<u>Travaux pratiques et approche expérimentale</u>

- **PGCD** et **coefficients** de **Bézout** de **deux entiers naturels a** et **b**
  *Le **PGCD** des deux entiers est obtenu par l'**algorithme d'Euclide** : boucle réalisée tant que le reste est non nul. Les **coefficients** de **Bézout**, **a** et **b** entiers sont tels que* $\boldsymbol{a} \times \boldsymbol{x} + \boldsymbol{b} \times \boldsymbol{y} = \boldsymbol{x} \wedge \boldsymbol{y}$*. Ils sont obtenus par **récursivité** (**Bezout**(.,.) étant présente dans sa propre définition.)*
  <u>Application</u> :
  409 × 22600956 – 543 × 17023545 = 17023545 ∧ 22600956 = 6069.

```
def PGCD(x, y):
    while x % y != 0:
        x, y = y, x % y
    return str(y)

def Bezout(x, y):
    if y == 0:
        return (1, 0)
    else:
        (q, r) = divmod(x, y)
        (w, z) = Bezout(y, r)
        return (z, w - q * z)

print (PGCD(17023545, 22600956))
print (Bezout(17023545, 22600956))

PGCD = 6069
(a, b) = (-543, 409)
```

- **Décomposition** en **facteurs premiers** d'un **entier naturel n ≥ 1** *(Cf. (Dfp))*
  *Tout **entier naturel n ≥ 1** s'écrit de façon **unique*** $\boldsymbol{n = p_1^{\delta_1}\, p_2^{\delta_2} \ldots p_r^{\delta_r}}$.
  *($\boldsymbol{\delta_i} \in \mathbb{N}^*$, calculées avec **chaine[di]**) de **nombres premiers** ($\boldsymbol{p_i}$ ↗, calculés avec **k**) :*

```
def decomposition(n):
    chaine = {}
    dec = 1
    x = n
    di = 2
    while dec < n:
        while x % di == 0:
            if di not in chaine:
                chaine[di] = 1
            else:
                chaine[di] += 1
            dec *= di
            x //= di
        di += 1
    resultat = str(n) + ' = '
    for k in sorted(chaine):
        resultat += str(k) + '^' + str(chaine[k]) + '.'
    return resultat

print (decomposition(17023545))
print (decomposition(22600956))
print (decomposition(6069))

17023545 = 3^2.5^1.7^1.17^3.
22600956 = 2^2.3^1.7^3.17^2.19^1.
6069 = 3^1.7^1.17^2.
```

- **Fonction ζ** et **comptage** des **nombres premiers** en **python** *(Cf. **M2 – TP TNP**)*

  . On **montre** que $\mathbf{1/\zeta(s) = \sum_{n=1}^{+\infty} \mu(n)/n^s}$, *où μ est la **fonction** de **Möbius** (Cf. **exo 4**)*, $\mathbf{\zeta(s-1)/\zeta(s) = \sum_{n=1}^{+\infty} \varphi(n)/n^s}$, *où φ est l'**indicatrice** d'**Euler** (Cf. **ci-après**) :* $\boldsymbol{\varphi} : \mathbb{N}^* \to \mathbb{N}^*, \boldsymbol{n \longmapsto card(m \in [\![1,n]\!], m \wedge n = 1)}$. ***Dém** : **produit** de **Cauchy** de $\sum n^{-s}$ et $\sum \mu(n)n^{-s}$ pour **obtenir** $\zeta(s)\sum_{n=1}^{+\infty}\mu(n)n^{-s} = 1$ et de même $\sum n^{-s}\sum\varphi(n)n^{-s}$ pour $\zeta(s)\sum_{n=1}^{+\infty}\varphi(n)n^{-s} = \zeta(s-1)$. (Cf. **produit eulérien** : **exo 8**)*

  MP

  Par la **décomposition** en **facteurs premiers** : $\mathbf{n = \prod_{i=1}^{r} p_i^{\gamma_i}}$.
  $\to \mathbf{\varphi(n) = \prod_{i=1}^{r} p_i^{\gamma_i - 1}(p_i - 1) = n\prod_{i=1}^{r}(1 - 1/p_i)}$. *On **sous-entend** $p \in \wp$* ↓

  . **Valuation p-adique** ($p \in \wp$) de $\mathbf{m \in \mathbb{N}^*}$, $\mathbf{v_p(m)}$: **exposant** de **p** en la **décomposition** en **facteurs 1iers** de **m** *(et **plus grand entier**, **v**, tel que $p^v | m$)*. $\pi(x) := |\{p \in \wp, p \le x\}|$
  On **obtient** la **formule** de **Legendre** : $\mathbf{v_p(n\,!) = \sum_{k=1}^{+\infty} E(n/p^k) = E(n/p) - E(n/p^2) + 2\big(E(n/p^2) - E(n/p^3)\big) + 3(\dots)\dots = \big(n - s_p(n)\big)/(p-1)}$, *où $s_p(n)$: **somme** des **chiffres** de **n** en **base p**. $\forall n \in \mathbb{N}^*$, **nombre** d'**entiers** $\le$ **n**, **divisibles** par **aucun** p_{i_k}, **1iers** $\le \sqrt{n}$:*
  $\pi(n) - \pi(\sqrt{n}) + 1 = n - \sum_{i_1} E(n/p_{i_1}) + \sum_{i_1<i_2} E(n/p_{i_1}p_{i_2}) - \sum_{i_1<i_2<i_3} E(n/p_{i_1}p_{i_2}p_{i_3}) + \cdots$
  *par le **principe** d'**inclusion-exclusion**. (Cf. **def pi(n)**, **partie droite-def legendre(n)** ↓)*

  . On **obtient** : $\mathbf{n\ln n + \mathcal{O}(n) = \ln n! = \sum_{p\le n} v_p(n!)\ln p = n\sum_{p\le n}\ln p/p + \mathcal{O}(n)}$.
  $\to \mathbf{\sum_{p\le n}\ln p/p = \ln n + \mathcal{O}(1)}$, puis $\mathbf{\sum_{p\le n} 1/p \sim \ln\ln n}$ *(**th.** de **Mertens**, $p \in \wp$ ds Σ)*
  Encadrt de Tchebychev : $\mathbf{(n/\ln n)(\ln 2 + o(1)) \le \pi(n) \le (n/\ln n)(\ln 4 + o(1))}$
  ***Dém.** avec $v_p\left(\binom{2n}{n}\right)$, $\ln\binom{2n}{n} \sim n\ln 4$, $\pi(n) = \sum_{p\le n} 1$, $\vartheta(n) = \sum_{p\le n}\ln p$, $p \in \wp$ ds Σ*
  $\vartheta(n) \sim \pi(n).\ln n \sim n, \vartheta(n) \le \vartheta(n/2) + n.\ln 2 + C\ln(n/2) \to \vartheta(n) \le n\ln 4 + o(n)$
  TNP : en $+\infty$, $\mathbf{x/\ln x \sim \pi(x) \sim Li(x) = \int_2^x du\, u/\ln u}$ *(Cf. **partie gauche**, **graphe** ↓)*

```python
import math as mt
import numpy as np
import scipy.integrate as integ
import matplotlib.pyplot as plt

def Li(x):
    return integ.quad(lambda t: 1/mt.log(t), 2, x)[0]

def test_premier(n):
    m = int(mt.sqrt(n))
    for i in range(2, m+1):
        if n%i == 0: return False
    return True

def parties(L):
    if L == []: return [[]]
    else: p = parties(L[1:])
    return p + [[L[0]] + partie for partie in p]

def premiers(n): # nbe premiers <= n
    p = []
    for i in range(2, n+1):
        if test_premier(i):
            p.append(i)
    return p

def pi(n):
    return np.size(premiers(n))

x_val = range(2, 200)
Li_val =[Li(x) for x in x_val]
y_val = [pi(x) for x in x_val]
plt.plot(x_val, Li_val, label = "Li(n)")
plt.plot(x_val, y_val, label = "Nombres premiers")
plt.legend()
plt.show()
```

```python
def legendre(n):
    prt = parties(premiers(int(mt.sqrt(n))))
    x = 0
    for pr in prt:
        npr = np.size(pr)
        exppr = 1-2*np.mod(npr, 2) # = mt.pow(-1, npr)
        if (npr == 0): x+= n
        else:
            pepr = 1
            for epr in pr:
                pepr *= epr
            x += int(n/pepr)*exppr
    return int(x)
u = 1557
print(legendre(u))
print(pi(u)-pi(int(mt.sqrt(u)))+1)

Out :
234
234
```