f'add : {num1} + {num2} = {add}')
print(f'sub : {num1} - {num2} = {sub}')
```

Area of

```
def cal
    s=(
    area
    re
a =
b =
c =
ar
p
```

Quad

```
inp
def
```