Séance du 13 février 2015 :

Bases d'algorithmique, algorithmes d'Euclide

1 Le programme sur ce sujet :

> **2 Algorithmique et informatique**
> Notions de variable et de type. Instructions d'affectation, conditionnelles, d'itération. Fonctions et procédures (ou sous-programmes) ; passage de paramètre, variables locales, notion de récursivité. Rédaction en français ou dans un langage au choix du candidat de programmes ne comportant qu'un faible nombre d'instructions et pouvant utiliser des fonctions (ou sous-programmes). Aucun développement théorique n'est exigé. Exemples d'algorithmes illustrant les notions figurant dans le présent programme.
>
> **3 Algèbre générale**
>
> **3.1 Extensions successives de la notion de nombre**
> Anneau $\mathbb{Z}$ des entiers relatifs. Division euclidienne. Sous-groupes additifs et idéaux de $\mathbb{Z}$. Nombres premiers. Décomposition en facteurs premiers. Plus grand commun diviseur (PGCD) et plus petit commun multiple (PPCM). Théorème de Bachet-Bézout. Algorithme d'Euclide étendu.

2 Références

Pour python :
- le livre de G.Swinnen sur python : `http://inforef.be/swi/download/python_notes_hyper.pdf` , ce document est sous licence GNU, donc libre d'accès. A lire sur ordi ou tablette, ça n'est pas rentable de l'imprimer intégralement, autant acheter le livre en librairie.
- un cours en ligne de python très bien fait : `http://www.courspython.com/sommaire.html`, pour " mettre le pied à l'étrier " et plus...

Pour l'arithmétique :
- Très précisément : "Mathématiques Tout-en-un première année" (2ieme édition MPSI/PCSI) de Ramis/Deschamps/Warusfel. page 683 et suivantes.
- Toute bonne "suite de CPGE", au rayon arithmétique.

3 Travail proposé

On propose de réaliser une implémentation en python des algorithmes d'Euclide suivants :

1) Algorithme des différences pour calculer quotient et reste d'une division Euclidienne d'entiers naturels
2) Algorithme d'Euclide pour le calcul du PGCD de deux entiers naturels
3) Algorithme d'Euclide étendu pour le calcul simultané du PGCD et des coefficients de Bézout d'un couple d'entiers naturels.

On considère $\mathbb{N}$ connu par ses "4 propriétés fondamentales" :

a) $\mathbb{N}$ est un ensemble non-vide totalement ordonné par une relation d'ordre notée $\leq$, appelée ordre naturel des entiers.

b) Toute partie non-vide de $\mathbb{N}$ possède un plus petit élément

c) Toute partie non-vide majorée de $\mathbb{N}$ possède un plus grand élément

d) $\mathbb{N}$ lui-même ne possède pas de plus grand élément

3.1 Algorithme des différences

3.1.1 Base mathématique

Théorème de la division euclidienne dans $\mathbb{N}$:

$$\forall (a,b) \in \mathbb{N} \times \mathbb{N}^*, \exists ! (q,r) \in \mathbb{N}^2, \begin{cases} a = bq + r \\ r < b \end{cases}$$

Remarques :
- vocabulaire : q est le quotient, r le reste etc...
- Le diviseur b ne peut pas être nul .

Démonstration : (uniquement l'existence, l'unicité ne nous intéresse pas pour la construction de l'algorithme)
Considérons l'ensemble $E = \{k \in \mathbb{N}, a - bk \geq 0\}$:
- c'est une partie non-vide de $\mathbb{N}$ car il contient 0
- c'est une partie majorée de $\mathbb{N}$: $a + 1$ est un majorant.

E possède donc un plus grand élément, que l'on note q, on pose alors $r = a - bq$, comme $q + 1$ n'est pas dans E, on a alors $r - b < 0$... Donc (q, r) convient.

3.1.2 Construction de l'algorithme

Cette démonstration de l'existence nous indique la marche à suivre : "il suffit de retrancher b autant de fois que nécéssaire jusqu'à l'obtention d'un résultat négatif. Le nombre d'étape est alors un de plus que le quotient, le reste s'en déduit...

3.1.3 Algorithme en pseudo-langage

Version récursive

Version itérative

3.1.4 Algorithme en python

Version récursive

Version itérative

3.2 Calcul du PGCD

3.2.1 Base mathématique

Pour $a \in \mathbb{N}$, et $k \in \mathbb{N}$, on rappelle que $k|a \Leftrightarrow \exists c \in \mathbb{N}, a = kc$. (on lit "k divise a").
Notons $D(a) = \{k \in \mathbb{N}, k|a\}$ l'ensemble des diviseurs de a .
Remarques :
- Avec ces définitions $D(0) = \mathbb{N}$, cependant la locution "diviseur de zéro" à un autre sens en algèbre...
- $\forall a \in \mathbb{N}, \{1, a\} \subset D(a)$
- a est dit premier lorsque $D(a)$ est de cardinal 2. (Ce qui exclut 1 des nombres premiers...)

– Lorsque a est non-nul, $D(a)$ est une partie non-vide majorée de $\mathbb{N}$, son plus grand élément est a.

Pour $(a,b) \in \mathbb{N}^2$, notons $CD(a,b) = D(a) \cap D(b)$, l'ensemble des diviseurs communs (ou "Communs Diviseurs") de a et de b.

Remarques :
– il y a évidemment symétrie...
– $CD(a,0) = D(a)$
– $CD(a,1) = \{1\}$
– a et b sont premiers entre eux ssi $CD(a,b) = \{1\}$, cela peut-être pris comme une définition.
– $\forall (a,b) \in \mathbb{N}^2, \{1\} \in CD(a,b)$, en particulier $CD(a,b)$ est non-vide.
– $CD(a,b)$ est majoré par a et par b (lorsqu'ils sont non-nuls)
– $CD(a,b)$ possède donc un plus grand élément noté $PGCD(a,b)$
– $PGCD$ n'est défini que pour des couples d'entiers non-nuls...

Propriété : Si (a,b,q,r) sont quatre entiers naturels tels que $a = bq + r$,Alors $CD(a,b) = CD(b,r)$

Démonstration : procéder par double inclusion

3.2.2 Construction de l'algorithme

La propriété suivante s'applique donc avec le reste de la DE de a par b :
– Si le reste est 0 (ou 1) le PGCD est alors connu d'après les remarques précédentes.
– Sinon on itère le procédé en divisant b par r ...

On itère autant de fois que nécéssaire, la suite des restes successifs et une suite d'entiers naturels strictement décroissante, ce qui assure la terminaison de l'algorithme.

On dispose là directement d'une formulation récursive de l'algorithme...

3.2.3 Algorithme en pseudo-langage

Version récursive

Version itérative

3.2.4 Algorithme en python

Version récursive	Version itérative

3.3 Calcul simultané du PGCD et des coefficients de Bézout

3.3.1 Base mathématique

Rappelons le théorème de Bezout :

$$\forall (a,b) \in \mathbb{N}^{*2}, \exists (u,v) \in \mathbb{N}^2, au + bv = PGCD(a,b)$$

Remarques :
- il n'y a pas unicité du couple (u,v)
- la démonstration théorique n'est pas proposée ici : elle n'est d'aucun secours pour l'aspect algorithmique... C'est l'algorithme qui sera lui-même un preuve.

3.3.2 Construction de l'algorithme

Il faut reprendre et examiner en détail l'étape d'itération de l'algorithme précédent :
- à la $n-$ième étape le dividende a_n s'écrit par division euclidienne $a_n = b_n q_n + r_n$, puis en cas de poursuite :
- le diviseur devient le nouveau dividende : $a_{n+1} \leftarrow b_n$
- et le reste le nouveau diviseur : $b_{n+1} \leftarrow r_n$.

Le dernier reste non-nul sera le PGCD cherché...

Supposons qu'à cette $n-$ième étape le dividende a_n s'écrive $a_n = au_n + bv_n$
et qu'à la précédente a_{n-1} s'écrivait $a_{n-1} = au_{n-1} + bv_{n-1}$:

Alors $q_{n-1}a_n = q_{n-1}b_{n-1} = a_{n-1} - r_{n-1} = a_{n-1} - b_n = a_{n-1} - a_{n+1}$

Mais par ailleurs $q_{n-1}a_n = aq_{n-1}u_n + bq_{n-1}v_n$

Ainsi $a_{n+1} = a_{n-1} - (aq_{n-1}u_n + bq_{n-1}v_n) = a(u_{n-1} - q_{n-1}u_n) + b(v_{n-1} - q_{n-1}v_n) = au_{n+1} + bv_{n+1}$

On dispose donc d'une relation de récurrence d'ordre 2 pour la suite (u_n, v_n)...

Il faut initialiser cette suite avec $(u_0, v_0) = (\quad , \quad)$ et $(u_1, v_1) = (\quad , \quad)$.
- si le reste r_n est nul, on est dans le cas de terminaison, $a_{n+1} = b_n = r_{n-1}$ est le dernier reste non-nul, c'est donc bien le PGCD et $(u,v) = (u_{n+1}, v_{n+1})$ convient
- sinon on passe à l'itération suivante... L'algorithme se termine car la suite r_n est, comme précédemment, une suite d'entiers naturels strictement décroissante.

3.3.3 Algorithme en pseudo-langage

Version récursive

Version itérative

3.3.4 Algorithme en python

Version récursive

Version itérative