TP4
Arithmétique

3 janvier 2018

MILHA El Mehdi & ERDEMIR Eren

1. **Un nombre est-il premier ?**
   (a) En python, le quotient et le reste de la division euclidienne d'un entier par un autre entier sont réprésentés par // et % respectivement.
   (b) Un nombre premier est un nombre entier naturel (non nul) qui admet exactement 2 diviseurs distincts : 1 et lui-même.
   (c) 1001 n'est pas premier. 2017 est premier. 3001 est premier. 49999 est premier. 89999 n'est pas premier.
   (d) La fonction python `is_prime` (qui prend en argument un entier n et qui renvoie `true` si n est premier, `false` sinon) est : voir exo1-4.py
   (e) Les nombres de Fermat F_n sont définis par $F_n = 2^{2^n} + 1$; les nombres $F_0, F_1, F_2, F_3 et F_4$ sont premiers mais F_5 n'est pas premier.
2. **Crible d'Erathostène,distribution des nombres premiers.**
   (a) A l'aide du crible d'Erathostène, on a calculé la liste de tous les nombres premiers inférieurs à 200 : 2,3,5,7,11,13,17,19,23,29,31,37,41,43,47,53,59,61,67 ,71,73,79,83,89,97,101,103,107,109,113,127,131,137 ,139,149,151,157,163,167,173,179,181,191,193,197 et 199.
   (b) On a défini la fonction python `primes` (qui prend en argument un entier n et qui renvoie la liste de tous les premiers inférieurs à n) voir exo2-4.py.
   (c) Voici la liste de tous les premiers inférieurs à 1000 écrite dans un fichier primes.txt, dix nombres par ligne : voir primes.txt.
   (d) On note $\pi(n)$ le nombre d'entiers premiers inférieurs à n.On a représenté graphiquement $\pi(n)$ en fonction de n pour n variant de 2 à 1000. Malheureusement on arrive pas a représenter les deux fonction on ne peut rien observer.
   (e)

n	$\pi(n)$	$\frac{n}{\log n}$
10^1	4	4.34294481903
10^2	25	21.7147240952
10^3	168	144.764827301
10^4	1229	1085.73620476
10^5	9592	8685.88963807
10^6	78498	72382.4136505

3. **Factorisation d'un entier en premiers.**
   (a) Théorème Fondamental de l'arithmétique : tout entier strictement positif peut être écrit comme un produit de nombres premiers d'une unique façon, à l'ordre près des facteurs.
   (b) Effectuons la décomposition en facteurs premiers de 924 : $924 = 2^2 * 3 * 7 * 11$

(c) On a défini une fonction python `factors` qui prend en arguments un entiers n et qui renvoie la liste, dans l'ordre croissant, des facteurs premiers de n, chaque facteur étant répété autant de fois que nécessaire. Ainsi, pour $n = 60$, on obtiendra $[2, 2, 3, 5]$ voir exo3-4.py ou tp4.py.

4. **pgcd de deux entiers, identité de Bézout, algorithme d'Euclide.**

   (a) Le PGCD de deux nombres entiers, non nuls tous les deux, est le plus grand des diviseurs communs de ces deux nombres.

   (b) On a $a = 4864 = 2^8 * 19$ et $b = 3458 = 2 * 7 * 13 * 19$ donc $pgcd(a, b) = 2 * 19 = 38$.

   (c) L'Identité de Bézout est un résultat d'arithmétique élémentaire, qui prouve l'existence de solutions à l'équation diophantienne linéaire : $ax + by = pgcd(a, b)$ d'inconnues x et y entiers relatifs, où a et b sont des coefficients entiers relatifs et où pgcd(a, b) est le plus grand commun diviseur de a et b.

   (d) Calculons $d = pgcd(a = 4864, b = 3458)$ Pour trouver d, utilisons l'algorithme d'Euclide : $4864 = 1 * 3458 + 1406$
   $3458 = 2 * 1406 + 646$
   $1406 = 2 * 646 + 114$
   $646 = 5 * 114 + 76$
   $114 = 1 * 76 + 38$
   $76 = 2 * 38 + 0$
   donc $d = 38$
   Trouvons x et y tels que $4864 * x + 3458 * y = d$ $114 - 76 = 38$
   $114 - (646 - 5 * 114) = -646 + 6 * 114 = 38$
   $-646 + 6 * (1406 - 2 * 646) = 6 * 1406 - 13 * 646 = 38$
   $6 * 1406 - 13 * (3458 - 2 * 1406) = 32 * 1406 - 13 * 3458 = 38$
   $-13 * 3458 + 32 * (4864 - 1 * 3458) = 32 * 4864 - 45 * 3458 = 38$
   donc $x = 32$ et $y = -45$

   (e) On a défini la fonction python `euclide` qui prend en arguments deux entiers a, b et qui renvoie x, y, d, où d est le pgcd de a, b et x, y les coefficients de Bézout, voir exo4-4.py.