Institut Joseph Fourier
Master II C.S.C.I
2004-2005 - semaine 40

TP Maple - Théorie des Nombres.

TP1

Sébastien Varrette `Sebastien.Varrette@imag.fr`
Gérard Vinel `Gerard.Vinel@ujf-grenoble.fr`

Conseils Préliminaires

Il vous est demandé de réaliser les exercices qui suivent sur une feuille de style Maple (format `.mws`). Cette dernière devra comporter outre les différentes procédures et quelques exemples pour les illustrer, les élements suivants :
- Titre du TP
- Noms des binômes
- Titre des exercices réalisés et justification des choix d'implémentation si besoin est.
- Tout commentaire qui vous paraît le bienvenu.

Pour chaque procédure vous sera précisé son prototype (variables d'entrée (Input), de sortie (Output) et les eventuelles contraintes sur ces paramètres). Merci de les rappeler et d'en tenir compte dans la réalisation pratique de ces procédures.
Cette feuille de style devra être envoyée par mail aux adresses mails ci-dessus.
TODO : voir éventuellement un autre mode de dépot

L'aide de Maple est très utile et pratique : utilisez la !
Enfin, n'oubliez pas de commenter et d'indenter votre code.

1 Euclide et Euclide Etendu

1.1 (pour se mettre en jambe) Algorithme d'Euclide

Théorème 1 (Euclide)**.** *Soit a, $b \in \mathbb{N}$ / $a \leq b$. Soit r le reste de la division euclidienne de a par b. Alors pgcd(a,b)= pgcd(b,r).*

L'algorithme d'Euclide consiste donc a réitérer les manipulations suivantes :
- Effectuer la division euclidienne de a par b. Soit r le reste.
- Remplacer a par b et b par r. (on a donc $0 \leq r < b$ d'après la définition de la division euclidienne).

Le P.G.C.D. est le dernier reste non nul. (Si b divise a, le P.G.C.D. est b et par convention, pgcd(a,0)=a)

On itère donc la fonction G suivante :

$$G \ : \ \begin{bmatrix} a \\ b \end{bmatrix} \longmapsto \begin{bmatrix} 0 & 1 \\ 1 & -a \text{ div } b \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix}$$

Exercice 1. *Réaliser les procédures* `Euclide_rec` *et* `Euclide_it`*, qui correspondent respectivement à une version récursive et itérative de l'algorithme d'Euclide (aidez vous de la fonction Maple* `irem`*). Tester vos procédures sur quelques exemples.*

Procedure `Euclide_rec(`a,b`)`

Input: a, $b \in \mathbb{N}$

Output: d=pgcd(a,b)

Procedure `Euclide_it(`a,b`)`

Input: a, $b \in \mathbb{N}$

Output: d=pgcd(a,b)

1.2 (parce qu'il le vaut bien) Bézout et l'algorithme d'Euclide Etendu

Théorème 2 (Bezout)**.** *Soit a, $b \in \mathbb{N}$. Soit $d = pgcd(a, b)$*
Alors $\exists u, v \in \mathbb{Z}$ / $a.u + b.v = d$

L'algorithme d'Euclide Etendu permet de construire effectivement les coefficients de Bézout. L'idée est la suivante :
D'après la division euclidienne de a par b : $a = bq + r$ avec $0 \leq r < b$. Soient u' et v' les coefficients de Bézout de b et r : $b.u' + r.v' = d$ on a donc :

$$\begin{aligned} d &= pgcd(a, b) = pgcd(b, r) \\ &= b.u' + r.v' = b.u' + (a - bq)v' \\ &= a.v' + (u' - qv')b \end{aligned}$$

On peut donc prendre $u = v'$ et $v = u' - qv'$.

Exercice 2. *Réaliser la procédure* `Bezout` *(version récursive et si vous avez le temps, en version itérative*[1]*) permettant d'obtenir les paramètres u, v et d. Vous utiliserez pour cela l'algorithme d'Euclide Etendu.*
Tester votre procédure sur quelques exemples.

Procedure `Bezout(`a,b`)`

Input: a, $b \in \mathbb{N}$

Output: $[d, u, v]$ / $d = pgcd(a, b)$

---

1. Aidez vous de la fonction G

1.3 (pour le fun) Application : réalisation de l'inverse modulaire

Exercice 3. *Utilisez la procédure* `Bezout` *pour créer la procédure* `Inv_Mod`. *Tester votre procédure sur quelques exemples.*

Procedure `Inv_Mod(`a`,`n`)`

Input: a, $n \in \mathbb{N}$

Output: $\tilde{a} \in \mathbb{N} \cup \{-1\}$

Result: $\tilde{a} = \begin{cases} a^{-1} \bmod \text{n} & \text{si pgcd}(a,n)=1 \\ -1 & \text{sinon} \end{cases}$

Remarque : en Maple, l'inverse modulaire de v modulo n est obtenu par la commande `1/v mod n`

2 Exponentiation modulaire dans $\mathbb{Z}/n\mathbb{Z}$

Principe :
On cherche à calculer $a^e \bmod n$. L'algorithme se déroule en trois étapes :

1. Décomposition de e en binaire :

$$e = \sum_{i=0}^{r} e_i 2^i \text{ avec } e_i \in \{0,1\}$$

2. Calculer les carrés successifs $\{a^{2^k} \bmod n\}_{k\in[0,r]}$ en utilisant la formule : $a^{2^{k+1}} = \left(a^{2^k}\right)^2$.
3. Utiliser la formule suivante :

$$\begin{aligned} a^e \bmod n &= a^{\sum_{i=0}^{r} e_i 2^i} \bmod n \\ &= \prod_{i=0}^{r} \left(a^{2^i}\right)^{e_i} \bmod n \end{aligned}$$

On remarque enfin que si pgcd(a,n)=1, alors $\forall e < 0, a^e = \left(a^{-1}\right)^{|e|}$

Exercice 4. *Réaliser la procédure* `Expo_rapide` *selon le principe décrit ci-dessus. Tester votre procédure sur quelques exemples.*

Procedure `Expo_Rapide(`a`,`e`,`n`)`

Input: $(a,e,n) \in \mathbb{N} \times \mathbb{Z} \times \mathbb{N}$

Output: $a^e \bmod n$

---

3 Théorème des restes chinois constructifs

(si vous êtes en retard, passez directement au §3.2)

3.1 (juste pour rire) A deux variables n et m

Principe :
Soient n et m deux entiers premiers entre eux.
Alors le théoreme des restes chinois assure l'existence de $x \in \mathbb{Z}/nm\mathbb{Z}$ tel que :

$$\forall (x_1, x_2) \in \mathbb{Z}/n\mathbb{Z} \times \mathbb{Z}/m\mathbb{Z}, \begin{cases} x \equiv x_1 \bmod n \\ x \equiv x_2 \bmod m \end{cases}$$

Le theoreme des restes chinois constructifs permet de calculer explicitement x à partir de x_1, x_2, n et m.
Idee :
D'apres le théorème de Bézout, on peut trouver $u, v \in \mathbb{Z}$ tels que $n.u + m.v = 1$ (voir §1.2). On en déduit que $n.u \equiv 1[m]$ et $m.v \equiv 1[n]$.
En particulier, $\begin{cases} x_1.n.u \equiv x_1[m] \\ x_2.m.v \equiv x_2[n] \end{cases}$
Finalement, $x = x_1.n.u + x_2.m.v$ convient.

Exercice 5. *Realiser la procédure* `Reste_Chinois_Constructif_2var` *implémentant l'algorithme décrit précédemment. Tester votre procédure sur quelques exemples.*

Procedure `Reste_Chinois_Constructif_2var`(x_1,x_2,n,m)

Input: (n, m, x_1, x_2) / pgcd(n,m)=1

Output: $x \in \mathbb{Z}/nm\mathbb{Z}$ / $\begin{cases} x \equiv x_1 \bmod n \\ x \equiv x_2 \bmod m \end{cases}$

3.2 (pour le sport) Généralisation à r variables

Principe :
Soient $n_1, \ldots, n_r$ des entiers premiers entre eux deux à deux.
Alors, d'après le théorème des restes chinois, $\exists x \in \mathbb{Z}/n_1 \ldots n_r\mathbb{Z}$ tel que :

$$\forall (x_1, \ldots, x_r) \in \mathbb{Z}/n\mathbb{Z} \times \mathbb{Z}/m\mathbb{Z}, \begin{cases} x \equiv x_1 \bmod n_1 \\ \vdots \\ x \equiv x_r \bmod n_r \end{cases}$$

Le but est de construire à partir de la donnée de $n_1, \ldots, n_r$ et de $x_1, \ldots, x_r$ un x qui convient.
Idee : On pose :

$$M_i = \frac{\prod_{j=1}^{r} n_j}{n_i}$$

Les M_i sont alors premiers entre eux deux à deux.
Il existe donc $(a_1, \ldots, a_r)$ tels que $a_1 M_1 + \ldots + a_r M_r = 1$.
Plus précisément, $\forall i \in [1, r], a_i = M_i^{-1} \bmod n_i$
En particulier, $\forall i \in [1, r], a_i.M_i.x_i = x_i \bmod n_i$ et $x = \sum_{i=1}^{r} a_i.M_i.x_i$ convient.

Exercice 6. *Realiser la procédure* `Reste_Chinois_Constructif` *implémentant cet algorithme. Tester votre procédure sur quelques exemples.*

Procedure `Reste_Chinois_Constructif`$([n_1, \ldots, n_r], [x_1, \ldots, x_r])$

Input: $[n_1, \ldots, n_r], [x_1, \ldots, x_r] / pgcd(n_1, \ldots, n_r) = 1$

Output: $x \in \mathbb{Z}/n_1 \ldots n_r\mathbb{Z}$ / $\begin{cases} x \equiv x_1 \mod n_1 \\ \vdots \\ x \equiv x_r \mod n_r \end{cases}$
