```
import math as MT

#  PN:Prime Number Class
#           creator: endy jordan
#           create date:2019.12.28(sat)
#
class PN:

#
#  P(x):Prime Number Calculate
#            Parameter :
#                     x:Number
#            return:
#                     prime number -> input number
#                     not prime number -> min prime number
#

    def P(self,x):

        if x%2==0: #Even
            return 2
        else: #Odd
            if x%3==0:
                return 3
            else:
                if x%5==0:
                    return 5
                else:
                    if x%7==0:
                        return 7
                    else:
                        n=11
                        while x>=n:
                            if x%n==0:
                                break
                            else:
                                n=n+2
                        return n

#
#  primeFactorization(x):Prime Factorization Calculate
#            Parameter :
#                     x:Number
#            return:
#                     prime Factorizationr
#
```

```
    def primeFactorization(self,x):
        print('--------------------')
        print('Prime Factorization')
        print('--------------------')

        if self.jadgePN(x)==1:
            print(str(x)+'...Prime Number')
            return x
        elif self.jadgePN(x)==0:
            pF=x
            while self.jadgePN(pF)==0:
                print(self.P(pF))
                pF=int(pF/self.P(pF))
            print(self.P(pF))

#
#  jadgePN(x):jadge for prime number
#            Parameter :
#                      x:Number
#            return:
#                     0:not prime number
#                     1:prime number
#


    def jadgePN(self,x):
        if self.P(x)!=x:
            #print('x='+str(x)+'(/'+str(self.P(x))+') is not prime number')
            return 0

        elif self.P(x)==x:
            #print('x='+str(x)+' is prime number')
            return 1

#
#  cntPN(x):count for range of prime number
#            Parameter :
#                      p1:prime number
#                      p2:prime number
#            return:
#                     number of prime number(p1 count,p2 not count)
#


    def cntPN(self,p1,p2):
        n=p1
        cnt=0
        while n<p2:
```

```
            if self.jadgePN(n)==1:
                cnt=cnt+1
            n=n+1
        return cnt

#
#  nextPN(x):next prime number
#            Parameter :
#                      x:any number
#
#            return:
#                     neiborhod next prime number
#
    def nextPN(self,x):
        if x%2==0:
            n=x+1
        else:
            n=x+2

        while self.jadgePN(n)==0:
              n=n+2
        return n

#
#  prePN(x):pre prime number
#            Parameter :
#                      x:any number
#
#            return:
#                     neiborhod next prime number
#
    def prePN(self,x):
        if x%2==0:
            n=x-1
        else:
            n=x-2

        while self.jadgePN(n)==0:
              n=n-2
        return n

#
#  sqrtPN(x):sqrt prime number
#            Parameter :
#                      x:any number
#
#            return:
#                     sqrt prime number
```

```
#

    def sqrtPN(self,x):
        sPN=x
        flg=0
        while flg==0:
            if MT.sqrt(sPN)-MT.floor(MT.sqrt(sPN))==0:
                if self.P(MT.sqrt(sPN))==MT.sqrt(sPN):
                    flg=1
                    return int(MT.sqrt(sPN))
            sPN=sPN-1

#  ------------------------------------------------------
#  Main Routine
#  ------------------------------------------------------

#
#  rangePN(x):pre prime number
#            Parameter :
#                     x:any number
#
#            return:
#                     neiborhod next prime number
#

def rangePN(x):

    pn=PN()
    n=pn.P(x)
    if x==n:
        print('x='+str(x)+' is Prime Number')
    else:
        print('x='+str(x)+' is not Prime Number')

    pn.primeFactorization(x)

    pPN=pn.prePN(x)
    nPN=pn.nextPN(x)
    PNn0=pn.sqrtPN(x)
    PNn1=pn.nextPN(PNn0)
    PNn02=PNn0**2
    PNn12=PNn1**2
    cntPN2=pn.cntPN(PNn02,PNn12)
    rNumber=PNn12-PNn02
    ratioPN=cntPN2/rNumber*100

    print('----------Range Prime Number----------')
```

```
    print('prePN=[%10d] <= x=[%10d] < postPN=[%10d]' %(pPN,x,nPN))

    print('')
    print('----------Range SQRT Prime Number----------')
    print('SQRT Pn=%10d --> Pn+1=%10d ' %(PNn0,PNn1))
    print('[Pn^2=%10d --> Pn+1^2=%10d] Rcount=%5d Pcount=%5d ratio=%5.2f' %
(PNn02,PNn12,rNumber,cntPN2,ratioPN))
```