UNIVERSITE DE LIMOGES

Laco

Laboratoire d'Arithmétique,
de Calcul formel et d'Optimisation

irem

Institut de Recherche
sur l'Enseignement des Mathématiques

Apprenons l'Arithmétique Élémentaire pour comprendre la Cryptographie Moderne

Guy ROBIN

Publication de l'IREM de LIMOGES **mai 1998**

Apprenons

l' Arithmétique Elémentaire

pour comprendre la

Cryptographie Moderne

Guy Robin

LACO, Laboratoire d'Arithmétique, de Calcul formel et d'Optimisation.

Edité par l'I.R.E.M. de l'Université de Limoges

mai 1998

Table des matières

Préface 7

Notations 11

1 Cryptographie élémentaire 13
- 1.1 Premiers éléments de cryptographie . . . 13
  - 1.1.1 Le chiffrement . . . 13
  - 1.1.2 Pourquoi l'expression "chiffrer" ? . . . 14
- 1.2 Autres exemples . . . 16
- 1.3 Algorithmes de chiffrement . . . 17
  - 1.3.1 Définition . . . 17
  - 1.3.2 L'algorithme "PLUS" . . . 18
  - 1.3.3 L'algorithme "MULT" . . . 19
- 1.4 Historique . . . 19
  - 1.4.1 Scytale . . . 19
  - 1.4.2 Permutation . . . 20

2 Diviseurs 21
- 2.1 Diviseurs et multiples dans $\mathbb{N}$ . . . 21
- 2.2 Le pgcd dans $\mathbb{N}$ . . . 22
- 2.3 Le ppcm dans $\mathbb{N}$ . . . 23
- 2.4 Divisibilité dans $\mathbb{Z}$ . . . 23
- 2.5 Généralisations . . . 24

3 La division euclidienne 25
- 3.1 La division euclidienne. . . . 25
- 3.2 L'algorithme d'Euclide . . . 26
- 3.3 Retour sur le pgcd et le ppcm . . . 28
- 3.4 Le théorème de Gauss . . . 29
  - 3.4.1 Le théorème . . . 29
  - 3.4.2 Ses conséquences . . . 30
- 3.5 Preuve constructive du théorème de Bézout . . . 31
- 3.6 Exemples . . . 33
- 3.7 Algorithme . . . 34
- 3.8 Compléments . . . 35

3.8.1 Étude des couples (x, y) qui vérifient la relation de Bézout. 35
3.8.2 Résolution de $ax + by = c$ 36

4 Les nombres premiers **39**
4.1 Premières propriétés 39
4.2 Le crible d'Eratosthène 40
4.3 Factorisation 42
4.4 Encore le pgcd et le ppcm 44

5 Numération **47**
5.1 La représentation que l'on connaît 47
5.2 D'autres systèmes de position 48
5.3 D'autres systèmes de numération 48
5.3.1 Le système grec du troisième siècle avant J.-C. 48
5.3.2 Le système romain 49
5.4 La notion de base 49
5.5 Changement de base 51
5.5.1 À la main 51
5.5.2 Les algorithmes 52

6 Calcul modulaire **55**
6.1 Introduction 55
6.2 La notion de modulo 56
6.3 Les classes 58
6.4 Retour au théorème de Gauss 59
6.5 Les correspondances entre classes 59
6.5.1 Première correspondance 60
6.5.2 Deuxième correspondance 60
6.5.3 Algorithme de calcul de l'inverse modulaire 62
6.5.4 Retour à l'exemple de l'introduction 63
6.6 Le petit théorème de Fermat 63
6.6.1 Le théorème 63
6.6.2 Ses corollaires 64
6.7 Caractères de divisibilité 66
6.8 Preuve d'opération 66

7 Retour à la cryptographie **69**
7.1 Retour sur le chapitre un 69
7.2 Généralisation à $\mathcal{E}_n$ 69
7.2.1 Les applications affines 70
7.3 Notion de clé 71
7.4 Le chiffrement de Vernam 72
7.5 Un algorithme de chiffrement plus mathématique 73
7.5.1 Un exemple 73
7.5.2 Le cadre général 73

7.5.3 Un autre exemple 74
7.6 L'algorithme RSA 74
7.6.1 Le cadre général 74
7.6.2 Un exemple simple 75
7.6.3 Un autre exemple 76
7.6.4 Sécurité de l'algorithme 76
7.7 Notion de signature 77
7.7.1 La signature 77
7.7.2 Signature et chiffrement 77

Pour en savoir plus **79**

Annexe **81**

Préface

Apprenons l'arithmétique élémentaire pour comprendre la cryptographie moderne

Cartes bancaires, codes secrets, communications sur Internet et bientôt cartes de santé et monnaie électronique, sécurité des transmissions, sécurité informatique, sécurité des réseaux, sécurité des transactions commerciales...,
tous ces noms vous les lisez dans vos journaux, vous les entendez à la radio ou à la télévision. Pour toute la sécurité logicielle - c'est-à-dire celle qui ne résulte pas d'un dispositif physique - il est nécessaire de faire appel aux **méthodes cryptographiques modernes.**

La cryptographie fut d'abord l'art de dissimuler à autrui la compréhension d'un message. Ainsi Jules César (100 - 44 av. J.C.) lorsqu'il transmettait un message à ses généraux utilisait-il le code ou chiffre suivant : chaque lettre du message était décalée de trois lettres. "a" était remplacé par "d", "b" par "e" et ainsi de suite. Ainsi "adversaire" était transformé en "dgyhuvdluh".

Cette cryptographie élémentaire, utilisée essentiellement par les militaires et les diplomates, est presque restée en l'état jusqu'en 1976.

En 1976, deux chercheurs américains, Whitfield Diffie et Martin Hellman, dans un article devenu célèbre, ont posé la question de savoir s'il était possible que la cryptographie soit publique. Jusqu'alors tout était secret, l'expéditeur d'un message et son destinataire devaient s'être préalablement entendu sur une méthode à utiliser et sur une clé qui tenait lieu de paramètre (le chiffre des diplomates). Les transactions qui s'effectuaient autrefois par lettres se réalisent, maintenant, de plus en plus par communications électroniques. Il est devenu indispensable alors de sécuriser les transmissions, d'authentifier le contenu d'un message, d'identifier l'expéditeur du message et ceci ne peut bien se faire que par un chiffrement à clé publique.

Aujourd'hui, si vous voulez téléphoner à quelqu'un, vous consultez un annuaire et à moins que votre correspondant ne soit sur la liste rouge - son numéro de téléphone est

alors un secret entre lui et vous - vous obtenez son numéro et pouvez le joindre. Peut-on réaliser ceci en cryptographie ? Est-il possible de chiffrer à l'aide de renseignements publics que chacun peut lire sur un annuaire ? Est-il possible qu'une personne interceptant votre message ne puisse pas le décrypter bien qu'elle sache comment il a été chiffré et que seul le destinataire puisse le déchiffrer ? La réponse est positive et fut donnée assez rapidement par trois autres chercheurs et elle est connue sous le nom de "méthode RSA".

Que faut-il connaître en arithmétique, pour comprendre la cryptographie moderne, (au moins la méthode RSA) et pour bien expliquer la cryptographie conventionnelle, celle d'avant 1976 ?

Nous devons commencer par introduire la notion de diviseur d'un nombre et parler de pgcd, notion fondamentale.

Nous devons définir la division euclidienne et donner ses applications dont la plus importante pour nous est ici le calcul de l'inverse d'un entier modulo un autre entier.

Nous devons introduire les nombres premiers qui, d'une part sont à la base de toute l'arithmétique et qui, d'autre part, jouent pour la méthode RSA un rôle fondamental. On sait facilement multiplier deux entiers entre eux et l'utilisation d'un ordinateur nous facilite le calcul. Une fois le résultat obtenu, si l'on veut retrouver les nombres que l'on a multiplié, c'est beaucoup plus difficile, même si l'on dispose d'un bon ordinateur et d'une bonne méthode. Ainsi essayez, à la main, de factoriser un nombre de 10 ou 20 chiffres ! Actuellement, factoriser un entier de 150 chiffres, produit de deux nombres premiers bien choisis, est pratiquement impossible, et cela, en utilisant les ordinateurs les plus puissants.

Mais la notion la plus fondamentale à étudier est celle de "congruence modulo un entier", notion introduite par Gauss au dix-neuvième siècle. Dans la vie courante, chacun sait que deux heures après 23 heures, il est une heure. Chacun a transformé $23 + 2 = 25$ en 1. Chacun a travaillé modulo 24, en enlevant 24 à 25. Quarante-huit heures après 3 heures, il est encore 3 heures. 3 heures et $(48 + 3)$ heures sont considérées dans la même classe. Cette notion doit être approfondie car elle est constamment utilisé en Cryptographie.

La cryptographie moderne a donné un coup de fouet à l'arithmétique car elle a besoin de résultats effectifs. Au lieu d'énoncer : il existe n vérifiant la propriété "$\mathcal{P}$" et de faire la démonstration, il est utile maintenant de trouver effectivement un "n" répondant à la question et si possible d'étudier tous les entiers vérifiant la propriété "$\mathcal{P}$".

Mais la nécessité de calculs effectifs va de pair avec l'utilisation de calculettes et d'ordinateurs, car les nombres entiers deviennent vite grands. Ainsi, en Cryptographie, les entiers utilisés ont plus de 50 chiffres. On les appelle des grands entiers et des calculs exacts sur ces grands entiers sont indispensables. Les chercheurs ont dû mettre au point des algorithmes de calculs performants sur ces nombres. Seuls, jusqu'aux années 80, les

physiciens et les astronomes utilisaient de grands nombres mais, en réalité, ils ne faisaient que des calculs approchés. La construction de logiciels de calculs exacts a été nécessaire. L'utilisation des cartes à puces a conduit les chercheurs à trouver des méthodes performantes. Ces recherches sont vraiment d'actualité.

Pour aller plus loin dans l'étude de la cryptographie, on ne peut se limiter aux entiers de $\mathbb{Z}$. Il faut généraliser aux entiers des corps de nombres, essentiellement aux nombres quadratiques, c'est-à-dire aux nombres de la forme $a+b\sqrt{d}$, pour $a, b, d \in \mathbb{Z}$. Il faut aussi apprendre des éléments d'algèbre, plus particulièrement sur les corps finis, des éléments d'algèbre linéaire, de probabilité, de statistique, d'algorithmique,...

La cryptographie est maintenant devenue une Science, la science de la confiance, partie prenante de la théorie mathématique de l'Information qui comprend aussi le codage de l'information ainsi que les codes correcteurs permettant de corriger les erreurs de transmission, la compression des données, ...

Notations

$\mathbb{N}$: l'ensemble des entiers naturels soit $\{0, 1, 2, \ldots\}$

$\mathbb{N}^*$: l'ensemble des entiers naturels non nuls.

$\mathbb{Z}$: l'ensemble des entiers relatifs soit $\{\ldots, -2, -1, 0, 1, 2, \ldots\}$.

$\mathbb{Z}^*$: l'ensemble des entiers relatifs non nuls.

$:=$ est le signe d'affectation
$x := 3$ affecte la valeur 3 à la variable x;
$x := y + 2$ donne à la variable x la valeur de la variable y augmentée de 2.

Pour la multiplication des nombres, on utilisera la notation $*$ ou l'absence de notation, ainsi :

$$10 * 15 = 150$$

$$10 * a = 10a$$

$$ab = a * b$$

Chapitre 1

Cryptographie élémentaire

Où l'on parle de cryptographie élémentaire, où l'on montre qu'il est nécessaire de faire un peu de mathématiques pour simplifier les méthodes utilisées et où l'on s'aperçoit qu'il est nécessaire d'en connaître encore plus pour pouvoir déchiffrer.

1.1 Premiers éléments de cryptographie

Responsable d'un service de sécurité, vous recevez le message ou **cryptogramme** suivant:

TGPFGBXQWUXGPFTGFKUQKT

Ce message a été **chiffré** par votre correspondant pour que vous soyez le seul à pouvoir connaître sa signification. Vous devez donc être capable de **déchiffrer** pour obtenir le message de votre correspondant. Ce messsage, qu'on appelle **clair**, est ici :

RENDEZ-VOUS VENDREDI SOIR.

Précisons cela :

1.1.1 Le chiffrement

Le **chiffreur** a fait subir au clair une transformation, lettre par lettre. Cette transformation consiste simplement à remplacer A par C, B par D, et ainsi de suite jusqu'à X par Z, puis Y par A et Z par B. Regardez le cercle ci-dessous. Cette tranformation **"PLUS2"** permet d'obtenir le cryptogramme suivant :

TGPFGBXQWUXGPFTGFKUQKT

à partir du clair :

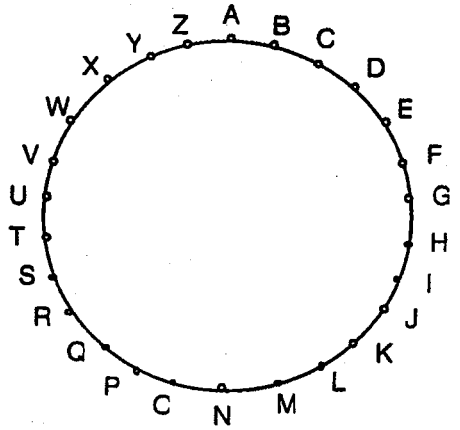
RENDEZVOUSVENDREDISOIR

Nous avons utilisé la transformation d'alphabet suivante :

$$\begin{pmatrix} A & B & C & D & E & F & G & H & I & J & K & L & M & N & O & P & Q & R & S & T & U & V & W & X & Y & Z \\ \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow \\ C & D & E & F & G & H & I & J & K & L & M & N & O & P & Q & R & S & T & U & V & W & X & Y & Z & A & B \end{pmatrix}$$

Pour **déchiffrer**, comme vous pouvez le deviner tout seul, il faut faire un décalage des lettres de l'aphabet de deux vers la gauche, c'est la transformation "**MOINS2**", explicitée comme suit :

$$\begin{pmatrix} A & B & C & D & E & F & G & H & I & J & K & L & M & N & O & P & Q & R & S & T & U & V & W & X & Y & Z \\ \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow \\ Y & Z & A & B & C & D & E & F & G & H & I & J & K & L & M & N & O & P & Q & R & S & T & U & V & W & X \end{pmatrix}$$



Celui qui ne connaît pas la méthode de chiffrement doit **décrypter**; il doit deviner la façon dont le message a été transformé. Dans ce premier exemple, évidemment, ce n'est pas trop difficile.

1.1.2 Pourquoi l'expression "chiffrer" ?

Tout simplement parce qu'il est plus facile d'utiliser dans les transformations les chiffres et les nombres que les lettres. En effet, affectons à chaque lettre de l'alphabet son numéro d'ordre, comme le montre le tableau ci-après. Remarquez que l'on code les dix premiers entiers avec deux chiffres en mettant un 0 devant.

$$\begin{pmatrix} A & B & C & D & E & F & G & H & I & J & K & L & M & N & O & P & Q & R & S & T & U & V & W & X & Y & Z \\ \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow \\ 00 & 01 & 02 & 03 & 04 & 05 & 06 & 07 & 08 & 09 & 10 & 11 & 12 & 13 & 14 & 15 & 16 & 17 & 18 & 19 & 20 & 21 & 22 & 23 & 24 & 25 \end{pmatrix}$$

Si l'on transforme notre message :

RENDEZ-VOUS VENDREDI SOIR

nous obtenons le message écrit dans une autre langue, celle qui utiliserait les 26 premiers nombres entiers comme alphabet. Pour faciliter la lecture, nous décomposons le message en sous messages de 5 lettres :

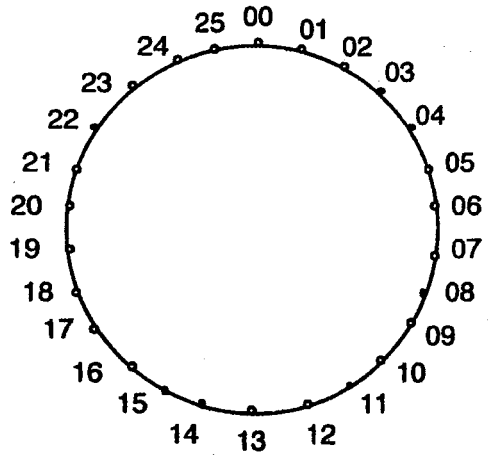
RENDE	*ZVOUS*	*VENDR*	*EDISO*	*IR*
1704130304	2521142018	2104130317	0403081814	0817

Ce dernier message est peu clair, mais on ne considère cependant pas que c'est un cryptogramme; il est seulement écrit avec de nouveaux signes : on dit que c'est un **codé** du clair. La transformation "*PLUS2*" est donc la suivante :

$$\begin{pmatrix} 00 & 01 & 02 & 03 & 04 & 05 & 06 & 07 & 08 & 09 & 10 & 11 & 12 & 13 & 14 & 15 & 16 & 17 & 18 & 19 & 20 & 21 & 22 & 23 & 24 & 25 \\ \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow \\ 02 & 03 & 04 & 05 & 06 & 07 & 08 & 09 & 10 & 11 & 12 & 13 & 14 & 15 & 16 & 17 & 18 & 19 & 20 & 21 & 22 & 23 & 24 & 25 & 00 & 01 \end{pmatrix}$$

Avec les nombres, le chiffrement est plus aisé. Dans notre exemple "PLUS2", on ajoute 2 à chaque nombre de 2 chiffres avec la convention, facile à lire sur le dessin, que $24+2$ donne 00 et $25+2$ donne 01. C'est cela que l'on mathématisera au chapitre 6. Notre clair codé et le cryptogramme sont donc :

1704130304	2521142018	2104130317	0403081814	0817
1906150506	2723162220	2306150519	0605102016	1019.



Quelques exercices pour vous entraîner :

1. Comparer "MOINS1" et "PLUS25", "PLUSa" et "MOINS (26 - a)" pour $a \in \{1, 2, \ldots, 25\}$.
2. Chiffrer le clair du texte avec "PLUS7", "PLUS30", "MOINS4". Déchiffrer les cryptogrammes obtenus.
3. Décrypter

   UFW KFN YAT ZXF AJE HTR UWN X

1.2 Autres exemples

1. Prenons un autre exemple **"MULT3"**.

   Nous allons le présenter avec les lettres et les chiffres . Aidez-vous des dessins. Cette méthode, consiste à multiplier les positions des lettres par 3. Ainsi, la lettre "A" en position zéro n'est pas transformée, ensuite "B" (position 1) est transformée en "D" (position 3),.., c'est à dire que à chaque lettre on décale de 3 positions, en lisant sur le cercle. L'image de "I" est "Y", celle de "J" est "B". La transformation d'alphabet est donc la suivante :

$$\begin{pmatrix} A & B & C & D & E & F & G & H & I & J & K & L & M & N & O & P & Q & R & S & T & U & V & W & X & Y & Z \\ \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow \\ A & D & G & J & M & P & S & V & Y & B & E & H & K & N & Q & T & W & Z & C & F & I & L & O & R & U & X \end{pmatrix}$$

   Ecrivons aussi cette transformation à l'aide des nombres :

$$\begin{pmatrix} 00 & 01 & 02 & 03 & 04 & 05 & 06 & 07 & 08 & 09 & 10 & 11 & 12 & 13 & 14 & 15 & 16 & 17 & 18 & 19 & 20 & 21 & 22 & 23 & 24 & 25 \\ \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow \\ 00 & 03 & 06 & 09 & 12 & 15 & 18 & 21 & 24 & 01 & 04 & 07 & 10 & 13 & 16 & 19 & 22 & 25 & 02 & 05 & 08 & 11 & 14 & 17 & 20 & 23 \end{pmatrix}$$

   Notre message codé

   1704130304 2521142018 2104130317 0403081814 0817,

   puis chiffré, donne le cryptogramme

   2512130912 2311160802 1112130925 1209240216 2425,

   que l'on peut écrire :

   ZMN JMX LQI CLM NJZ MJY CQY Z.

   Comme vous l'avez remarqué, il est beaucoup plus facile d'utiliser les nombres que les lettres; ainsi chiffrer la lettre O est difficile alors que le chiffrement de 14, qui est sa représentation chiffrée, s'obtient facilement. En effet, on calcule $3 * 14 = 42$. Il faut compter 42 pas sur le cercle. Nous commençcons à 1 puis 2,..., on arrive à 25 et l'on continue sur le cercle. Il reste encore 17 pas à faire et comme le premier pas est noté 0, on termine sur 16, ce qui traduit l'équation $42 = 26 + 16$. Pour la transformation de V, on calcule $3 * 21 = 63$. On doit faire deux tours sur le cercle et avancer encore de 11 pas. On a $63 = 26 * 2 + 11$.

2. Prenons un autre exemple **"MULT9"**. La transformation est la suivante :

$$\begin{pmatrix} 00 & 01 & 02 & 03 & 04 & 05 & 06 & 07 & 08 & 09 & 10 & 11 & 12 & 13 & 14 & 15 & 16 & 17 & 18 & 19 & 20 & 21 & 22 & 23 & 24 & 25 \\ \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow \\ 00 & 09 & 18 & 01 & 10 & 19 & 02 & 11 & 20 & 03 & 12 & 21 & 04 & 13 & 22 & 05 & 14 & 23 & 06 & 15 & 24 & 07 & 16 & 25 & 08 & 17 \end{pmatrix}$$

Comme on peut le vérifier "MULT9" permet de déchiffrer tout cryptogramme chiffré par "MULT3" et inversement d'ailleurs. Ainsi l'image de 04 par "MULT9" est 10 et l'image de 10 par "MULT3" est 04.

Encore quelques exercices

(a) Chiffrer le clair du texte à l'aide de "MULT7".

(b) Décrypter

KUN UMR XAM UNK SHU RHS XPO ZZO KOD U

3. Pour compliquer, nous pouvons utiliser les deux transformations précédentes successivement; nous allons faire du **"surchiffrage"**. Faisons d'abord "MULT3" suivi de "PLUS2". La transformation d'alphabet est la suivante :

$$\begin{pmatrix} A & B & C & D & E & F & G & H & I & J & K & L & M & N & O & P & Q & R & S & T & U & V & W & X & Y & Z \\ \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow \\ C & F & I & L & O & R & U & X & A & D & G & J & M & P & S & V & Y & B & E & H & K & N & Q & T & W & Z \end{pmatrix}$$

4. Réalisons maintenant "PLUS2" suivi de "MULT3".

$$\begin{pmatrix} A & B & C & D & E & F & G & H & I & J & K & L & M & N & O & P & Q & R & S & T & U & V & W & X & Y & Z \\ \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow \\ G & J & M & P & S & V & Y & B & E & H & K & N & Q & T & W & Z & C & F & I & L & O & R & U & X & A & D \end{pmatrix}$$

Comme nous le voyons, nous n'obtenons pas la même transformation que dans le cas précédent. Nous expliquerons pourquoi dans les chapitres ultérieurs et nous montrerons comment on peut obtenir facilement la façon de déchiffrer.

5. Pour terminer considérons **"MULT2"**. L'image de 00 est 00 , celle de 13 est aussi 00. Ainsi 00 provient de deux nombres possibles : il y a donc ambiguité dans le déchiffrement, d'autant plus que tous les nombres obtenus ont deux antécédents. Les nombres obtenus sont pairs et comme vous pouvez le vérifier, le nombre $2N$ avec $0 \leq N < 13$ est l'image de N et de $N + 13$. Ainsi "MULT2" ne convient pas car $2 * 13 = 26$. De même "MULT13" ne peut convenir. Les entiers 2 et 13 sont des diviseurs de 26. Cette notion est étudiée au chapitre suivant.

1.3 Algorithmes de chiffrement

1.3.1 Définition

Nous avons jusqu'à présent utilisé l'expression "méthode de chiffrement". On préfère de nos jours dire **"algorithme de chiffrement"**, puisque, en effet, le chiffrement entre bien dans le cadre général de cette notion. Nous n'avons pas encore de définition mathématique du vocable **"algorithme"**. Disons que c'est un procédé calculatoire systématique, que l'on peut programmer aisément afin qu'un ordinateur fournisse le résultat. "Algorithme" vient du nom du mathématicien arabe Al Khuwãrizmĩ qui vivait à Bagdad (763 - 850). Il

donna aussi, à la postérité le substantif **algèbre**, obtenu à partir du titre de son ouvrage Al Jabr.

Donnons deux exemples de présentation d'algorithme.

1.3.2 L'algorithme "PLUS"

Algorithme 1 (Chiffrement "PLUS")
Entrée : *un entier a et la suite des lettres codées du clair notées X, terminée par 26.*
Sortie : *la suite des lettres codées, notées Y, du cryptogramme.*
début
Soit X les 2 premiers chiffres du clair;
Tant que $X \leq 25$
faire
$Y := X + a$;
Si $Y > 25$ *alors* $Y := Y - 26$; *imprimer Y; finsi*
Soit X les deux chiffres suivants du clair;
refaire
fin

Commentaires
Faisons quelques remarques sur la construction d'un algorithme en général et sur celui-ci en particulier.

- Tant que $\mathcal{P}$
  faire
  corps d'instructions
  refaire
  est une **boucle**. Ceci signifie que, tant que la proposition $\mathcal{P}$ est vraie, on exécute le programme qui se trouve entre **faire** et **refaire**. Après l'instruction placée juste avant "refaire", on doit se placer à l'instruction "tant que". Dès que la proposition devient fausse, on sort de la boucle, en se plaçant à l'instruction qui suit immédiatement **refaire**.

- ":=" est une affectation. $Y := X + a$ signifie que l'on affecte à Y la somme des valeurs des variables X et a. Notez que l'écriture $Y := Y + a$ est correcte ; elle signifie que la nouvelle valeur de Y est l'ancienne à laquelle est ajoutée la valeur de a.

- Le symbole "Si" se comprend aisément sous la forme simplifiée utilisée ici. La ligne n'est exécutée que si la valeur de Y dépasse strictement 25. On termine l'instruction par "finsi".

- On termine chaque instruction par un ";" pour les séparer les unes des autres.

- Expliquons maintenant le déroulement de notre algorithme.
  Nous désirons chiffrer "071324" par "PLUS5". Nous utilisons l'algorithme PLUS à partir des données : $a = 5$ et le clair 07132426. On a d'abord $X = 07$ donc Y = 07 + 5 = 12. Comme $Y \leq 25$, on imprime la valeur de Y, soit 12. Ensuite, on lit 13 et l'on se place au début de la boucle. Y vaut alors 18 et l'on imprime 18. On lit alors $X = 24$, d'où $Y = 29$. Comme $Y > 25$ le programme calcule $Y = 29 - 26 = 3$ et on imprime 3. On lit alors $X = 26$, ce qui nous fait sortir de la boucle et termine l'exécution de l'algorithme.

1.3.3 L'algorithme "MULT"

Algorithme 2 (Chiffrement "MULT")
Entrée : *un entier a et la suite des lettres codées du clair notées X, terminée par 26.*
Sortie : *la suite des lettres codées, notées Y, du cryptogramme.*
début
Soit X les 2 premiers chiffres du clair;
Tant que $X \leq 25$
faire
  $Y := X * a$;
  Tant que $Y > 25$
  faire
    $Y := Y - 26$;
  refaire
  Soit X, les deux chiffres suivants du clair;
refaire
fin

Commentaires

- Il y a une boucle "tant que " imbriquée dans une autre, ceci ne doit pas poser problème.

- Exécuter l'algorithme " MULT" avec le clair précédent et $a = 7$.

1.4 Historique

1.4.1 Scytale

La scytale des Spartiates consistait en un bâton sur lequel était enroulée une lanière de cuir. L'expéditeur écrivait son message sur la lanière enroulée. Celle- ci déroulée portait un texte inintelligible. Le destinataire pouvait retrouver le message en enroulant la lanière sur un bâton de même diamètre.

1.4.2 Permutation

Permuter les lettres d'un message brouille le message. A partir de cette idée de nombreuses méthodes ont été utilisées au cours des siècles.
Nous prenons notre message

RENDEZ-VOUS VENDREDI SOIR.

et donnons quelques exemples.

- On peut renverser les lettres 3 par 3; REN devenant NER et notre message chiffré est :

  NERZEDUOVEVSRNRIDEIOSR.

  Généralisez.

- On écrit le message sur deux rails

  $$\begin{array}{cccccccccccc} R & N & E & V & U & V & N & R & D & S & I \\ E & D & Z & O & S & E & D & E & I & O & R \end{array}$$

  et on relève ligne après ligne pour obtenir :

  RNEVUVNRDSIEDZOSEDEIOR.

  Et l'on peut prendre plus de deux rails.

- On écrit le message dans un tableau rectangulaire de largeur fixée et on relève par colonne à partir d'un ordre fixé.

  $$\begin{array}{ccc} R & E & N \\ D & E & Z \\ V & O & U \\ S & V & E \\ N & D & R \\ E & D & I \\ S & O & I \\ R & & \end{array}.$$

  En relevant d'abord la seconde colonne puis la troisième, puis la première, il vient

  ODDVOEERDVSNESRIIREUZN.

  Comme vous le remarquez, on a aussi relevé les colonnes de bas en haut ou de haut en bas.

Chapitre 2

Diviseurs

Où l'on introduit les diviseurs et les multiples d'un entier ainsi que la notion si fondamentale de pgcd : le plus grand commun diviseur à deux entiers.

2.1 Diviseurs et multiples dans N

On remarque que $2 * 13 = 26$. On dit que 2 et 13 divisent 26 et aussi que 26 est multiple de 2 et de 13.

Définition 1 *Soient a et b deux entiers, on dit que a* **divise** *b et l'on écrit $a \mid b$ si et seulement si il existe un entier c tel que $b = c * a$. On dit aussi que b est un* **multiple** *de a.*

Exemples

- 2 divise 72 car on peut écrire $72 = 36 * 2$. Cette égalité montre aussi que 36 divise 72.
- 1 divise tout nombre entier a puisque $a = a * 1$.
- a divise a puisque $a = a * 1$.
- a divise 0 ; en effet $0 = a * 0$.
- 1 a un seul diviseur 1.
- Tout entier a différent de 1 a au moins deux diviseurs a et 1.
- 0 ne divise que 0 car pour tout a, $a * 0 = 0$.

Lorsque b divise a, il n'existe en général qu'un seul entier c tel que $a = bc$. C'est le cas si $b \neq 0$; en effet dans ce cas une égalité telle que $bc = bc'$ donne $c = c'$. Si $b = 0$ alors $a = 0$ et tout nombre c convient.

Définition 2 *Soient a et b deux entiers tels que $b \neq 0$. Supposons que b divise a. On appelle* **quotient exact** *de a par b l'unique c tel que $a = bc$. On le note $\frac{a}{b}$.*

Définition 3 *On appelle* **nombre premier** *tout entier ayant exactement deux diviseurs. Ces deux diviseurs sont 1 et lui-même.*

- 13 est premier.
- 26 est non premier puisque 26 possède 4 diviseurs, à savoir $1, 2, 13$ et 26.
- 1 n'est pas premier puisqu'il n'a qu'un seul diviseur ; 0 n'est pas premier car tous les entiers le divisent.
- Les nombres premiers seront étudiés au chapitre 4.

Propriétés

1. Si a divise b et si b divise c alors a divise c.
   Démonstration
   On peut écrire $b = xa$ et $c = yb$ donc $c = (yx)a$ et a divise c. □

2. Si a divise b et b divise a alors $a = b$.
   Démonstration On peut écrire $b = xa$ et $a = yb$ donc $a = (yx)a$. Si a n'est pas nul, on peut diviser par a et obtenir $yx = 1$, soit $x = 1$ et $y = 1$, et par suite $a = b$.
   Si $a = 0$, l'égalité $b = xa$ montre que $b = 0$ et donc $a = b$. □

3. Si d divise a et b alors d divise tout entier de la forme $ra + sb$, r et s étant deux entiers.
   Démonstration On peut écrire $a = xd$ et $b = yd$ donc

   $$ra + sb = rxd + syd = (rx + sy)d,$$

   ce qui montre que d divise $ra + sb$. □

2.2 Le pgcd dans N

Les diviseurs de 18 sont $1, 2, 3, 6, 9, 18$. Ceux de 60 sont $1, 2, 3, 4, 5, 6, 10, 12, 15, 20, 30, 60$. Les diviseurs communs sont $1, 2, 3, 6$, et 6 est le plus grand. De façon générale, comme le nombre de diviseurs communs est fini, il y a toujours un plus grand diviseur commun.

Définition 4 *Le* **plus grand commun diviseur** *de deux entiers a et b, non nuls tous les deux, est le plus grand entier divisant à la fois a et b. La notation utilisée est* $\mathbf{pgcd}(a, b)$.

Exemples pgcd(18, 60) = 6, pgcd(36, 82) = 2, pgcd(36, 72) = 36, pgcd(0, 36) = 36, pgcd(1, 40) = 1.

Remarques

- Si a et b sont nuls tous les deux, tous les nombres sont leurs diviseurs communs, aussi, par convention, on pose : pgcd(0, 0) = 0.
- Cette notion de pgcd est très importante mais les principales propriétés ne pourront être établies qu'au chapitre suivant après avoir étudié l'algorithme d'Euclide.

Définition 5 *Deux entiers naturels a et b sont dits* **premiers entre eux** *si leur pgcd vaut* 1. *On dit encore que a* **est premier avec** *b ou encore que a* **est relativement premier à** *b.*

Exemple 25 et 26 sont premiers entre eux.
Remarque Il ne faut pas confondre les deux notions "premier" et "premiers entre eux" et surtout les deux expressions "a est premier" et "a est premier avec b". L'exemple montre que deux nombres non premiers peuvent être premiers entre eux.

2.3 Le ppcm dans N

Considérons maintenant les multiples non nuls de 18,

$$18, 36, 48, 72, 90, 108, 126, 144, 162, 180, 198, \ldots$$

et les multiples non nuls de 30,

$$30, 60, 90, 120, 150, 180, 210, 240, \ldots$$

Les multiples communs, non nuls, sont

$$90, 180, \ldots.$$

Ils constituent un ensemble infini mais ce qui nous intéresse ici c'est le plus petit de ces multiples, c'est 90.
Si l'un des entiers est nul, seul zéro peut être un multiple commun.

Définition 6 *Le* **plus petit commun multiple** *à deux entiers a et b, non nuls est le plus petit entier non nul, multiple à la fois de a et b. On le note* $\mathbf{ppcm}(a, b)$.
Si l'un des entiers est nul, on posera $\mathbf{ppcm}(a, 0) = 0$ *quel que soit a.*

Exemples ppcm(6, 82) = 246, ppcm(36, 72) = 72, ppcm(36, 0) = 0, ppcm (1, 82) = 82, ppcm(0, 0) = 0.

2.4 Divisibilité dans $\mathbb{Z}$

Il n' est pas toujours possible de nous limiter aux entiers naturels, aussi devons nous prolonger les définitions précédentes aux entiers relatifs.

Définition 7 *Soient a et b deux entiers relatifs, on dit que* **a divise b** *et on écrit $a \mid b$ si et seulement s'il existe un entier relatif c tel que $b = c * a$. On dit aussi que b est un* **multiple** *de a.*

Ainsi 3 divise -6, -5 divise 25, 5 divise -5. On remarque que a divise b si et seulement si la valeur absolue de a divise celle de b, ce qui permet de se ramener aux entiers naturels. Les propriétés 1 et 3 du paragraphe précédent sont encore vérifiées. La conclusion de la propriété 2 doit s'écrire $a = \pm b$.
On peut définir le pgcd et le ppcm de deux entiers de $\mathbb{Z}$, comme étant le pgcd et le ppcm de leurs valeurs absolues, mais ceci n'est pas utile pour nous dans la suite.

2.5 Généralisations

Définition 8 *On définit le pgcd de plusieurs entiers naturels comme étant leur plus grand diviseur.*

Exemples

- $\text{pgcd}(6, 8, 9) = 1$, $\text{pgcd}(20, 22, 24, 26) = 2$.

- Dès que l'un des entiers est nul, le pgcd est nul.

La propriété suivante est simple à démontrer :
Propriété Pour calculer le pgcd de plusieurs entiers, on peut remplacer deux d'entre eux par leur pgcd, soit

$$\text{pgcd}(a, b, c) = \text{pgcd}(\text{pgcd}(a, b), c).$$

Démonstration Soit $d = \text{pgcd}\ (a, b, c)$, alors d divise a et b et donc d divise pgcd (a, b), par suite d divise $\delta = \text{pgcd}\ (\text{pgcd}\ (a, b), c)$. Inversement δ divise pgcd (a, b) et c, donc δ divise a, b, c et donc d.

Définition 9 *Plusieurs nombres sont dits premiers entre eux dans leur ensemble si leur pgcd vaut 1.*

Exemple C'est le cas des trois entiers $6, 8, 9$.

Définition 10 *Plusieurs nombres sont premiers entre eux deux à deux, si, pris deux à deux, ils sont premiers entre eux.*

Exemples $6, 8, 9$ ne sont pas premiers entre eux deux à deux puisque 6 et 8 ne sont pas premiers entre eux, mais $16, 15, 49, 143$ sont premiers entre eux deux à deux, comme on le vérifie en calculant six pgcd.

Propriété Si des nombres sont premiers entre eux deux à deux, ils sont premiers entre eux dans leur ensemble.

Chapitre 3

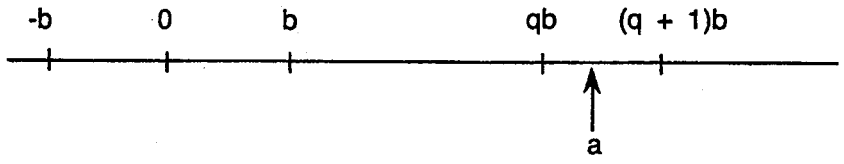
La division euclidienne

Où l'on vous fait part des travaux d'Euclide, travaux qui sont toujours et peut-être de plus en plus d'actualité et à la base de nombreuses recherches. Où l'on montre que l'algorithme d'Euclide permet de réaliser des calculs indispensables pour bien travailler en cryptographie.

Euclide, mathématicien grec, vivait au troisième siècle avant J.C. à Alexandrie en Egypte actuelle, sous domination grecque alors. Par la rigueur de ses méthodes, par la clarté de ses démonstrations, ses ouvrages ont servi de modèle pendant plus de deux millénaires. Nous allons étudier dans ce chapitre ce qu'on appelle la division euclidienne, c'est à dire la division avec reste et l'algorithme dit d'Euclide, fondamental et généralisable à d'autres domaines mathématiques.

3.1 La division euclidienne.

Soit $b \in \mathbb{N}^*$ et considérons sur la droite tous les multiples de b.



Soit $a \in \mathbb{Z}$; il est clair qu'il a sa place sur la droite et que nécessairement il se trouve dans un intervalle de longueur b. Il existe donc $q \in \mathbb{Z}$ tel que

$$qb \leq a < (q+1)b$$

ou encore

$$a = qb + r \text{ avec } 0 \leq r \leq (b-1),$$

ce qui peut aussi être écrit :

$$a = qb + r \text{ avec } 0 \leq r < b.$$

Définition 1 *Nous venons de définir la* **division euclidienne** *de $a \in \mathbb{Z}$ par $b \in \mathbb{N}^*$. L'entier q est appelé* **le quotient** *et l'entier r* **le reste.**

Exemples Division de 36 par 5,

$$36 = 5 * 7 + 1.$$

Division de -36 par 5,

$$-36 = 5 * (-8) + 4.$$

Division de 19 par 5,

$$19 = 5 * 3 + 4.$$

Remarquer que ce n'est pas la division de 19 par 3 puisque $4 > 3$. On a :

$$19 = 3 * 6 + 1.$$

Remarque Il est possible de définir la division par un élément de $\mathbb{Z}^*$. Nous ne le faisons car il est possible de s'en passer.

3.2 L'algorithme d'Euclide

Commençons par écrire une propriété simple mais très importante.

Propriété Soient deux entiers naturels non nuls a et b et la division euclidienne de a par b : $a = bq + r$ avec $0 \leq r < b$. Alors l'ensemble des diviseurs communs à a et b coïncide avec l'ensemble des diviseurs communs à b et r.

Démonstration Si d divise a et b, il divise a et bq donc leur différence r. Inversement, si d divise b et r, il divise bq et r donc leur somme a. □

Exemple Soient 381 et 51 et cherchons leurs diviseurs communs. La division euclidienne

$$381 = 51 * 7 + 24$$

montre, d'après la propriété, que ce sont les diviseurs communs à 51 et à 24. Mais alors on peut réappliquer de nouveau la propriété à 51 et 24. On a

$$51 = 24 * 2 + 3.$$

Les diviseurs communs cherchés sont les diviseurs communs à 24 et de 3. Une nouvelle division euclidienne donne

$$24 = 3 * 8 + 0.$$

Il reste à déterminer les diviseurs communs à 3 et à 0; il y en a deux : 1 et 3.
Le pgcd de 381 et de 51 est donc 3 qui est le dernier reste non nul dans la suite des divisions euclidiennes écrites.
Formalisons ceci dans le cadre général pour deux entiers naturels a et b non nuls. Supposons $a > b$, le cas $a = b$ étant trivial.
Posons $a_0 := a$, $a_1 := b$, $q_1 := q$, $a_2 := r$, de sorte que la division euclidienne

$$a = bq + r \text{ avec } r < b$$

s'écrive :

$$a_0 = a_1q_1 + a_2 \text{ avec } a_2 < a_1$$

On écrit alors successivement

$$a_1 = a_2q_2 + a_3 \text{ avec } a_3 < a_2$$

$$a_2 = a_3q_3 + a_4 \text{ avec } a_4 < a_3$$

$$\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots$$

$$a_i = a_{i+1}q_{i+1} + a_{i+2} \text{ avec } a_{i+2} < a_{i+1}$$

$$\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots$$

$$a_{m-2} = a_{m-1}q_{m-1} + a_m \text{ avec } a_m < a_{m-1}$$

$$a_{m-1} = a_mq_m + (a_{m+1} = 0) \text{ avec } 0 < a_m.$$

Nous venons d'écrire, en détail, **l'algorithme d'Euclide**, que nous allons maintenant étudier.
La suite des entiers naturels a_i est strictement décroissante, donc il existe un indice (ici $(m+1)$) tel que $a_{m+1} = 0$. D'après la propriété précédente, appliquée m fois, les diviseurs communs à a et b sont les diviseurs communs à a_{m-1} et à a_m, donc ce sont les diviseurs de a_m puisque a_{m-1} est multiple de a_m, par suite, on peut énoncer le théorème fondamental suivant :

Théorème 1 *Le dernier reste non nul dans l'algorithme d'Euclide relatif à deux entiers naturels a et b est le pgcd de a et b. De plus, tous les diviseurs communs à a et b sont les diviseurs de leur pgcd.*

Algorithme 1 (Algorithme d'Euclide)
Entrée : *deux entiers a et b.*
Sortie : *d, leur pgcd.*
début
$A := a$; $B := b$;
Tant que $B \neq 0$
faire
$R :=$ *Reste* (A, B);

$A := B; B := R;$
refaire
d:= A;
fin

Commentaire
Reste (A, B) est connu de celui qui doit exécuter l'algorithme ; c'est une fonction qui donne le reste de la division euclidienne de A par B.

3.3 Retour sur le pgcd et le ppcm

Outre le théorème précédent, l'algorithme ci-dessus permet d'obtenir facilement les propriétés suivantes :
Propriétés

1. Pour trois entiers naturels a, b, c, on a
$$\text{pgcd}(ac, bc) = \text{pgcd}(a, b)c.$$
2. Si d divise a et b, alors
$$\text{pgcd}(\frac{a}{d}, \frac{b}{d}) = \frac{\text{pgcd}(a, b)}{d}.$$
3. $d = \text{pgcd}(a, b)$ si et seulement si $a = da', b = db'$ et $\text{pgcd}(a', b') = 1$.

Démonstration Pour les deux premières propriétés, on reprend les formules de l'algorithme d'Euclide en les multipliant par c pour la première propriété et en divisant par d pour la seconde propriété.
Prouvons la troisième : on applique la propriété 2 avec $d = \text{pgcd}\ (a, b)$ pour obtenir pgcd $(a', b') = 1$. Dans l'autre sens, on applique la propriété 1 avec $c = d$. □

Nous allons maintenant introduire le théorème de Bézout $(1730 - 1783)$ dont la paternité devrait revenir, en fait, à Bachet de Méziriac $(1581 - 1638)$.
Pour $(a, b) \in \mathbb{N}^2$, non nuls tous les deux, considérons l'ensemble suivant :

$$\mathcal{E} = \{ax + by, x \in \mathbb{Z}, y \in \mathbb{Z}\}.$$

Etudions ses propriétés.

- La somme de deux éléments de $\mathcal{E}$ est dans $\mathcal{E}$, le produit d'un élément de $\mathcal{E}$ par un élément de $\mathbb{Z}$ est dans $\mathcal{E}$, tous les multiples d'un élément de $\mathcal{E}$ sont dans $\mathcal{E}$.
- $\mathcal{E} \cap \mathbb{N}^* \neq \emptyset$, puisque a et b en sont des éléments.
- Soit d le plus petit élément de $\mathcal{E} \cap \mathbb{N}^*$. Tous les multiples de d sont dans $\mathcal{E}$. Inversement, soit z un élément quelconque de $\mathcal{E}$ et considérons la division euclidienne de z par d.
$$z = dq + r \quad \text{avec } 0 \leq r \leq d - 1.$$

L'entier r est dans $\mathcal{E}$ car il s'exprime comme la différence de deux éléments de $\mathcal{E}$; comme $r < d$, et comme d est le plus petit élément positif de $\mathcal{E}$, il vient $r = 0$, d'après la définition de d. On a donc montré que $\mathcal{E}$ est l'ensemble des multiples de d. En particulier d divise a et b.

- Soit δ un diviseur commun à a et à b; δ divise tous les éléments de $\mathcal{E}$, donc en particulier δ divise d, d est donc le plus grand diviseur, soit $d = \text{pgcd}\ (a, b)$.

Nous retrouvons ainsi le théorème 1 et en exprimant que d est élément de $\mathcal{E}$, il vient :

Théorème 2 (Bézout) *Pour deux entiers naturels a et b, non nuls tous les deux, il existe (x, y) dans $\mathbb{Z}^2$ tels que $pgcd(a, b) = ax + by$.*

Exemple Nous avons vu plus haut que $pgcd(381, 51) = 3$. On peut alors vérifier que $51 * 15 - 381 * 2 = 3$. Comment avons nous trouvé les entiers 15 et -2 ? On peut pour les petits entiers le faire par tâtonnement, en se servant de la remarque du paragraphe suivant mais surtout on déterminera, au paragraphe 5, un algorithme donnant les couples vérifiant le théorème de Bézout et ceci sera très important pour les méthodes de cryptographie. D'autres couples satisfont la relation de Bézout, pour les entiers $a = 381$ et $b = 51$ par exemple $(142, -19)$, $(-112, 15)$. Voir le dernier paragraphe pour une explication détaillée.

Théorème 3 *Deux entiers a et b sont premiers entre eux si et seulement s'il existe (x, y) dans $\mathbb{Z}^2$ tels que*

$$xa + yb = 1$$

Démonstration Le théorème de Bézout nous dit que si pgcd $(a, b) = 1$ alors il existe (x, y) dans $\mathbb{Z}^2$ tels que $xa + yb = 1$. Inversement si cette relation est vérifiée et si d divise a et b alors il divise $ax + by$ donc 1 et par suite $d = 1$. □

3.4 Le théorème de Gauss

Ce théorème est déjà cité dans les travaux de Jean Prestet (1650? – 1690), mais on ne prête qu'aux riches (!). Nous retrouverons le " Prince des Arithméticiens", l'allemand C.F. Gauss (1777 – 1855) dans le chapitre 6 pour une étude plus importante. Néanmoins, le théorème suivant sera utilisé constamment dans la suite soit sous la forme présentée, soit sous d'autres formes équivalentes.

3.4.1 Le théorème

Théorème 4 (Gauss) *Soit $(a, b, c) \in \mathbb{N}^3$. On suppose que a divise bc et que a et b sont premiers entre eux, alors a divise c.*

Démonstration Les entiers a et b étant premiers entre eux, ne peuvent être nuls tous les deux. D'après le théorème de Bézout, il existe $(x, y) \in \mathbb{Z}^2$ tels que

$$ax + by = 1$$

Multiplions la relation par c pour obtenir

$$axc + byc = c$$

L'hypothèse nous dit que a divise bc donc aussi byc; comme a divise axc, a divise $byc+axc$ soit c. □

Remarque On peut écrire le théorème dans un cas particulier très intéressant : si p premier divise un produit ab, il divise nécessairement a ou b. En effet si p ne divise pas a, il est premier avec a et donc il divise b
Exemples 3 divise $6 * 8$, premier avec 8 il divise 6.
3 divise $6 * 9$ et il divise à la fois les deux facteurs.
3 ne divise ni 8, ni 22, il ne peut donc diviser leur produit $8 * 22$.

Ecrivons ce théorème sous une autre forme qui se rencontre très souvent

Corollaire 1 *Soient quatre entiers a, b, c, d tels que $ab = cd$ et b et d premiers entre eux, alors b divise c.*

Démonstration b divise ab donc cd qui lui est égal. On applique alors le théorème de Gauss : b et d sont premiers entre eux, alors b divise c. □

3.4.2 Ses conséquences

Le théorème de Gauss a de nombreuses applications. Donnons-en trois ici sous forme de corollaires.

Corollaire 2 *Soit $(a, b, N) \in \mathbb{N}^3$; on suppose que a divise N, que b divise N et que a et b sont premiers entre eux, alors ab divise N.*

Démonstration Comme a divise N, on peut écrire $N = an$. Comme b divise N, b divise an, et d'après le théorème de Gauss, b divise n, ce que l'on peut écrire, $n = bm$ et par suite $N = (ab)m$. □

Remarque L'hypothèse pgcd (a, b) est nécessaire : en effet, 3 divise 12 et 6 divise 12 mais $3 * 6 = 18$ ne divise pas 12.

Corollaire 3 *Soient deux entiers a et b non nuls, premiers entre eux et soit d divisant ab alors on peut écrire $d = d_1 d_2$ avec d_1 divisant a et d_2 divisant b, et cela de façon unique. Evidemment, d_1 et d_2 sont premiers entre eux.*

Démonstration Soit $\delta := pgcd(d, b)$ et écrivons $d = \delta d'$ et $b = \delta b'$. d divise ab donc $\delta d'$ divise $\delta b' a$. Comme $\delta \neq 0$, on peut dire que d' divise $b'a$; comme d' et b' sont premiers entre eux, grâce au théorème de Gauss, d' divise a. Ainsi $d = \delta d'$ avec d'divisant a et δ divisant b.
Supposons deux écritures de d, soit $d = d_1 d_2 = d'_1 d'_2$, avec d_1 etd'_1 divisant a et d_2 et d'_2 divisant b. d_1 et d'_2 sont premiers entre eux comme diviseurs d'entiers premiers entre

eux. D'après le corollaire 1 d_1 divise d'_1. Par symétrie d'_1 divise d_1 et par suite $d_1 = d'_1$. De façon analogue $d_2 = d'_2$. □

Exemple Pour trouver les diviseurs de $a = 175$, on peut remarquer que $a = bc$ avec $b = 7$ et $c = 25$. Les diviseurs de b sont 1 et 7, ceux de 25 sont $1, 5, 25$. Par suite les diviseurs de 175 sont $1, 5, 25, 7, 35, 175$.

Corollaire 4 *On suppose b et c, non nuls, premiers entre eux. On peut alors écrire pour tout entier a :*

$$pgcd(a, bc) = pgcd(a, b) * pgcd(a, c).$$

Démonstration Posons $\delta := pgcd(a, bc)$, $\delta_1 := pgcd(a, b)$, $\delta_2 := pgcd(a, c)$. Comme δ_1 divise a et b, il divise a et bc donc δ_1 divise δ. Pour la même raison δ_2 divise δ. Comme δ_1 et δ_2 sont premiers entre eux puisque il en est ainsi de b et c, $\delta_1\delta_2$ divise δ,(cor.2). Inversement, comme δ divise bc, d'après le corollaire précédent, on peut écrire $\delta = d_1d_2$ avec d_1 divisant b et d_2 divisant c. Comme en plus δ divise a, alors d_1 et d_2 divisent a et par suite d_1 divise δ_1 et d_2 divise δ_2. Finalement δ divise $\delta_1\delta_2$, d'où l'égalité demandée. □

Pour terminer ce paragraphe, nous donnons le théorème suivant :

Théorème 5 *Pour deux entiers naturels a et b, non nuls tous les deux, on a :*

$$pgcd(a, b)ppcm(a, b) = ab$$

et tout multiple commun à a et b est multiple de leur ppcm.

Démonstration Si a ou b est nul, le résultat est immédiat. Dans le cas contraire, soit $d := pgcd(a, b)$ et m un multiple commun à a et b; on peut écrire $m = xa = yb$ et $a = da'$, $b = db'$ avec $pgcd(a', b') = 1$, Par suite, on a $m = xda' = ydb'$, soit $xa' = yb'$ et d'après (cor.1), $x = b'x'$. Finalement $m = xa = b'x'a = x'\frac{ab}{d}$. On en déduit donc que tout multiple de a et de b est multiple de $\frac{ab}{d}$, par suite $\frac{ab}{d}$ est bien le ppcm. □

3.5 Preuve constructive du théorème de Bézout

Réécrivons l'algorithme d'Euclide :

$$a_0 = a_1q_1 + a_2 \text{ avec } a_2 < a_1$$

$$a_1 = a_2q_2 + a_3 \text{ avec } a_3 < a_2$$

$$a_2 = a_3q_3 + a_4 \text{ avec } a_4 < a_3$$

$$\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots$$

$$a_i = a_{i+1}q_{i+1} + a_{i+2} \text{ avec } a_{i+2} < a_{i+1}$$

$$\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots$$

$$a_{m-2} = a_{m-1}q_{m-1} + a_m \text{ avec } a_m < a_{m-1}$$

$$a_{m-1} = a_m q_m + (a_{m+1} = 0) \text{ avec } 0 < a_m.$$

Nous pouvons écrire pour tout $n \in \{1, 2, \ldots, m\}$,

$$a_{n+1} = a_{n-1} - a_n q_n.$$

Considérons pour $n \in \{1, 2, \ldots, m\}$, deux suites définies par les mêmes relations de récurrence mais avec des conditions initiales différentes :

$$s_{n+1} = s_{n-1} - s_n q_n \text{ avec } s_0 = 0 \text{ et } s_1 = 1$$

$$t_{n+1} = t_{n-1} - t_n q_n \text{ avec } t_0 = 1 \text{ et } t_1 = 0$$

Avant de donner la nouvelle preuve du théorème de Bézout, regardons deux exemples qu'il suffira de généraliser.

- Considérons l'algorithme d'Euclide appliqué à 128 et 37

$$\begin{aligned} 128 &= 37 * 3 + 17 \\ 37 &= 17 * 2 + 3 \\ 17 &= 3 * 5 + 2 \\ 3 &= 2 * 1 + 1 \\ 2 &= 1 * 2 + 0 \end{aligned}$$

q_i			3	2	5	1	2
a_i	128	37	17	3	2	1	0
s_i	0	1	−3	7	−38	45	−128
t_i	1	0	1	−2	11	−13	37

On remarque que

$$45 * 37 - 128 * 13 = 1,$$

avec $m = 5$ et pgcd $(a, b) = 1$.

- Considérons l'algorithme d'Euclide appliqué à 261 et 42

$$\begin{aligned} 261 &= 42 * 6 + 9 \\ 42 &= 9 * 4 + 6 \\ 9 &= 6 * 1 + 3 \\ 6 &= 3 * 2 + 0 \end{aligned}$$

q_i			6	4	1	2
a_i	261	42	9	6	3	0
s_i	0	1	−6	25	−31	87
t_i	1	0	1	−4	5	−14

On remarque que

$$87 * 5 - 31 * 14 = 1,$$

avec $m = 4$ et pgcd $(a, b) = 3$.

Dans les deux cas on s'aperçoit que, pour $n \in \{0, 1, \ldots, m\}$, on a la relation

$$a_n = at_n + bs_n,$$

relation que nous allons maintenant démontrer.

Lemme 1 *Avec les notations de l'algorithme d'Euclide, on a*

$$a_n = at_n + bs_n,$$

pour $n \in \{0, 1, \ldots, m\}$.

Démonstration La propriété est vraie pour $n = 0$ et $n = 1$. Calculons $at_{n+1} + bs_{n+1}$ en utilisant la formule de récurrence.

$$\begin{aligned} at_{n+1} + bs_{n+1} &= a(t_{n-1} - t_n q_n) + b(s_{n-1} - s_n q_n) \\ &= (at_{n-1} + bs_{n-1}) - q_n(at_n + bs_n) \\ &= a_{n-1} - q_n a_n \\ &= a_{n+1} \end{aligned}$$

□

Cette proposition appliquée pour la valeur $n = m$ donne

$$pgcd(a, b) = a_m = at_m + bs_m$$

On obtient donc, de façon effective, les coefficients de la relation de Bézout.

3.6 Exemples

Donnons trois exemples que nous utiliserons plus loin.

1. Considérons les entiers 970 et 9.

$$\begin{aligned} 970 &= 9 * 107 + 7 \\ 9 &= 7 * 1 + 2 \\ 7 &= 2 * 3 + 1 \\ 2 &= 1 * 2 + 0 \end{aligned}$$

q_i			107	1	3	2
a_i	970	9	7	2	1	0
s_i	0	1	−107	108	−431	970
t_i	1	0	1	−1	4	−9

Nous avons donc

$$4 * 970 - 9 * 431 = 1$$

2. Pour les entiers 1440 et 7 nous pouvons écrire :

$$\begin{aligned} 1\,440 &= 7*205 + 5 \\ 7 &= 5*1 + 2 \\ 5 &= 2*2 + 1 \\ 2 &= 1*2 + 0 \end{aligned}$$

q_i			205	1	2	2
a_i	1 440	7	5	2	1	0
s_i	0	1	-205	206	-617	1440
t_i	1	0	1	-1	3	-7

Nous avons la relation :

$$1\,440 * 3 - 7 * 617 = 1$$

3. Avec un plus grand entier $a = 1\,234\,567\,890$ et avec $b = 167$

$$\begin{aligned} 1\,234\,567\,890 &= 167 * 7\,392\,622 + 16 \\ 167 &= 16 * 10 + 7 \\ 16 &= 7 * 2 + 2 \\ 7 &= 2 * 3 + 1 \\ 2 &= 1 * 2 + 0 \end{aligned}$$

q_i			7 392 622	10	2	3	2
a_i	a	b	16	7	2	1	0
s_i	0	1	$-a$	x	y	z	$-a$
t_i	1	0	1	-10	21	-73	b

avec $x = 73\,926\,221$, $y = -155\,245\,064$, $z = 539\,661\,413$.
Nous avons la relation :

$$167 * 539\,661\,413 - 1\,234\,567\,890 * 73 = 1$$

3.7 Algorithme

Terminons ce chapitre en écrivant l'algorithme utilisé implicitement dans les exemples précédents.

Algorithme 2 (Algorithme d'Euclide-Bézout)
Entrée : *deux entiers naturels a et b.*
Sortie : *d, leur pgcd, et (x, y) tel que $ax + by = d$.*
début
$s_0 := 0; s_1 := 1; t_0 := 1; t_1 := 0;\ A := a; B := b;$
Si $b = 0$
alors

$d := a; x := 1; y := 0;$
sinon
Tant que $B > 0$
faire
$Q :=$ *Quotient* $(A, B);$
$R := A - Q * B;$
$S := s_0 - Q * s_1;$
$T := t_0 - Q * t_1;$
$A := B; B := R;$
$s_0 := s_1; s_1 := S;$
$t_0 := t_1; t_1 := T;$
refaire
Finsi
$d := A; x := t_0; y := s_0;$
fin

Commentaires

• Expliquons l'instruction suivante :

Si $\mathcal{P}$
alors
corps d'instructions 1
sinon
corps d'instructions 2
Finsi

Si la proposition $\mathcal{P}$ est vraie, alors on exécute le corps d'instructions 1, puis on passe après l'instruction "Finsi". Si elle est fausse, on réalise le corps d'instructions 2, puis on passe après l'instruction "Finsi".

- Quotient (a, b) donne le quotient entier de a par b , c'est-à-dire le quotient de la division euclidienne de a par b.
- Appliquer l'algorithme sur l'un des exemples précédents.

3.8 Compléments

3.8.1 Étude des couples (x, y) qui vérifient la relation de Bézout.

Soient deux couples (x, y) et (x', y') tels que

$$d = ax + by = ax' + by',$$

où d est le pgcd de a et b.
Alors $a(x - x') = b(y' - y)$. Ecrivons $a = da'$ et $b = db'$, par suite

$$(*) \qquad a'(x - x') = b'(y' - y).$$

On applique le théorème de Gauss, sous la forme du corollaire 1 : puisque a' et b' sont premiers entre eux, a' divise $(y - y')$. On peut donc écrire

$$y = y' + ca' \text{ avec } c \in \mathbb{Z},$$

et d'après (*)

$$a'(x - x') = -ca'b'$$

$$x - x' = -cb' \text{ puisque } a' \neq 0$$

$$x = x' - cb'.$$

Par suite l'équation $d = ax + by$ a une infinité de solutions et si l'on connait l'une d'entre elles (x', y') , toutes les autres s'écrivent : (x, y) avec $x = x' - cb', y = y' + ca'$ pour $c \in \mathbb{Z}$. Ainsi si x est le premier élément d'un couple solution, alors $x + b'$ et $x - b'$ sont aussi les premiers éléments respectivement d'un couple solution, par suite il existe une solution x dans l'ensemble $\{1, 2, \ldots, b' - 1\}$. On a alors nécessairement y négatif et la relation $b'y = 1 - a'x$ montre que $-b'y < a'b'$ et donc $-y < a'$.
En conclusion, il existe une solution (x, y) de l'équation $d = ax + by$ telle que $|x| < b/d$ et $|y| < a/d$. Si pour cette solution, x est positif, alors y est négatif et une autre solution vérifie les mêmes hypothèses, à savoir le couple (x', y') avec $x' = x - b/d < 0$ et $y' = y + a/d > 0$.
Exemples
• $17 * 1 + 8 * (-2) = 1$ et $17 * (-7) + 8 * (15) = 1$. Les deux couples $(1, -2)$ et $(-7, 15)$ correspondent à la dernière remarque et tous les couples solutions sont de la forme :

$$x = 1 + 8c \text{ et } y = -2 - 17c.$$

• On a pgcd(25, 35) = 5 et les deux égalités

$$25 * 3 - 35 * 2 = 5,$$

$$25 * (-4) + 35 * 3 = 5.$$

Remarque Il est possible de montrer que l'algorithme d'Euclide donne le meilleur couple cherché, à savoir le couple (x, y) tel que $|x|$ et $|y|$ soient les plus petits possibles.

3.8.2 Résolution de $ax + by = c$

a, b, c étant trois entiers naturels donns, on demande de trouver l'ensemble des couples $(x, y) \in \mathbb{Z}^2$, vérifiant $ax + by = c$.
Soit d = pgcd (a, b) alors d doit diviser c. Par suite, si d ne divise pas c, l'équation est impossible. On suppose désormais que d divise c et l'on écrit :

$$a = a'd,\ b = b'd,\ c = c'd.$$

La relation à étudier devient

$$a'x + b'y = c' \text{ avec pgcd}(a', b') = 1.$$

D'après l'identité de Bézout, il existe u et v tels que

$$a'u + b'v = 1$$

et après multiplication par c', on a

$$a'uc' + b'vc' = c'.$$

Cette relation et l'antépénultième donnent par différence :

$$a'(x - uc') = b'(vc' - y').$$

D'après le théorème de Gauss, il existe λ tel que

$$x - uv' = \lambda b',$$

$$c'v - y = \lambda a'.$$

L'ensemble des solutions est donc de la forme

$$x = uc' + \lambda b',$$

$$y = vc' - \lambda a',$$

λ décrivant $\mathbb{Z}$.
Par exemple résolvons $3x+5y = 7$. L'équation a des solutions puisque 3 et 5 sont premiers entre eux. Comme $3*2-5 = 1$, on peut choisir $u = 2$ et $v = -1$. L'ensemble des solutions est donc donné par

$$x = 14 + 5\lambda,$$

$$y = -7 + 3\lambda.$$

Ainsi $(14,-7),(4,-1),(-1,2)$ sont solutions.

Chapitre 4

Les nombres premiers

Où l'on découvre les nombres premiers qui sont à la base de toute l'arithmétique.

4.1 Premières propriétés

Rappelons que :

Définition 1 *Un* **nombre premier** *est un entier ayant exactement deux diviseurs. Un* **nombre composé** *est un entier, non nul, ayant au moins trois diviseurs.*

• Les entiers 0 et 1 ne sont ni premiers, ni composés.

Théorème 1 *Tout entier $n \geq 2$ possède un diviseur premier.*

Démonstration Comme n a au moins deux diviseurs, il suffit de considérer le plus petit diviseur supérieur strictement à 1. □

Corollaire 1 *Soit n un nombre composé. Alors n a deux diviseurs premiers distincts ou n est divisible par le carré d'un nombre premier.*

Démonstration D'après le théorème précédent, on peut écrire $n = pm$ avec p premier et $m \geq 2$. On réapplique alors le théorème. □

• $n = 360$ est divisible par 2 et 3 et $n = 361$ est le carré de 19.

Un premier algorithme Pour déterminer si un entier N est premier, une méthode consiste à constater qu'il n'est divisible par aucun premier inférieur à $\sqrt{N}$. En effet, si N a un diviseur premier propre p, c'est à dire différent de 1 et de N, alors il en a un autre (éventuellement égal à p) et forcément l'un des deux est moindre que $\sqrt{N}$ sinon leur produit dépasserait N. Cette méthode nécessite la connaissance de tous les nombres premiers jusqu'à $\sqrt{N}$.

Exemple Pour vérifier la primalité ou non des entiers 1499 et 1501, il faut regarder s'ils sont divisibles par les nombres premiers inférieurs à 37. On voit alors que 1501 est

divisible par 19 et que 1499 est premier. On pourrait se poser la question : sachant que 1499 est premier, que peut-on dire sur 1501, ou inversement sachant que 1501 est premier, que peut-on savoir sur 1499 ? Aucun résultat général n'existe sur ce genre de question.

Algorithme 1 (Algorithme de Primalité)
Entrée : *un entier positif n.*
Sortie : *n est premier ou n posséde comme plus petit facteur premier p*
début
I := 1;
Tant que $P(I) \leq \sqrt{n}$
faire
  Si *Reste* $(n, P(I)) = 0$
    alors
      écrire n est composé et son plus petit facteur premier est $P(I)$*;fin;*
    sinon $I := I + 1$;
  Finsi
refaire
Imprimer n est premier;
fin

Commentaire • P est un tableau de nombres premiers ; P(i) représente le ième nombre premier. On suppose ici qu'il est connu et que le tableau est assez grand.

Théorème 2 *Il y a une infinité de nombres premiers.*

Démonstration Supposons qu'il existe seulement un nombre fini r de nombres premiers, que nous appelons $p_1, p_2, p_3, \ldots\ldots p_{r-1}, p_r$ et soit N leur produit. Soit q un diviseur premier de $(N-1)$, c'est forcément un des p_i donc il divise aussi N. L'entier q divise donc la différence entre N et $N-1$, c'est à dire 1 ce qui n'est pas possible. □

4.2 Le crible d'Eratosthène

Pour déterminer tous les nombres premiers inférieurs ou égaux à N, nombre entier donné, le mathématicien grec Eratosthène (284 - 192 av. J.-C.), procéda comme suit.
On écrit tous les nombres entiers supérieurs ou égaux à 2 jusqu'à N. Nous l'expliquons parallèlement avec $N = 40$.

2	3	4	5	6	7	8	9	10	11	12	13	14	15
16	17	18	19	20	21	22	23	24	25	26	27	28	29
30	31	32	33	34	35	36	37	38	39	40			

L'entier 2 est premier ; tous ses multiples ne peuvent être premiers; on les barre.

2	3	4	5	6	7	8	9	10	11	12	13	14	15
16	17	18	19	20	21	22	23	24	25	26	27	28	29
30	31	32	33	34	35	36	37	38	39	40			

Le premier nombre non barré est 3 ; il est premier. On barre tous ses multiples. Certains ont d'ailleurs déjà été barré à l'étape précédente.

2	3	~~4~~	5	~~6~~	7	~~8~~	~~9~~	~~10~~	11	~~12~~	13	~~14~~	~~15~~
~~16~~	17	~~18~~	19	~~20~~	~~21~~	~~22~~	23	~~24~~	25	~~26~~	~~27~~	~~28~~	29
~~30~~	31	~~32~~	~~33~~	~~34~~	35	~~36~~	37	~~38~~	~~39~~	~~40~~			

Le premier nombre non barré est 5 ; il est premier et l'on barre tous ses multiples.

2	3	~~4~~	5	~~6~~	7	~~8~~	~~9~~	~~10~~	11	~~12~~	13	~~14~~	~~15~~
~~16~~	17	~~18~~	19	~~20~~	~~21~~	~~22~~	23	~~24~~	~~25~~	~~26~~	~~27~~	~~28~~	29
~~30~~	31	~~32~~	~~33~~	~~34~~	~~35~~	~~36~~	37	~~38~~	~~39~~	~~40~~			

et l'on continue ainsi jusqu'à ce que le nouveau nombre premier obtenu ait son carré supérieur à N. Dans notre exemple, c'est terminé et les nombres non barrés fournissent la liste des nombres premiers jusqu'a 40, soit

$$2,\ 3,\ 5,\ 7,\ 11,\ 13,\ 17,\ 19,\ 23,\ 29,\ 31,\ 37.$$

Pourquoi peut-on s'arrêter là ? Les multiples des nombres premiers p suivants ont déjà été barrés car ils sont de la forme ap avec $a < p$ et ont donc un autre diviseur moindre que p.

Cette méthode, dite méthode de crible, est un exemple fondamental de méthodes plus générales.

Algorithme 2 (Algorithme d'Eratosthène)
Entrée : *un entier positif N.*
Sortie : *la liste des nombres premiers inférieurs à N*
début
{*Construction du tableau des entiers $\leq N$.*}
$I := 2$;
Tant que $I \leq N$
faire
  $T(I) := 1$;
  $I := I + 1$;
refaire
{ *Crible*}
$J := 2$;
$Q := \sqrt{N}$;
Tant que $J < Q$
faire
  $K := J + J$;
  Tant que $K \leq N$
  faire
    $T(K) = 0$;
    $K := K + J$;

```
    refaire
    J := J + 1;
    Tant que T(J) = 0
    faire
        J := J + 1;
    refaire
refaire
I := 2;
Tant que I < N + 1
faire
    si T(I) ≠ 0 imprimer I; finsi
    I:= I + 1;
refaire
fin
```

Commentaires

- Le tableau T contient d'abord 1 pour chaque valeur de $I \geq 2$: ainsi T(4) = 1. Ensuite, pour les nombres barrés, on affecte 0 ; ainsi T(4) = 0 en fin d'algorithme.

- Une ligne entre accolades est un commentaire. La ligne n'est pas prise en compte par l'algorithme; elle sert à la compréhension du lecteur.

- Pourquoi avons nous posé $Q := \sqrt{N}$ et utilisé "Tant que $J < Q$" alors que nous aurions pu écrire plus simplement "Tant que $J < \sqrt{N}$" ?
  Dans notre formulation $\sqrt{N}$ est calculé une seule fois, alors que dans la deuxième, l'expression est calculée à chaque début de boucle.

- L'instruction $K := J + J$ peut être remplacée par l'instruction $K := J * J$. Cette instruction évite de barrer des nombres déjà barrés. Ainsi lorsque l'on veut barrer les multiples de 7, il est inutile de barrer $2 * 7, 3 * 7, 4 * 7, 5 * 7, 6 * 7$ qui ont déjà subi cette mutililation; il suffit de commencer à barrer $7 * 7$.

4.3 Factorisation

Théorème 3 *Tout entier $n \geq 2$ s'écrit :*

$$n = \prod_{1}^{k} p_i^{a_i} = p_1^{a_1} p_2^{a_2} \ldots\ldots p_{k-1}^{a_{k-1}} p_k^{a_k},$$

où les p_i, pour $i \in \{1, 2, \ldots, k\}$, sont des nombres premiers. Cette écriture est unique à l'ordre des facteurs près.

Démonstration

- Montrons d'abord l'existence d'une telle décomposition. Si n est premier, c'est fini, sinon soit p son plus petit diviseur; c'est un nombre premier. On écrit $n = pm$ avec $m < n$. On recommence le raisonnement avec m. On construit ainsi une suite strictement décroisssante de nombres et nécessairement, on arrive sur un nombre premier, ce qui termine la démonstration.

- Montrons maintenant l'unicité. Considérons deux décompositions supposées distinctes que nous écrivons

$$n = AN = AM.$$

A regroupe tous les facteurs premiers identiques dans les deux décompositions. Soit p un facteur premier intervenant dans la décomposition de N. Il n'intervient donc pas dans celle de M. On a cependant p divise

$$M = p_1^{b_1} p_2^{b_2} \ldots\ldots p_{k-1}^{b_{k-1}} p_k^{b_k}.$$

On peut encore dire que p divise

$$M = p_1^{b_1} (p_2^{b_2} \ldots\ldots p_{k-1}^{b_{k-1}} p_k^{b_k}).$$

p ne divise pas p_1, sinon $p = p_1$, donc d'après le théorème de Gauss, p divise $(p_2^{b_2} \ldots\ldots p_{k-1}^{b_{k-1}} p_k^{b_k})$, ce qui n'est pas possible pour les mêmes raisons. On arrive donc à une contradiction. □

Remarque On comprendra mieux cette démonstration lorsque l'on verra plus tard des ensembles où cette propriété n'est pas vraie.

Pratique de la factorisation

Pour factoriser 332 640 on peut adopter la disposition suivante :

332 640	2
166 320	2
83 160	2
41 580	2
20 790	2
10 395	3
3 465	3
1 155	3
385	5
77	7
11	

Par suite $332\,640 = 2^5 * 3^3 * 5 * 7 * 11$.

Algorithme 3 (Factorisation)

Entrée : *un entier $n \geq 2$, P un tableau ordonné des premiers nombres premiers.*

Sortie : *la factorisation de n.*

début

$I := 1$; $N := n$; $A(I) = 0$;

Tant que $N \neq 1$
faire
  $p := P(I)$;
  Tant que Reste $(N, p) = 0$
  faire
    $A(I) := A(I) + 1$;
    $N := N/p$;
  refaire
  $I := I + 1$;
  $A(I) := 0$;
refaire
{ *Impression* }
$J := 1$;
Tant que $J < I$
faire
  si $A(J) \neq 0$ *alors imprimer* $(P(I); A(I))$; *finsi*
  $J := J + 1$;
refaire
fin

Commentaires

- P est le tableau des nombres premiers. Il doit être connu jusqu'au plus grand nombre premier inférieur à $\sqrt{n}$
- A (I) est le ième élément d'un tableau. Il représente l'exposant de P(I) dans la décomposition de n.
- Reste (a, b) représente le reste dans la division euclidienne de a par b.

Remarque L'algorithme présenté ici est simple mais pas très efficace. Il ne permet pas de factoriser des nombres de plus de 10 chiffres en un temps raisonnable. D'autres algorithmes de factorisation, faisant appel à des mathématiques plus sophistiquées ont été découverts, mais de toute façon le problème de la factorisation est difficile et de nos jours factoriser un nombre de 100 chiffres est encore presque impossible. Evidemment la question de temps est primordiale.
On verra, au chapitre 6, un algorithme de chiffrement dont la sécurité est basée sur la difficulté de factoriser.

4.4 Encore le pgcd et le ppcm

Propriétés

- Lorsque l'on connaît la factorisation de deux nombres a et b, il est facile de trouver leur pgcd et leur ppcm. Soit en effet

$$n = \prod_1^k p_i^{a_i} \text{ et } m = \prod_1^k p_i^{b_i}$$

alors

$$pgcd(n,m) = \prod_1^k p_i^{min(a_i,b_i)} \text{ et } ppcm(n,m) = \prod_1^k p_i^{max(a_i,b_i)}$$

Démonstration En effet si p_i est un facteur premier commun à n et à m, il interviendra dans la factorisation de leur pgcd avec l'exposant le plus petit entre a_i et b_i. Pour le ppcm, il faut prendre, au contraire, le plus grand. □

Exemple Calculons pgcd(332 640, 286). Plus avant, nous avons vu que $332\,640 = 2^5 * 3^3 * 5 * 7 * 11$. Comme $286 = 2 * 11 * 13$, le pgcd cherché est $2 * 11$, soit 22.

- On peut aussi calculer facilement le nombre $d(n)$ des diviseurs d'un entier $n \geq 2$. On a

$$n = \prod_1^k p_i^{a_i} \text{ et } d(n) = \prod_1^k (a_i + 1).$$

Démonstration En effet les diviseurs de n sont de la forme

$$d = \prod_1^k p_i^{c_i} \text{ avec pour tout } i \in \{1, \ldots, k\}, \ c_i \leq a_i.$$

Il y a donc pour chaque valeur de i, $a_i + 1$ valeurs possibles pour c_i, d'où le résultat. □

Exemple Les diviseurs de 332 640 sont de la forme $2^a * 3^b * 5^c * 7^d * 11^e$ avec $a \in \{0,1,2,3,4,5\}$, $b \in \{0,1,2,3\}$, $c, d, e \in \{0,1\}$. Il y a $6 * 4 * 2 * 2 * 2$ soit 192 diviseurs.

Chapitre 5

Numération

Où l'on apprend à écrire les nombres, où l'on raconte comment les anciens les écrivaient et où l'on explique la notion de base.

Les êtres humains ont d'abord compté avec les doigts avant d'être capable d'écrire les nombres. La représentation des nombres, telle qu'elle existe maintenant dans le monde entier, nous semble tellement naturelle, nous qui l'utilisons depuis notre plus tendre enfance, qu'il nous est difficile d'imaginer qu'elle n'a atteint l'Occident qu'au treizième siècle. Il est même difficile de penser à d'autres représentations.

5.1 La représentation que l'on connaît

Il est à peu près sûr maintenant que les chiffres $0, 1, 2, 3, 4, 5, 6, 7, 8, 9$ et le système de numération, c'est à dire la façon d'écrire les nombres sont d'origine indienne. Tous les chiffres sauf le zéro datent du deuxième siècle avant J.-C.. Le zéro apparaît pour la première fois dans un document de 458. Après avoir transité par le Moyen - Orient, d'où le nom "chiffres arabes", les chiffres et la méthode de numération sont arrivés en Occident au dixième siècle, et c'est le mathématicien italien Fibonacci ($1170 - 1250$) qui propagea chiffres et méthode en Occident. La graphie des chiffres, telle que nous la connaissons, date du treizième siècle.

Un nombre tel que 202 utilise :

- un des 10 symboles appelés chiffres et ces 10 symboles suffisent à écrire tous les nombres,
- le chiffre 2 n'a pas la même valeur suivant sa position dans le nombre. Le premier 2 représente $200 = 2 * 10^2$, le dernier ne représente que 2. Notre système est un système de numération de position en base 10,
- le zéro - prodigieuse invention - marque, ici, qu'il n'y a pas d'unité du deuxième ordre.

5.2 D'autres systèmes de position

La notion de chiffre a arrêté d'autres civilisations qui auraient pu prétendre à la découverte du système de numération de position.

1. Le système babylonien, 1800 avant J.-C., avec l'apparition d'un zéro 4 siècles avant J.-C., était un système de position, de base 60, dans sa conception la plus avancée. Mais il n'y avait pas 60 symboles pour représenter les unités mais uniquement deux indiquant le 1 et le 10. Le 1 était représenté par le symbole et le 10 par  .

   Un nombre compris entre 1 et 59 était représenté par une méthode d'addition. Ainsi 35 était écrit 

   et $204 = 3 * 60 + 24$ était représenté par 

2. Le système savant chinois du deuxième siècle avant J.-C., était un système de position, en base 10. Il ne possédait que deux chiffres le 1 écrit I et le 5 écrit –. Les nombres de 1 à 9 étaient écrits par le procédé d'addition: ainsi 4 s'écrivait IIII. Il est à remarquer que les savants chinois utilisaient un autre système de numération avec 9 chiffres, chiffres actuellement encore en vigueur, mais ce système n'était pas positionnel.

3. Le système des savants mayas conçu entre les cinquième et neuvième siècle utilisait la base 20, et n'avait pas non plus 20 symboles pour écrire les unités de base mais uniquement deux : un point pour le 1 et un tiret pour le 5.

5.3 D'autres systèmes de numération

Pour bien comprendre l'intérêt de notre système de numération, donnons deux exemples de systèmes qui ont été abondamment utilisés y compris par de grands mathématiciens.

5.3.1 Le système grec du troisième siècle avant J.-C.

C'est ce système qu'ont utilisé Euclide (315? – 235? av. J.-C.) et Archimède (287 – 212 av. J.-C.), alors que Pythagore, cinquième siècle avant J.-C., n'avait à sa disposition qu'un système encore plus primitif. Ce système était alphabétique, c'est à dire qu'il utilisait les lettres de l'alphabet. Comme l'alphabet français nous est plus familier, nous le présentons "à la française".
27 symboles étaient nécessaires pour écrire tous les nombres jusqu'à 999. Les chiffres de 1 à 9 étaient représentés par les lettres de A à I, les dizaines $10, 20, \ldots, 90$ par les lettres de J à R, les centaines par les lettres de S à Z et disons ç pour 900.
Ainsi notre 794 s'écrivait-il YRD et 704 se notait-il YD.
'A, 'B, ..., 'I représentaient 1 000, 2 000, ..., 9 000.
Les mathématiciens grecs ne connaissaient pas le zéro, aussi étaient-ils obligés de définir tous ces symboles.

La myriade, ensuite, représentait 10 000. Nous la noterons $\mathcal{M}$. La myriade surmontée d'un nombre n inférieur à 9 999 représentait n myriades, ainsi

$$\overset{JB}{\mathcal{M}}$$

représente 12 000 et

$$\overset{TJB}{\mathcal{M}}\,'BKA$$

est la représentation de 212 myriades auxquelles on ajoute 2 024, soit du nombre 2 122 024. On peut ainsi représenter tous les nombres jusqu'à 99 999 999 $= 10^8 - 1$.
Archimède alla plus loin dans la représentation en utilisant la base 10^8 (voir section suivante).

Vérifier l'opération

$$SKB + TLI = UOA.$$

5.3.2 Le système romain

C'est un système qui perdure encore dans certaines notations mais il est vraiment archaïque et n'est pas du tout opératoire, c'est à dire ne permet pas de réaliser des opérations. Les symboles I, V, X, L, D, M représentent respectivement 1, 5, 10, 50, 100, 500, 1000. C'est un système additif et soustractif : tout signe placé à gauche d'un signe de valeur supérieure s'en retranche: ainsi 4 s'écrit IV. Donnons d'autres exemples :

$MMCCCXLIV$ se lit 2344

et $MCDXXXVII$, 1437.

Essayez d'additionner ces deux nombres écrits en chiffres romains !

5.4 La notion de base

Ecrivons un nombre sous la forme

$$N = a_n a_{n-1} \dots a_1 a_0,$$

ce qui signifie:

$$N = a_n * 10^n + a_{n-1} * 10^{n-1} + \dots + a_1 * 10 + a_0$$

et que les symboles a_i, pour $i \in \{0, 1, .., n\}$ sont à prendre dans l'ensemble $\{0, 1, 2, \dots, 9\}$. D'autre part $a_n \neq 0$. Ainsi 3423 signifie:

$$3423 = 3 * 10^3 + 4 * 10^2 + 2 * 10 + 3 = 3000 + 400 + 20 + 3$$

C'est Blaise Pascal (1623 – 1662) qui en 1652 définit correctement la notion de système de numération dans une base B quelconque. L'écriture

$$N = (a_n a_{n-1} \dots a_1 a_0)_B,$$

signifie que

1.
$$N = a_n * B^n + a_{n-1} * B^{n-1} + ... + a_1 * B + a_0,$$

2. les a_i pour $i \in \{0, 1, .., n\}$ appartiennent à un ensemble de B symboles contenant un zéro,

3.
$$a_n \neq 0.$$

Donnons quatre exemples:

- $B = 2$. C'est la numération binaire utilisée par les ordinateurs, la première fois introduite par l'allemand Leibnitz (1646−1716). Deux symboles suffisent: on choisit le 0 et le 1. Notre 2 s'écrit $(10)_2$, notre 3, $(11)_2$. Il va de soi que, si l'on travaille souvent en base 2, on laisse tomber l'indice 2 et la suite des nombres écrits en base 2 est donc
$$0, 1, 10, 11, 100, 101, 110, 111, 1000, ...$$
Comme on le voit, moins de symboles utilisés entraîne une écriture plus longue des nombres.
Pourquoi ne pas tenir vos comptes personnels dans le système binaire afin de les rendre hermétiques aux regards indiscrets, comme le faisait l'allemand Georg Brander (1713−1783), constructeur d'instruments de mathématique, physique et astronomie ?

- $B = 16$. Compte tenu de la remarque précédente les informaticiens utilisent la base 16 pour écrire l'adresse et le contenu des mémoires. Il nous faut dans ce système 16 symboles et l'on a choisi les 10 chiffres classiques puis les lettres A, B, C, D, E, F pour représenter $11, 12, 13, 14, 15$.
Ainsi $(A9B)_{16}$ représente le nombre que nous avons l'habitude d'écrire 2 715 puisque
$$10 * 16^2 + 9 * 16 + 11 = 2\,715.$$

- $B = 10^8$. Cette base a été utilisée par le mathématicien et physicien grec Archimède qui a vécu à Syracuse (en Sicile actuelle) (287 − 212 av. J.-C.). Il est célèbre entre autre
  - pour avoir estimé le nombre π,
  - pour avoir énoncé le principe d'hydrostatique qui porte son nom,
  - pour avoir aidé à la défense de sa ville par les romains.

  Archimède a suivi les cours d'Euclide à Alexandrie et à cette occasion rencontra Eratosthène.
  On l'a vu les savants grecs savaient représenter les nombres jusqu'à $10^8 - 1$. Ils pouvaient écrire ainsi toutes les octades. Archimède proposa alors de mettre côte à côte deux octades et de continuer ainsi. Ce n'était pas autre chose qu'une représentation en base 10^8.

- $B = 60$. Ce système se retrouve encore partiellement dans la vie courante,
  - pour calculer les heures en minutes et secondes,
  - dans le calcul de la mesure des angles en minutes et secondes.

5.5 Changement de base

5.5.1 À la main

Lorsque l'on utilise plusieurs bases, on doit savoir écrire un même nombre en plusieurs bases.

- Le nombre $(123)_{16}$ s'écrit en base 10, $16^2 + 2 * 16 + 3$ soit 291.
- Soit n le nombre 123 écrit en base 10. Quel est son écriture en base 16 ? Comme $123 = 7 * 16 + 11$, l'entier n s'écrit $(7B)_{16}$.
- Soit a $= (62)_7$. L'écrire en base 11. Nous allons utiliser la base 10 comme base de repli car c'est dans cette base que l'on sait bien calculer. Ce nombre s'écrit $6*7+2$ en base 10; c'est donc 44 et comme $44 = 4 * 11$, a s'écrit 40 en base 11.
- De façon générale, écrire, en base 10, un nombre connu en base B est facile car il suffit de réaliser un calcul dans notre base favorite. Si
$$N = (a_n a_{n-1} \dots a_1 a_0)_B.$$
il suffit de calculer
$$N = a_n * B^n + a_{n-1} * B^{n-1} + \dots . a_1 * B + a_0$$
- En revanche écrire un nombre a, donné en base 10, dans une autre base B est plus délicat. Si
$$a = (a_n a_{n-1} \dots a_1 a_0)_B$$
est l'écriture recherchée, on remarque que
$$a = (a_n * B^{n-1} + a_{n-1} * B^{n-2} + \dots . + a_1) * B + a_0.$$
a_0 est donc le reste de la division euclidienne de a par B,
$$a = qB + a_0.$$
Pour obtenir a_1, on remplace alors a par q et l'on continue ainsi.
Prenons un exemple: soit 43 981 à écrire en base 16. On a :
$$43\,981 = 2\,748 * 16 + 13,$$
$$2\,748 = 171 * 16 + 12,$$
$$171 = 10 * 16 + 11,$$
par suite, $a_0 = 13$, $a_1 = 12$, $a_2 = 11$, $a_3 = 10$, et
$$(43\,981)_{10} = (A\,BCD)_{16}.$$

5.5.2 Les algorithmes

Ecrivons les algorithmes réalisant les changements de base.

Algorithme 1 (Changement de base : $Base \mapsto 10$)
Entrée : la suite A(i) des chiffres d'un entier positif a écrit en base Base.
Sortie : N, le même entier écrit en base 10.
{ *on rentre le nombre chiffre par chiffre, les chiffres étant écrit en base* 10 *et l'on termine par Base* }
début
$i := 0;$
Tant que $X < Base$
faire
$A(i) := X;$
$i := i + 1;$
refaire
$K := i - 1; N := A(0); k := 1; B := Base;$
Tant que $k \leq K$
faire
$N := N + A(k) * B;$
$B := B * Base;$
$k := k + 1;$
refaire
$N;$
fin

Exemple Soit A 375 un nombre écrit en base 16. On veut connaître sa valeur en base 10. La suite de ces chiffres est $16, 11, 3, 7, 5$. Par suite A(0) = 5, A(1) = 7, A(2) = 3, A(3) = 11.
On a successivement , en déroulant l'algorithme :
$K = 3, N = 5, k = 1, B = 16,$
$N = 5 + 7 * 16 = 117, B = 16 * 16 = 256, k = 2,$
$N = 117 + 3 * 256 = 885, B = 256 * 16 = 4\,096, k = 3,$
$N = 885 + 11 * 4\,096 = 25\,641, B = 4\,096 * 16, k = 4.$
Comme $k > K$, on sort de la boucle avec $N = 45\,941$.

Algorithme 2 (Changement de base : $10 \mapsto Base$, chiffres de poids faibles d'abord)

Entrée : un nombre a écrit dans la base 10, *la nouvelle base : Base.*
Sortie: la suite A(i) des chiffres du nombre nombre a écrit en base Base.
début
$i := 0; N := a;$
Tant que $N > 0$
faire
$A(i) :=$ *Reste (N , Base);*
$N :=$ *Quotient (N, Base);*

$i := i + 1;$
refaire
$K := i - 1;$
fin

Exemple Soit $a = 45\ 941$.
Au départ $i = 0$ et $N = 45\ 941$, ensuite,
$A(0) = 5, N = 2\ 871, i = 1$, puisque $45\ 941 = 2\ 871 * 16 + 5$,
$A(1) = 7, N = 179, i = 2$ puisque $2\ 871 = 179 * 16 + 7$,
$A(2) = 3, N = 11, i = 3$ puisque $179 = 11 * 16 + 3$,
$A(3) = 11, N = 0, i = 4$ puisque $11 = 0 * 16 + 11$.
On sort avec $K = 3$.

Algorithme 3 (Changement de base bis : 10 $\mapsto$ *Base*, chiffres de poids forts d'abord)

Entrée : un nombre a écrit dans la base 10 et la nouvelle base : Base.
Sortie: la suite A(i) des chiffres du nombre a écrit en base Base.
début
N := a; X := Base; k := 0;
Tant que X < N
faire
k := k + 1;
*X := X*Base;*
refaire
X := Quotient (X, Base); i := k;
Tant que i > 1
faire
A(i) = Quotient(N, X);
N := Reste (N, X);
X := Quotient(X, Base);
i := i - 1;
refaire
A(0) := N;
fin

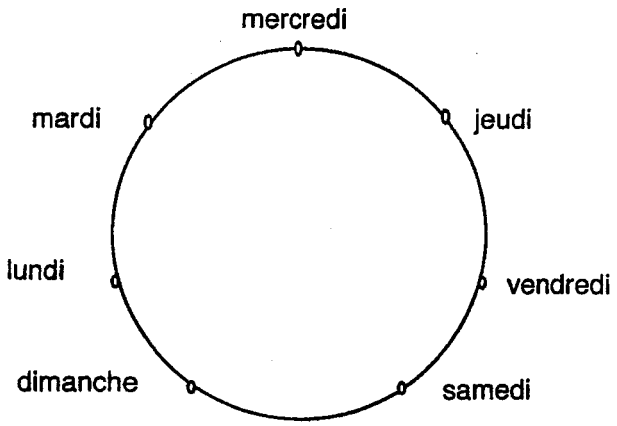
Exemple Soit $a = 45\ 941$
On a d'abord $N = 45\ 941, X = 16, k = 0$, puis
$k = 1$ et $X = 256$, $k = 2$ et $X = 4\ 096$, $k = 3$ et $X = 65\ 536$.
Comme $X > N$, il vient $X = 4\ 096$ et $i = 3$.
Déroulons la deuxième boucle "tant que" :
$A(3) = 11, N = 885, X = 256, i = 2$, puisque $45\ 941 = 11 * 4\ 096 + 885$,
$A(2) = 3, N = 117, X = 16, i = 1$, puisque $885 = 3 * 256 + 117$,
$A(1) = 7, N = 5, X = 1, i = 0$, puisque $117 = 7 * 16 + 5$.
On sort de la boucle avec $A(0) = 5$.
Remarque Comparez les algorithmes 1 et 3.

Chapitre 6

Calcul modulaire

Où l'on définit, avec Gauss, les classes modulo n, où l'on montre comment on peut travailler avec et comment on va pouvoir les utiliser pour nos applications

6.1 Introduction



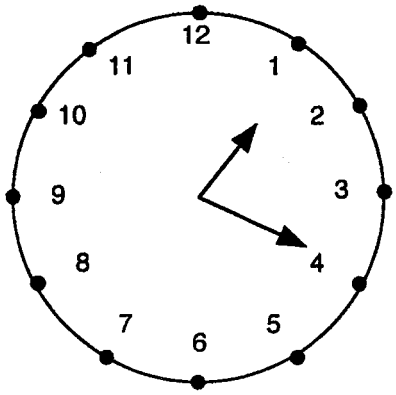
Considérons l'année 1997. Le premier de l'an est un mercredi. Nous mettons dans la même classe, appelée mercredi, tous les jours de l'année tombant un mercredi : on y trouve, entre autre, le 8 janvier, le 21 mai et le dernier jour de l'année. On crée ainsi les classes lundi, mardi, mercredi, jeudi, vendredi, mais vous connaissez... et reconnaissez sur la figure ci-dessus.
Il y a sept classes comme les sept jours de la semaine. On désignera par $\mathcal{S}$ - comme

semaine - l'ensemble de ces classes.

Nous allons procéder de même avec les nombres en suivant les définitions proposées par le mathématicien allemand Carl Friedrich Gauss (1777 - 1855). En dehors de l'arithmétique, son thème favori en mathématique, il s'est aussi occupé d'astronomie et de physique, plus particulièrement de magnétisme, d'électricité et d'optique.

6.2 La notion de modulo

Pour la compréhension, on peut s'aider d'une pendule et chacun sait bien qu'une heure après midi, c'est treize heures ou encore une heure et que 24 heures après, on est à la même heure du jour suivant. On a l'habitude de travailler modulo 12 ou modulo 24, selon les cas.



Définition 1 *Soient $n \in \mathbb{N}$ et $(a,b) \in \mathbb{Z}^2$. On dit que a et b sont* **congrus modulo** n *si et seulement si n divise $(a-b)$ On note*

$$\mathbf{a \equiv b \pmod{n}}$$

n est appelé **module de la congruence.**

Exemple Modulo 26, 5 et 31 sont congrus et congrus aussi à -47. Modulo 2, tous les entiers pairs sont congrus entre eux et tous les entiers impairs de même ; ainsi, on voit l'apparition de deux classes. Plus généralement, pour n fixé, on met dans la même classe tous les éléments congrus entre eux.

Prenons $n = 5$, on a les classes

$$\ldots\ldots, -20, -15, -10, -5, 0, 5, 10, 15, \ldots\ldots$$

$$\ldots\ldots,-21,-16,-11,-6,-1,4,9,14,\ldots\ldots$$
$$\ldots\ldots,-22,-17,-12,-7,-2,3,8,13,\ldots\ldots$$
$$\ldots\ldots,-23,-18,-13,-8,-3,2,7,12,\ldots\ldots$$
$$\ldots\ldots,-24,-19,-14,-9,-4,1,6,11,\ldots\ldots$$

Il y a 5 classes et il est habituel de les noter le plus simplement possible $\overline{0},\overline{1},\overline{2},\overline{3},\overline{4}$. Commençons par donner quelques propriétés simples de la **congruence modulo l'entier** n.

Théorème 1 *On suppose* $a \equiv b \pmod{n}$ *et soit* d *un diviseur de* n, *alors* $a \equiv b \pmod{d}$

Démonstration $(a-b)$ est multiple de n donc aussi des diviseurs de n. □

Montrer que si on a $a \equiv 2 \pmod{17}$, alors on a $a \not\equiv 1 \pmod{119}$.

Théorème 2 *On suppose* $a \equiv b \pmod{n}$, $a \equiv b \pmod{m}$ *et* m *et* n *premiers entre eux, alors* $a \equiv b \pmod{nm}$

Démonstration n et m divisent $(a-b)$. D'après (cor.2, 3.4.2), on en déduit que nm divise $(a-b)$. □

Théorème 3 *Si* $a \equiv b \pmod{n}$ *et* $c \equiv d \pmod{n}$ *alors*

$$a+c \equiv b+d \pmod{n}$$

et

$$ac \equiv bd \pmod{n}.$$

Démonstration L'entier n divise $(a-b)$ et $(c-d)$ donc leur somme $(a+c)-(b+d)$ donc

$$a+c \equiv b+d \pmod{n}.$$

De même n divise $(a-b)$ donc aussi $(ac-bc)$; n divise $(c-d)$ donc aussi $(cb-bd)$. Divisant $(ac-bc)$ et $(cb-bd)$, n divise leur somme $(ac-bd)$, donc

$$ac \equiv bd \pmod{n}.$$

□

Exemples Calculer $5^7 \bmod 21$. On peut calculer $5^7 = 78125$ puis déterminer le reste de la division par 21 et trouver 5. Il est préférable, de travailler avec des nombres plus petits, en utilisant le théorème :

$$5^7 = 5^2 * 5^2 * 5^2 * 5 \equiv 4 * 4 * 4 * 5 \equiv 16 * 20 \equiv 16 * (-1) = -16 \equiv 5 \pmod{21}$$

Corollaire 1 *Si* $a \equiv b \pmod{n}$, *alors pour tout* $k \in \mathbb{N}$, $a^k \equiv b^k \pmod{n}$

Exemple $2^{44497} - 1 \equiv (-1)^{44497} - 1 \equiv -2 \equiv 1 \pmod{3}$

6.3 Les classes

On notera l'ensemble des classes modulo l'entier n par $\mathcal{E}_n$. Dans chaque classe de $\mathcal{E}_n$, il y a un et un seul élément de l'ensemble $\{0, 1, \ldots, n-1\}$, d'où la définition :

Définition 2 *On note $a \bmod n$, le seul élément b vérifiant*

$$b \equiv a \pmod{n}$$

$$0 \leq b \leq n-1$$

Ainsi

$$36 \bmod 7 \text{ , c'est } 1,$$

$$54 \bmod 17 \text{ , c'est } 3,$$

$$-36 \bmod 8 \text{ , c'est } 4.$$

Les classes de $\mathcal{E}_n$ peuvent donc être notées

$$\overline{0}, \overline{1}, \ldots, \overline{(n-2)}, \overline{(n-1)}.$$

Dans la classe $\overline{a}$, il y a tous les éléments de la forme $a + kn$ pour toutes les valeurs de $k \in \mathbb{Z}$.
Le théorème 3 ci-dessus nous permet de définir des opérations sur les classes en posant :

$$\overline{a} + \overline{b} := \overline{a+b}$$

$$\overline{a} * \overline{b} := \overline{a*b}$$

On peut ainsi dresser la table d'addition et de multiplication dans $\mathcal{E}_5$

$+$	$\overline{0}$	$\overline{1}$	$\overline{2}$	$\overline{3}$	$\overline{4}$
$\overline{0}$	$\overline{0}$	$\overline{1}$	$\overline{2}$	$\overline{3}$	$\overline{4}$
$\overline{1}$	$\overline{1}$	$\overline{2}$	$\overline{3}$	$\overline{4}$	$\overline{0}$
$\overline{2}$	$\overline{2}$	$\overline{3}$	$\overline{4}$	$\overline{0}$	$\overline{1}$
$\overline{3}$	$\overline{3}$	$\overline{4}$	$\overline{0}$	$\overline{1}$	$\overline{2}$
$\overline{4}$	$\overline{4}$	$\overline{0}$	$\overline{1}$	$\overline{2}$	$\overline{3}$

$*$	$\overline{0}$	$\overline{1}$	$\overline{2}$	$\overline{3}$	$\overline{4}$
$\overline{0}$	$\overline{0}$	$\overline{0}$	$\overline{0}$	$\overline{0}$	$\overline{0}$
$\overline{1}$	$\overline{0}$	$\overline{1}$	$\overline{2}$	$\overline{3}$	$\overline{4}$
$\overline{2}$	$\overline{0}$	$\overline{2}$	$\overline{4}$	$\overline{1}$	$\overline{3}$
$\overline{3}$	$\overline{0}$	$\overline{3}$	$\overline{1}$	$\overline{4}$	$\overline{2}$
$\overline{4}$	$\overline{0}$	$\overline{4}$	$\overline{3}$	$\overline{2}$	$\overline{1}$

Exemple Faites donc le même travail pour $\mathcal{E}_6$ et $\mathcal{E}_7$. L'addition et la multiplication des classes ont toutes les propriétés de l'addition et de la multiplication des entiers et des propriétés supplémentaires qui tiennent compte que $\mathcal{E}_n$ est un ensemble fini.

Liste de propriétés faciles à vérifier et que l'on peut faire à titre d'exercice. Pour tout (a, b, c) dans $\mathbb{Z}^3$, on a :

$$\overline{a} + \overline{b} = \overline{b} + \overline{a}.$$

$$(\overline{a} + \overline{b})\overline{c} = \overline{a}\overline{b} + \overline{b}\overline{c}.$$

$$\overline{a} + \overline{0} = \overline{a}.$$

$$\overline{a} + \overline{(-a)} = \overline{0}.$$

$$\overline{a}\overline{b} = \overline{b}\overline{a}$$

$$\overline{a}(\overline{b}\overline{c}) = (\overline{a}\overline{b})\overline{c}$$

$$\overline{a}\overline{1} = \overline{a}.$$

Le problème que l'on va étudier au paragraphe suivant est de savoir si, une classe étant donnée, elle possède une classe inverse, c'est à dire $\overline{a}$ étant donné, existe-t-il $\overline{b}$ tel que

$$\overline{a}\overline{b} = \overline{1} \ ?$$

On peut encore dire a étant donné existe-t-il b tel que

$$ab \equiv 1 \pmod{n}?$$

Si tel est le cas, on dira aussi que b **est l'inverse de** a **modulo** n

Remarque La dernière notation est plus précise en ce sens que l'entier n est indiqué. Dans l'égalité $\overline{a}\overline{b} = \overline{1}$, il faut savoir que l'on travaille dans $\mathcal{E}_n$

6.4 Retour au théorème de Gauss

Le théorème de Gauss, que l'on rappelle :

- Si a divise bc et si $pgcd(a, b) = 1$, alors a divise c,

peut être écrit aussi, en utilisant la notion de modulo, sous l'une des formes suivantes :

- Si $bc = 0 \pmod{a}$ et si $pgcd(a, b) = 1$, alors $c \equiv 0 \pmod{a}$.
- Si $bc = bd \pmod{a}$ et si $pgcd(a, b) = 1$, alors $c \equiv d \pmod{a}$.
- Dans $\mathcal{E}_n$, si $\overline{a}\overline{b} = \overline{0}$ et si $\overline{a}$ est inversible, alors $\overline{b} = \overline{0}$

6.5 Les correspondances entre classes

Nous avons vu dans le premier chapitre que la cryptographie élémentaire nécessitait la connaissance de $\mathcal{E}_{26}$. En particulier on a besoin d'étudier les transformations possibles de $\mathcal{E}_{26}$ dans lui même. Nous avions défini ainsi "PLUS2", "MULT3".

6.5.1 Première correspondance

Considérons l'application

$$\mathcal{E}_n \mapsto \mathcal{E}_n$$

$$\overline{x} \mapsto \overline{x+a}.$$

Cette application est **bijective** ce qui signifie que (i) tout élément de $\mathcal{E}_n$ est atteint (ii) il n'est atteint qu'une seule fois. Comme $\mathcal{E}_n$ est fini le (ii) est automatiquement vérifié dès que (i) l'est ou inversement. Ici tout $\overline{y}$ est image de $\overline{y-a}$.
Exemple Pour la cryptographie nous avons vu au premier chapitre l'application :

$$\mathcal{E}_{26} \mapsto \mathcal{E}_{26}$$

$$\overline{x} \mapsto \overline{x+2},$$

cette dernière ligne pouvant être écrite, si on n'a pas peur des confusions,

$$x \mapsto x+2.$$

L'application réciproque qui permet de déchiffrer est

$$x \mapsto x-2.$$

6.5.2 Deuxième correspondance

Considérons l'application

$$\mathcal{E}_n \mapsto \mathcal{E}_n$$

$$\overline{x} \mapsto \overline{ax}.$$

Cette application n'est pas toujours bijective comme le montre un des deux exemples suivants :
Exemples L'application

$$\mathcal{E}_6 \mapsto \mathcal{E}_6$$

$$x \mapsto 2x,$$

transforme respectivement $0,1,2,3,4,5$ en $0,2,4,0,2,4$.
Quant à l'application

$$\mathcal{E}_5 \mapsto \mathcal{E}_5$$

$$x \mapsto 2x,$$

elle transforme respectivement $0,1,2,3,4$ en $0,2,4,1,3$.
On voit une différence importante entre les deux exemples : dans le premier certains éléments ne sont pas atteints et en conséquence certains ont plusieurs antécédents. Dans le deuxième exemple, chaque élément a un et un seul antécédent.
Nous avons le théorème :

Théorème 4 *L'application*

$$\mathcal{E}_n \mapsto \mathcal{E}_n$$
$$\overline{x} \mapsto \overline{ax},$$

est bijective si et seulement si a et n sont premiers entre eux. En particulier, ceci est vérifié si n est premier et si $\overline{a} \neq \overline{0}$.

Démonstration On se doit d'abord de remarquer que la propriété a et n premiers entre eux vaut aussi pour tout b congru à a modulo n, c'est à dire ;

$$\overline{a} = \overline{b} \text{ alors } pgcd(a,n) = pgcd(b,n).$$

Soit d :=pgcd(a,n).

- Si $d \neq 1$, on peut écrire $a = a'd$ et $n = n'd$. L'image de $\overline{n'}$ est alors

$$\overline{an'} = \overline{a'dn'} = \overline{a'n} = \overline{0}.$$

  0 et n' ont donc la même image et l'application ne peut être bijective.

- Si $d = 1$, on applique le théoréme de Bézout ; il existe deux entiers c et e tels que :

$$ac + ne = 1,$$

  soit

$$ac \equiv 1 \pmod{n},$$

  soit

$$\overline{ac} = \overline{1}.$$

  Ceci montre que l'application est bijective puisque si on prend $\overline{y}$ dans $\mathcal{E}_n$ alors c'est l'image de $\overline{cy}$ puisque $\overline{a}(\overline{cy}) = \overline{y}$. □

Le théorème peut encore s'énoncer comme suit :

Corollaire 2 *Si a et n sont premiers entre eux, alors a est inversible modulo n. Ceci signifie qu' il existe b tel que $ab \equiv 1 \pmod{n}$, ou encore $\overline{a}\overline{b} = \overline{1}$ dans $\mathcal{E}_n$.*

Comme on le montre au pararagraphe suivant cet inverse peut être calculé à partir de l'algorithme d'Euclide-Bézout.

Corollaire 3 *Si $a \equiv b \pmod{n}$, alors pour tout $k \in \mathbb{Z}$, $a^k \equiv b^k \pmod{n}$*

Démonstration
- Si $k \in \mathbb{N}$, c'est le corollaire 1.
- si $-k \in \mathbb{N}$, a^{-k} et b^{-k} sont égaux donc il en est de même de leurs inverses. □

6.5.3 Algorithme de calcul de l'inverse modulaire

Comme l' a montré la démonstration du théorème 4, lorsque a et n sont premiers entre eux le théoréme de Bézout permet d'obtenir deux entiers c et e tels que :

$$ac + ne = 1,$$

ce qui donne

$$ac \equiv 1 \pmod{n}.$$

Les deux entiers c et e sont obtenus par l'algorithme d'Euclide-Bézout.

Algorithme 1 (Inverse modulaire)
Entrée : *deux entiers positifs a et n.*
Sortie : *x l'inverse de a modulo n ou a n'est pas inversible modulo n.*
début
Faire appel à l'algorithme "Euclide-Bézout" avec comme entrée (a, n) *pour obtenir* (d, x, y);
Si $d > 1$
alors imprimer a et n ne sont pas premiers entre eux;
sinon
Si $x < 0$
alors $x := x + n$;
Finsi
Finsi
fin

Exemples

- Dans $\mathcal{E}_{26}$, la correspondance $\overline{x} \mapsto \overline{ax}$, est bijective si et seulement si $a \not\equiv 2$ et 13 (mod 26). Pour déchiffrer l'algorithme de chiffrement "MULTa", il faut trouver l'inverse b de a modulo 26 et l'algorithme de déchiffrement est "MULTb". Donnons le tableau des inversibles de $\mathcal{E}_{26}$ avec leurs inverses.

1	3	5	7	9	11	15	17	19	21	23	25
1	9	21	15	3	19	7	23	11	5	17	25

  On peut vérifier l'exactitude du tableau, mais il vaut mieux utiliser l'algorithme d'Euclide-Bézout et construire le tableau.

- Dans $\mathcal{E}_{23}$, toutes les aplications $\overline{x} \mapsto \overline{ax}$, sont bijectives sauf l'application nulle, c'est-à-dire sauf l'application $\overline{x} \mapsto \overline{0}$.

- Déterminons 5^{-15} mod 26. D'après le tableau ci-dessus l'inverse de 5 est 21, par suite

$$5^{-15} \equiv 21^{15} \equiv (-5)^{15} \equiv ((-5)^2)^7 * (-5) \equiv (-1)^7 * (-5) \equiv 5 \pmod{26}.$$

6.5.4 Retour à l'exemple de l'introduction

Considérons l'application de $\mathcal{S}$, l'ensemble des jours de la semaine, défini au paragraphe 1, dans $\mathcal{E}_7$ par :

mercredi	1
jeudi	2
vendredi	3
samedi	4
dimanche	5
lundi	6
mardi	0

Cette correspondance est bijective et elle nous permet de travailler sur les jours comme avec des nombres.
Ainsi quel jour de la semaine tombe le 28 avril ? Ce jour est le cent-dix-huitième jour de l'année ($31 + 28 + 31 + 28 = 118$). Or $118 \equiv 6 \pmod 7$ et 6 représente le lundi. On peut donc parler du lundi 28 avril 1997.

6.6 Le petit théorème de Fermat

Fermat (1601 - 1665) était Conseiller au Parlement de Toulouse et passionné de mathématiques. Nous allons parler d'un de ses théorèmes, appelé petit par rapport au grand théorème de Fermat qui vient juste d'être démontré et qui affirme qu'il n'existe pas de quadruplet (a, b, c, n) d'entiers naturels, non nuls, tels que

$$a^n + b^n = c^n,$$

sauf pour $n = 1$ ou 2.

6.6.1 Le théorème

Avant d'énoncer le théorème en vue prenons un exemple.
Calculons les puissances de 3 modulo 7: $3^2 \equiv 2$, $3^3 \equiv 6$, $3^4 \equiv 4$, $3^5 \equiv 5$, $3^6 \equiv 1$. De même $2^2 \equiv 4$, $2^3 \equiv 1$, et donc $2^6 \equiv 1$.

Théorème 5 (Fermat) *Soit p un nombre premier et $a \in \mathbb{Z}$ un entier non multiple de p, alors*

$$a^{p-1} \equiv 1 \pmod p,$$

c'est-à-dire p divise $a^{p-1} - 1$.
On peut écrire dans $\mathcal{E}_p$

$$\overline{a}^{p-1} = \overline{1}.$$

Démonstration

Soit $a \in \mathbb{Z}^*$. Comme l'application $\overline{i} \mapsto \overline{ia}$ pour $i = 1..(p-1)$ est bijective (th. 4), les éléments de $\mathcal{E}_p$ peuvent aussi être décrits par les classes $\overline{ia}$ pour $i = 1..(p-1)$. On a ainsi

$$\overline{a}\ \overline{2a}\ \overline{3a} \ldots\ldots \overline{(p-1)a} = \overline{1}\ \overline{2}\ \overline{3} \ldots\ldots \overline{p-1},$$

donc

$$\overline{(p-1)!a^{p-1}} = \overline{(p-1)!}.$$

Comme $\overline{(p-1)!}$ n'est pas nul, il vient après division

$$\overline{a}^{p-1} = \overline{1}.$$

□

Exemple 7 divise $2^6 - 1 = 63$

6.6.2 Ses corollaires

On déduit aisément du théorème de Fermat les corollaires et théorèmes suivants :

Théorème 6 *Soit a un entier non divisible par le nombre premier p et soient r et s deux entiers tel que $r \equiv s \pmod{(p-1)}$ alors,*

$$a^r \equiv a^s \pmod{p}$$

Démonstration On écrit $r = s + c(p-1)$, pour un entier c, d'où

$$a^r = a^s(a^{(p-1)})^c \equiv a^s \pmod{p},$$

d'après le théorème de Fermat. □

Corollaire 4 *Soit a un entier non divisible par le nombre premier p et soit r un entier alors,*

$$a^r \equiv a^{r \bmod (p-1)} \pmod{p}$$

Exemples

- Calculons 3^{77} mod 7. On a $77 \equiv 5 \pmod{6}$, d'où

  $$3^{77} \equiv 3^5 \equiv 3^2 * 3^2 * 3 \equiv 2 * 2 * 3 \equiv 5 \pmod{7},$$

  donc $3^{77} \bmod 7 = 5$.

- Calculons maintenant l'inverse de 3 modulo 7.
  On peut écrire le théorème de Fermat sous la forme

  $$a^{p-2}a \equiv 1 \pmod{p},$$

  ce qui montre que l'inverse de a est $a^{p-2} \bmod p$.
  Calculons donc $3^5 \equiv 3^2 * 3^2 * 3 \equiv 2 * 2 * 3 \equiv 5 \pmod{7}$.
  L'inverse de 3 est donc 5 ce que l'on vérifie simplement puisque $3 * 5 = 15 \equiv 1 \pmod{7}$.

Corollaire 5 *Pour tout entier a et tout premier p, soit r tel que $r \equiv 1 \pmod{(p-1)}$ alors,*

$$a^r \equiv a \pmod{p}$$

en particulier,

$$a^p \equiv a \pmod{p} \text{ c'est-à-dire } p \text{ divise } a^p - a$$

Démonstration Si p ne divise pas a, c'est un cas particulier du théorème précédent. Si p divise a, les deux membres de la relation sont congrus à zéro. □

Exemples

- 4 ne divise pas $3^4 - 3 = 78$, donc 4 n'est pas premier. Plus généralement si un nombre n ne vérifie pas, pour un certain entier a la relation $a^n \equiv a \pmod{n}$, ceci prouve que n n'est pas premier. Nous avons ici **un test de composition**, qui prouve qu'un nombre n'est pas premier.

- Calculons $(2^{341} - 2) \bmod 341$. On remarque que $341 = 31 * 11$, $2^5 = 32 \equiv 1 \pmod{31}$, que $2^{10} \equiv 1 \pmod{11}$ (th.de Fermat). On a donc $2^{10} \equiv 1 \pmod{11}$ et $2^{10} \equiv 1 \pmod{31}$, soit $2^{10} \equiv 1 \pmod{341}$. Par suite $2^{340} \equiv 1 \pmod{341}$ et $2^{341} \equiv 2 \pmod{341}$.
  Cet exemple montre donc que l'on peut avoir $a^n \equiv a \pmod{n}$ sans que n soit premier. La réciproque du théorème de Fermat n'est donc pas vraie. Son étude a conduit à de nombreux travaux ces vingt dernières années.

Corollaire 6 *Soit a un entier non divisible par les deux nombres premiers distincts p et q. Soient r et s deux entiers tels que $r \equiv s \pmod{(p-1)(q-1)}$ alors,*

$$a^r \equiv a^s \pmod{pq}$$

Démonstration Compte tenu de (cor.2, 3.4.2) et de la symétrie en p et q, il suffit de montrer que $a^r \equiv a^s \pmod{p}$, ce qui est bien vrai, par application du théorème 6, puisque $r \equiv s \pmod{(p-1)}$. □

On en déduit alors le corollaire suivant qu'on érige en théorème puisque c'est le résultat fondamental nécessaire à la construction de l'algorithme de chiffrement RSA.

Théorème 7 *Soient p et q deux nombres premiers distincts et a un entier quelconque. Soit r un entier tel que $r \equiv 1 \pmod{(p-1)(q-1)}$ alors,*

$$a^r \equiv a \pmod{pq},$$

en particulier, pour tout entier c

$$a^{c(p-1)(q-1)+1} \equiv a \pmod{pq}.$$

Démonstration Si p et q ne divisent pas a, c'est un cas particulier du corollaire précédent. Dans le cas contraire, on applique le corollaire 5 et toujours (cor.2, 3.4.2).
Remarque Ce théorème est un cas particulier du théorème d'Euler.

6.7 Caractères de divisibilité

Ecrivons un nombre sous la forme

$$N = a_n a_{n-1} \dots a_1 a_0,$$

ce qui signifie, rappelons-le,

$$N = a_n * 10^n + a_{n-1} * 10^{n-1} + \dots + a_1 * 10 + a_0$$

avec $a_i \in \{0, 1, \dots, 9\}$ pour tout $i \in \{0, 1 \dots, n\}$ et $a_n \neq 0$. Comme on a $10 \equiv 0 \pmod 2$, $10 \equiv 1 \pmod 3$, $10 \equiv 0 \pmod 5$, $10 \equiv 1 \pmod 9$, $100 \equiv 0 \pmod 4$, $100 \equiv 0 \pmod{25}$, il est facile d'obtenir :

$$N \equiv a_0 \pmod 2,$$
$$N \equiv a_0 \pmod 5,$$
$$N \equiv (a_n + a_{n-1} + \dots + a_1 + a_0) \pmod 3,$$
$$N \equiv (a_n + a_{n-1} + \dots + a_1 + a_0) \pmod 9,$$
$$N \equiv (a_1 * 10 + a_0) \pmod 4,$$
$$N \equiv (a_1 * 10 + a_0) \pmod{25}.$$

On peut alors en déduire le théorème de divisibilité suivant :

Théorème 8 *Soit*

$$N = a_n a_{n-1} \dots a_1 a_0.$$

N est divisible par 2 si et seulement si son dernier chiffre est pair, soit 0, 2, 4, 6 ou 8.
N est divisible par 5 si et seulement si son dernier chiffre est 0 *ou* 5.
N est divisible par 3 si et seulement si la somme de ses chiffres est divisible par 3.
N est divisible par 9 si et seulement si la somme de ses chiffres est divisible par 9.
N est divisible par 4 si et seulement si le nombre constitué par ses deux derniers chiffres est divisible par 4.
N est divisible par 25 si et seulement si le nombre constitué par ses deux derniers chiffres est divisible par 25, *c'est-à-dire si N se termine par 00, 25, 50 ou 75.*

Exemple 257 836 945 n'est pas divisible par 9 puisque la somme de ses chiffres est congrue à 4 modulo 9. C'est un nombre divisible par 5 mais pas par 25.

6.8 Preuve d'opération

Remarquons d'abord que pour $(a, b, c, d) \in \mathbb{Z}^4$ et $n \in \mathbb{N}$, on a :

$$\text{si } a + b = c, \text{ alors } a + b \equiv c \pmod n,$$

$$\text{si } a * b = c, \text{ alors } a * b \equiv c \pmod{n},$$

et

$$\text{si } d = a * b + c, \text{ alors } d \equiv a * b + c \pmod{n},$$

Ceci permet de déceler une erreur éventuelle dans une opération, avec des calculs sur des nombres plus petits. Cela ne permet pas de la corriger. Ainsi, si je dis $13 * 7 = 81$ et que je regarde modulo 3 cette relation, j'obtiens $1 * 1 = 0$, qui est manifestement faux. Donc la première opération est fausse ou il y a une erreur dans la vérification. De toute façon, il faut trouver l'erreur, en recommençant les calculs.
Un autre exemple : soit $175 * 38 = 20$, égalité manifestement fausse. Regardons modulo 3. Nous obtenons $1 * 2 = 2$, la preuve est correcte, ce qui ne prouve absolument pas que notre opération est correcte, comme on le voit bien.
La preuve la plus utilisée est **la preuve par 9** puisque d'après la section précédente, il est très facile de calculer a mod 9.
Un autre exemple avec une addition :
soit

$$135\,273 + 287\,327 + 250\,028 = 672\,628.$$

Modulo 9, on a

$$3 + 2 + 8 \equiv 4,$$

ce qui est correct. Modulo 25, on obtient :

$$-2 + 2 + 3 \equiv 3,$$

ce qui est convenable... mais ne prouve pas que l'addition soit correcte. La"preuve" permet seulement d'espérer que l'opération est juste.

Chapitre 7

Retour à la cryptographie

Où l'on revient à la cryptographie avec tout ce que l'on vient d'apprendre. Où l'on explique une autre méthode de chiffrement. Où l'on présente enfin la méthode RSA.

7.1 Retour sur le chapitre un

Les notions de cryptographie étudiées au chapitre 1 se formalisent très bien dans l'ensemble $\mathcal{E}_{26}$. Nous utiliserons, pour simplifier les notations et, pour faire comme tout le monde, les notations $0, 1, 2, \ldots, 25$ pour décrire tous les éléments de $\mathcal{E}_{26}$ et nous écrirons $17 + 14 = 5$ au lieu d'écrire $17 + 14 \equiv 5 \pmod{26}$.

L'algorithme de chiffrement "PLUSa" est alors défini par l'application $x \mapsto x + a$ et l'algorithme "MULTa" par l'application $x \mapsto ax$. On a vu que la dernière application définit un algorithme de chiffrement si elle possède une application réciproque permettant de déchiffrer, soit si et seulement si a est premier avec 26.
L'algorithme "MOINSa" défini par $x \mapsto x - a$ peut aussi être défini par "PLUS (26 - a)". Ce dernier algorithme permet de déchiffrer tout cryptogramme chiffré avec "PLUSa".
L'application réciproque de $x \mapsto ax$ est l'application $x \mapsto bx$, a et b étant inverse l'un de l'autre modulo 26. Nous redonnons le tableau des inversibles du chapitre précédent :

a	1	3	5	7	9	11	15	17	19	21	23	25
b	1	9	21	15	3	19	7	23	11	5	17	25

Nous allons étudier la composition de ces applications dans un cadre plus général d'un alphabet comportant n lettres.

7.2 Généralisation à $\mathcal{E}_n$

Considérons les applications suivantes de $\mathcal{E}_n$ dans $\mathcal{E}_n$:

$$\text{“}PLUSa\text{”} : x \mapsto x + a,$$

$$\text{“}MOINSa\text{”} : x \mapsto x - a,$$

$$\text{“}MULTa\text{”} : x \mapsto ax.$$

Évidemment, on devrait écrire “PLUS (a, n)” pour être précis mais il faut bien souvent simplifier les notations en mathématiques ; nous parlerons donc de “PLUSa” dans $\mathcal{E}_n$. On conserve aussi les mêmes notations que plus haut. Les éléments de $\mathcal{E}_n$ sont écrits à l'aide des entiers $0, 1, 2, \ldots, (n-1)$ et les calculs se font modulo n.

Exemples

- Chiffrons avec “MULTc” puis surchiffrons avec “PLUSa”. On part de x, on obtient d'abord cx puis $cx + a$. Comment déchiffrer ? On peut faire “MOINSa” qui permet de retrouver cx puis appliquer “MULTd”, d étant l'inverse de c modulo n.
  On peut aussi procéder comme suit : soit $y := cx + a$ le chiffré de x alors $cx = y - a$ puis $x = d(y - a)$ après multiplication par d donc $x = dy - da$. En partant de y , on peut donc, pour retrouver, x chiffrer avec “MULTd” puis surchiffrer avec “MOINSda”.

- Chiffrons avec “PLUSa” puis surchiffrons avec “MULTc”. On part de x, on obtient d'abord $x + a$ puis $c(x + a) = cx + ca$. Ceci montre, avec l'exemple précédent, que la permutation des applications n'est pas commutative.

7.2.1 Les applications affines

Toutes les applications précédentes et leurs composées sont des cas particuliers des **applications affines** que l'on peut définir ainsi :

$$\text{“}AFFINE(a,b)\text{”} : x \mapsto ax + b.$$

Exemple Dans $\mathcal{E}_{61}$ considérons le chiffrement $f : x \mapsto 7x + 3$.
Dans ce chiffrement 20 est transformé en 21 puisque $7 * 20 + 3 \equiv 21 \pmod{61}$, 59 est transformé en 50 puisque $7 * 59 + 3 \equiv 7 * (-2) + 3 \equiv -11 \equiv 50 \pmod{61}$.
Pour déchiffrer, on considère $y = 7x + 3$, d'où $7x = y - 3$, soit $x = 35(y - 3)$, puisque 35 est l'inverse de 7 modulo 61(vérifier, retrouver).
Ainsi $x = 35y - 105 = 35y + 17$. Dans $\mathcal{E}_{61}$, la fonction réciproque de “$AFFINE(7,3)$” est “$AFFINE(35,17)$”

De façon générale déterminons la fonction de déchiffrement de “$AFFINE(a,b)$” dans $\mathcal{E}_n$.
Soit $y = ax + b$, alors $x = c(y - b)$. Si a est premier avec n alors soit c l'inverse de a modulo n; on peut écrire $x = cy - cb$.
La fonction de déchiffrement est donc “$AFFINE(c, cb)$”.
Ainsi “$AFFINE(a,b)$” est une fonction de chiffrement si et seulement si a est premier avec n et la fonction de déchiffrement est “$AFFINE(c, cb)$”
Faisons du surchiffrage avec des fonctions affines.
Appliquons d'abord dans $\mathcal{E}_n$, “$AFFINE(a,b)$”, puis “$AFFINE(c,d)$”. Si nous partons

de x, il est d'abord transformé en $ax+b$ puis en $c(ax+b)+d$ soit en $acx+cb+d$. Ainsi :

$$AFFINE(c,d) \circ AFFINE(a,b) = AFFINE(ac, cb+d)$$

7.3 Notion de clé

Jusqu'ici, dans $\mathcal{E}_{26}$ ou plus généralement dans $\mathcal{E}_n$, la même transformation était appliquée à toutes les lettres du message.
Nous allons faire mieux en utilisant des fonctions différentes selon la place de la lettre dans le mot. Comme il est indispensable de se souvenir de toutes les fonctions employées, on utilise la notion de clé qui sera un moyen mnémonique pour retrouver les fonctions.
Prenons un exemple.
Considérons comme clé le mot "nombre", facile à retenir. A ce mot est associé la suite des entiers 13, 14, 12, 01, 17, 04, obtenue en remplaçant chaque lettre par sa position dans l'alphabet. Cette suite va nous servir de paramètres.
Reprenons le message du premier chapitre :

RENDEZ-VOUS VENDREDI SOIR

que nous codons avec les 26 premiers nombres entiers.

170413 030425 211420 182104 130317 040308 181408 17.

Donnons quelques exemples d'utilisation

- Nous considérons les applications "PLUS13", "PLUS14", "PLUS12", "PLUS01", "PLUS17", "PLUS04", que l'on applique respectivement aux lettres du message

R	*E*	*N*	*D*	*E*	*Z*	*V*	*O*	*U*	*S*	*V*	*E*
17	04	13	03	04	25	21	14	20	18	21	04
13	14	12	01	17	04	13	14	12	01	17	04
04	18	25	04	21	03	08	02	06	19	12	08
E	*S*	*Z*	*E*	*V*	*D*	*I*	*C*	*G*	*T*	*M*	*I*

N	*D*	*R*	*E*	*D*	*I*	*S*	*O*	*I*	*R*
13	03	17	04	03	08	18	14	08	17
13	14	12	01	17	04	13	14	12	01
00	17	03	05	20	12	05	02	20	18
A	*R*	*D*	*F*	*U*	*M*	*F*	*C*	*U*	*S*

Le message chiffré est

ESZEV DICGT MIARD FUMFC US

Expliquons : on applique "PLUS13" à la première lettre du message ou du moins à son codé qui est 17, puis on utilise "PLUS14" pour la deuxième lettre, puis on applique "PLUS12" à la troisième lettre. Pour la septième lettre, on reprend "PLUS13" et ainsi de suite.

- On peut aussi considérer les applications "MULT13", "MULT14", "MULT12", "MULT1", "MULT17", "MULT4" et procéder comme dans l'exemple précédent.
- Les deux correspondants peuvent s'entendre sur "AFFINE(13,14)", "AFFINE(12,1)", "AFFINE(17,4)". Ici c'est à la quatrième lettre que l'on reprend "AFFINE(13,14)".

Les clés peuvent être plus longues : une phrase d'un ouvrage, plusieurs phrases, mais ceci complique évidemment chiffement et déchiffrement. On espère, qu'en compliquant, le décryptage sera plus difficile.

7.4 Le chiffrement de Vernam

De plus en plus les messages sont transmis écrits en base 2 puisque l'ordinateur ne connaît que le 0 et le 1, même si on n'en a pas conscience (téléphone numérique). Aussi il peut être intéressant de chiffrer des messages écrits en binaire. Un ingénieur américain Gérard Vernam a proposé l'algorithme de chiffrement suivant :
L'expéditeur et le destinataire du message, possédent en commun une clé qui est aussi un message binaire, par exemple

$$K := 01101010110010100010100101 10.$$

Soit

$$M := 1010001000110101,$$

le message à émettre. Désignons par K_i et M_i les ièmes éléments des messages. Nous les considérons comme éléments de $\mathcal{E}_2$ et c'est dans cet ensemble que nous travaillons. On a donc les opérations simples suivantes :

$$0+1=1;\ 0+0=1+1=0.$$

Le cryptogramme C sera défini par $C_i := M_i + K_i$, ce qui nécessite que la longueur de la clé soit plus grande que celle du message.
Le cryptogramme de l' exemple est alors :

$$\begin{array}{lcccccccccc} M & := & 101 & \| & 000 & \| & 100 & \| & 011 & \| & 010 & \| & 1 \\ K & := & 011 & \| & 010 & \| & 101 & \| & 100 & \| & 101 & \| & 0 \\ C & = & 110 & \| & 010 & \| & 001 & \| & 111 & \| & 111 & \| & 1 \end{array}$$

Le déchiffrement est facile. On utilise la propriété :

$$C_i + K_i = (M_i + K_i) + K_i = M_i;$$

ainsi le même algorithme permet de déchiffrer.

$$\begin{array}{lcccccccccc} C & := & 110 & \| & 010 & \| & 001 & \| & 111 & \| & 111 & \| & 1 \\ K & := & 011 & \| & 010 & \| & 101 & \| & 100 & \| & 101 & \| & 0 \\ M & = & 101 & \| & 000 & \| & 100 & \| & 011 & \| & 010 & \| & 1 \end{array}$$

La difficulté dans cette méthode est la construction et le stockage de longues clés dites aléatoires, c'est-à-dire où la connaissance de tout morceau de la clé ne permet pas de l'avoir en entier... mais cela est un autre problème.

7.5 Un algorithme de chiffrement plus mathématique

7.5.1 Un exemple

Reprenons toujours le message du premier chapitre :

RENDEZ-VOUS VENDREDI SOIR

codé avec les 26 premiers nombres entiers.

170413 030425 211420 182104 130317 040308 181408 17.

Nous allons décrire une nouvelle méthode de chiffrement basée sur le théorème de Fermat. Choisissons le nombre premier $p = 971$ et un entier $c = 9$ appelé **exposant de chiffrement**. Décomposons le clair en nombres de trois chiffres, qui, comme on le constate, sont tous inférieurs à 970.

170 | 413 | 030 | 425 | 211 | 420 | 182 | 104 | 130 | 317 | 040 | 308 | 181 | 408 | 173 |

Nous avons mis un 3 au hasard pour compléter le dernier nombre à trois chiffres. A chaque nombre a de 3 chiffres, nous allons faire correspondre $b := a^9 \bmod 971$; on applique **"PUISS9"**. Ceci donne :

071 | 160 | 852 | 585 | 721 | 076 | 079 | 150 | 949 | 642 | 920 | 651 | 194 | 503 | 035 |

Comme vous le remarquez, les nombres de deux chiffres que l'on obtient sont codés par trois chiffres de façon à ce que l'on puisse déchiffrer correctement. Ainsi l'image de 420 est 76 que l'on écrit 076. Si nous avions obtenu 3, nous aurions écrit 003.
Pour l'envoi, on peut regrouper de façon différente, par nombres de dix chiffres par exemple :

0711608525 8572107607 9150949642 9206511945 03035

Comment déchiffrer le cryptogramme ainsi reçu. Supposons connaître p et $d = 539$, **l'exposant de déchiffrement.** Alors pour chaque groupe b de trois chiffres reçus, on applique "PUISS539" c'est-à- dire que l'on calcule b^{539}. Ainsi $071^{539} \bmod p = 170, 160^{539} \bmod p = 413, \ldots$, ce qui permet de retrouver le codé du message, que l'on transforme ensuite en son expression littérale, si besoin est.
Pourquoi cela marche-t-il ?
Nous remarquons d'abord que $cd \equiv 1 \pmod{(p-1)}$; en effet $9 * 539 - 1$ est divisible par 970. Ceci permet d' appliquer (cor.5, 6.6.2); on peut donc écrire pour tout $a \in \mathbb{Z}$, $(a^c)^d = a^{cd} = a$.
Pour trouver 539, on peut utiliser l'algorithme de 6.5.3.

7.5.2 Le cadre général

Marie et Louise s'entendent sur un triplet (p, c, d), p étant un nombre premier, c et d deux entiers inférieurs à $(p-2)$ et inverse l'un de l'autre modulo $(p-1)$. Un message

élémentaire est un entier M inférieur à $(p-1)$ et les algorithmes de chiffrement et de déchiffrement sont "PUISSc" et "PUISSd". La composition des deux fonctions $x \mapsto x^c$ et $x \mapsto x^d$, donnant l'identité d'après le corollaire 5 du petit théorème de Fermat.
Le couple (p, c) ne peut être divulgué car alors d peut être calculé; nous avons ici un chiffrement à clé secrète.

7.5.3 Un autre exemple

Reprenons l'exemple 3 du paragraphe 3.6.
Soit $p = 1\ 234\ 567\ 891$ et $c = 167$. Son inverse modulo $(p-1)$ est 539 661 413.
Nous décomposons notre message en messages élémentaires qui sont des nombres de 9 chiffres pour que ces nombres soient inférieurs à p. Ces messages sont transformés en nombres de 10 chiffres et il faudra bien faire attention de mettre un nombre suffisant de zéros en préfixe pour que tous les nombres soient codés sur 10 chiffres.

M	C
170 413 030	1 119 420 390
425 211 420	0 101 124 399
182 104 130	0 696 865 493
317 040 308	0 438 759 538
181 408 173	0 353 722 240

Le cryptogramme est donc :

1 119 420 3900 101 124 3990 438 759 5380 353 722 240

7.6 L'algorithme RSA

7.6.1 Le cadre général

Marie désire recevoir des messages de ses amies, mais elle voudrait être la seule à pouvoir les lire. Evidemment, elle et ses amies sont accroc d'Internet et correspondent par courrier électronique. Elle va utiliser l'algorithme de chiffrement RSA du nom de trois chercheurs, Rivest, Shamir et Adleman, algorithme mis au point en 1978.
Marie détermine deux nombres premiers distincts de 50 chiffres. Comment fait-elle ? Elle choisit un nombre de 50 chiffres au hasard, en évitant tout nombre pair et utilise son ordinateur pour tester s'il est premier. Il existe actuellement des algorithmes performants et rapides donnant la réponse, et comme il y a assez de nombres premiers, elle va en trouver un. Comme vous le voyez, on passe par dessus de nombreuses questions théoriques que nous ne pouvons expliquer ici.
Elle dispose donc de deux nombres premiers p et q ; Elle calcule $n = pq$, puis $m = (p-1)(q-1)$, puis elle peut jeter p et q, mais alors elle ne pourra plus les retrouver. En effet, factoriser n un nombre de 100 chiffres ou factoriser m est pratiquement, à l'heure actuelle, impossible.

Elle choisit maintenant un nombre d premier avec m. Pour cela elle prend d au hasard tel que $2 \leq d \leq (m-2)$ et calcule $pgcd(m,d)$. Si elle obtient 1, elle garde d, sinon elle recommence et comme il a été montré - pas par nous - qu'il y a beaucoup de chance de tomber sur un nombre premier avec m, cette recherche est assez rapide. Elle calcule c, l'inverse de d modulo m. Pour cela pas de problème, l'algorithme 6.6.3 donne la réponse. Marie envoie à toutes ses amies la clé publique (n,c), clé qui peut être interceptée par n'importe qui, ce n'est pas grave. Elle garde pour elle (n,d) (et c, s'il lui prend l'envie de s'envoyer des messages).
Louise décide d'envoyer à Marie un message. Le message élémentaire de base est un nombre m compris entre 0 et $(n-1)$. Expliquons d'abord comment envoyer un message élémentaire.
Louise calcule à l'aide de son ordinateur : $C := M^c \bmod n$. Elle envoie C à Marie. Cette dernière, pour déchiffrer calcule $M' := C^d \bmod n$. D'après (th.7, 6.6.2), on a

$$M' \equiv C^d \equiv M^{(cd)} \equiv M \pmod{n},$$

et par suite $M' = M$ puisque ces deux nombres sont contenus dans le même intervalle de longueur n. Marie a donc recouvré le message de Louise.

7.6.2 Un exemple simple

Considérons toujours le message :

RENDEZ-VOUS VENDREDI SOIR.

Marie choisit $p = 971$ et $q = 977$, deux nombres premiers. Elle calcule $n = pq = 948\ 667$, puis $m = (p-1)(q-1) = 946\ 720$. Elle choisit $c = 9$ et transmet à ses amies le couple $(948\ 667, 9)$. Elle calcule, en utilisant l'algorithme du calcul de l'inverse vu en 6.6.3, l'inverse d de 9 modulo 946 720 et trouve $d = 841\ 529$.
Louise veut envoyer le message ci-dessus à Marie. Elle le code, comme il a déjà été vu, à l'aide de nombres représentant la position de la lettre dans l'alphabet et découpe le message ainsi obtenu en messages élémentaires de 6 chiffres, de façon à ce que chaque message élémentaire soit inférieur à $n = 948\ 667$.

REN	*DEZ*	*VOU*	*SVE*	*NDR*	*EDI*	*SOI*	*R*
170 413	030 425	211 420	182 104	130 317	040 308	181 408	17

Louise calcule $C := M^9 \bmod 948\ 667$ pour chaque message élémentaire.

M	C
170 413	947 841
030 425	909 875
211 420	535 838
182 104	573 617
130 317	905 809
040 308	658 606
181 408	032 768
179 999	890 073

Le message transmis à Marie est :

947 841 909 875 535 838 573 617 905 809 658 606 032 768 890 073

A la réception de ce message, Marie se hâte (que peut bien lui dire Louise ?) de calculer pour chaque groupe C de 6 chiffres $M := C^{841\,521} \bmod 948\,667$ pour trouver le message en clair de Louise.
Comme on le voit, il y a de gros calculs à faire et il est nécessaire d'avoir un ordinateur et aussi d'apprendre comment on peut calculer le plus rapidement possible de telles puissances. Disons seulement ici qu'une méthode utilise l'écriture en base 2 de l'exposant; c'est donc une application du chapitre sur la numération.
Dans notre exemple n est petit et peut donc facilement être factorisé. Il faut donc que le couple (n, c) soit secret; c'est l'algorithme de chiffrement du paragraphe 5 où l'on a simplement remplacé un nombre premier par un nombre composé.

7.6.3 Un autre exemple

Dans ce nouvel exemple, il est déjà beaucoup plus difficile de factoriser n car l' entier considéré a 19 chiffres; nous allons décomposer notre message - toujours le même - en nombres de 18 chiffres. Nous complétons le dernier message afin d'avoir effectivement 18 chiffres. Le cryptogramme comporte 57 chiffres alors que le clair en a 45.

On considère $p = 1\,234\,567\,891$ et $q = 987\,654\,323$ on a alors

$$n = pq = 1\,219\,326\,314\,583\,142\,793$$

et

$$m = (p-1)(q-1) = 1\,219\,326\,312\,360\,920\,580$$

Pour $c = 163$ il vient $d = 920\,105\,131\,413\,455\,407$

M	C
170 413 030 425 211 420	944 990 357 097 553 312
182 104 130 317 040 308	164 883 429 666 048 921
181 408 171 234 567 890	611 524 223 493 407 257

Le cryptogramme est donc

944 990 357 097 553 312 164 883 429 666 048 921 611 524 223 493 407 257

7.6.4 Sécurité de l'algorithme

RSA est un algorithme public et la clé est publique. Comment peut-il être un algorithme de chiffrement ?

Un curieux intercepte le message chiffré par Louise. Peut-il le décrypter ? Il connait le couple (n, c), puisque ce couple est public ou du moins peut être divulgué. S'il veut

procéder comme Marie il doit calculer d, l'inverse de c modulo m.
Peut-il trouver m ?
On a

$$m = (p-1)(q-1) = pq - (p+q) + 1 = n - (p+q) + 1$$

n étant connu, connaître m, c'est connaître $(p+q)$. Comme $n = pq$, la connaissance de m revient à la connaissance de p et q puisque si l'on connait la somme et le produit de deux nombres, on connait ces deux nombres. Ainsi trouver m revient à factoriser n.
Peut-il trouver d sans connaître m ?
Peut-il décrypter le message sans connaître m et d ?
On ne sait répondre actuellement à ces questions mais il est raisonnable de penser qu'il est nécessaire de connaître m et donc qu'il est nécessaire de savoir factoriser n.
Ainsi, la sécurité de l'algoritme est-elle liée à la difficulté se factoriser n. Dans la pratique, on prend des modules p et q de 256 bits, ce qui assure l'inviolabilité du chiffrement.

7.7 Notion de signature

7.7.1 La signature

Jusqu'ici, nous nous sommes intéressés à l'apport du chiffrement à la **confidentialité** des messages. Ce n'est plus maintenant la seule utilisation ; ainsi l'algorithme RSA peut aussi être utilisé comme **algorithme de signature.** Qu'est-ce dire ?
Il n'est pas toujours important de dissimuler le message que l'on envoie, parfois ce qui est le plus important c'est de savoir que le message a correctement été reçu et que le destinataire soit sûr du nom de l'expéditeur. "Verser 1 000 francs sur le compte de Pauline et débiter mon compte de la dite somme, signé : Marie". Avant de s'exécuter la banque doit s'assurer que c'est bien Marie qui a envoyé le message et d'autre part qu'il n'y a pas eu au cours de l'échange modification de la somme. La banque doit pouvoir s'assurer de l'**identité** de l'expéditeur et de **l'intégrité** du contenu du message. La confidentialité n'est pas ici importante.

Supposons que Marie a publié un module RSA (n, c) et a conservé l'exposant secret d. Elle code son message puis lui applique "PUISSd" qu'elle seule connait. On dit qu'elle **signe** le message. La banque déchiffre avec "PUISSc" et reconstruit ainsi le message. La banque est ainsi assurée que seule Marie a pu le signer, car elle seule connait d. Evidemment tout le monde peut aussi déchiffrer, mais ce n'est pas grave car le message peut être connu de tous. Comme on le voit la signature que l'on appelle électronique est plus générale que la signature manuelle. Cette dernière est toujours la même ou presque et peut facilement être imitée. La signature que l'on vient de décrire dépend du contenu du message.

7.7.2 Signature et chiffrement

Si l'on veut signer et chiffrer en même temps, c'est possible.
Marie publie un module RSA (n, c) et garde le secret d. Louise fait de même : elle publie

(N, C) et garde D. On suppose $n \leq N$.
Marie veut envoyer un message à Louise, message signé et chiffré. Le message de base est un nombre M tel que $0 \leq M \leq n-1$. Elle signe d'abord le message en utilisant "PUISSd" dans $\mathcal{E}_n$, puis elle chiffre le résultat par "PUISSC" dans $\mathcal{E}_N$.
Elle calcule donc successivement :

$$M' := M^d \bmod n$$

et comme M' est inférieur à n donc à N, elle peut ensuite calculer

$$M'' := M'^C \bmod N.$$

Louise reçoit M", message qu'elle déchiffre en utilisant d'abord "PUISSD" dans $\mathcal{E}_N$ puis "PUISSc" dans $\mathcal{E}_n$.

Maintenant, c'est Louise qui veut envoyer un message à Marie, dans les mêmes conditions. Elle choisit encore son message M inférieur à n. Elle le chiffre d'abord avec "PUISSc" dans $\mathcal{E}_n$, puis elle le signe avec "PUISSD" dans $\mathcal{E}_N$. Marie pour obtenir le clair, utilise d'abord "PUISSC" dans $\mathcal{E}_N$ puis "PUISSd" dans $\mathcal{E}_n$.

Un exemple simple :
prenons

$$(n, c, d) := (77, 7, 43)$$
$$(N, C, D) := (143, 17, 113)$$

Marie veut envoyer le message $M = 2$. Elle le transforme par "PUISS43" dans $\mathcal{E}_{77}$ et obtient $M' = 30$, auquel elle applique "PUISS17" dans $\mathcal{E}_{143}$; elle envoie $M'' = 101$.
Louise calcule alors $101^{113} \bmod 143$ (elle est la seule à pouvoir le faire), obtient 30, puis $30^7 \bmod 77$; elle obtient le message $M = 2$.

Le même message envoyé par Louise à Marie est tranformé comme suit : $M' = 2^{17} \bmod 143$, soit $M' = 84$, puis $M'' = 84^{43} \bmod 77 = 35$. C'est le message qu'elle transmet à Marie.

Ces méthodes ont des applications en monnaie electronique, en commerce électronique et pour le notariat électronique (certification de documents écrits).

Pour en savoir plus

- **En Arithmétique et Algèbre**

  - H. Cohen : *A Course in Computational Algebraic Number Theory.* Springer-Verlag, GTM 138,1993.
  - G.H. Hardy, E.M. Wright : *An Introduction to the Theory of Numbers.* Clarendon Press,1960-1979, (5th edition).
  - K. Ireland, M. Rosen : *A Classical Introduction to Modern Number Theory.* Springer-Verlag, New-York, 2nd edition, 1990.
  - P. Naudin, C. Quitté : *Algorithmique Algébrique.* Masson, 1992.
  - H. Riesel : *Prime Numbers and Computers Methods for Factorization.* Birkaüser Boston Inc., 1985.

- **En Cryptographie**

  - D.E.R. Denning : *Cryptography and data Security.* Addison-Wesley,1982.
  - D. Kahn : *The Codebrakers : the story of secret writing.* Macmillan Publ. Co. Inc., New-York,1977.
  - A.G. Konheim : *Cryptography, A Primer.* John Wiley and sons, 1981.
  - Menezes : *Handbook of Applied Cryptography,* Crc Press, ISBN : 0-8493-8523-7.
  - G. Robin : *Algorithmique et Cryptographie.* Mathématiques et Applications, 8, 1991, S.M.A.I., Ellipses.
  - Schneeier : *Cryptographie appliquée,* 2nd édition, ISBN : 2-84180-036-9.
  - D. Stinson : *Cryptographie : Theorie et Pratique.* International Thomson Publishing, Traduit par S.Vaudenay, 1996.

Annexe

On propose ici les programmes écrits en MAPLE de tous les algorithmes décrits dans l'ouvrage.

• # Algorithmes du chapitre 1:

```
• alphabet := ABCDEFGHIJKLMNOPQRSTUVWXYZ;
```

```
# Affecte un chiffre a une lettre
#
• for ii  from 1 to 26 do
numero [substring (alphabet,ii..ii)]:= ii-1;od;
```

```
•chif_let :=

proc(messagechiffre)
local i,messagelettre;
#cette procedure transforme un message ecrit a l'aide de chiffres
#en meesage ecrit a l'aide de lettres.
    messagelettre :=
        substring(alphabet,messagechiffre[1]+1 .. messagechiffre[1]+1);
    for i from 2 to nops(messagechiffre) do
        messagelettre := cat(messagelettre,
           substring(alphabet,messagechiffre[i]+1 .. messagechiffre[i]+1)
           )
    od;
    messagelettre
end:
```

```
let_chif := proc(messagelettre)
            local i,messagechiffre,nombre;
#cette procedure transforme un message ecrit a l'aide de lettres
#en meesage ecrit a l'aide de chiffres.
                messagechiffre := [];
                for i to length(messagelettre) do
                    nombre := numero[substring(messagelettre,i .. i)];
                    messagechiffre := [op(messagechiffre),nombre]
                od;
                messagechiffre
            end:
```

```
• plus := proc(a,liste)
       local i,Y,crypto;
#cette procedure travaille sur les chiffres
           crypto := [];
           for i to nops(liste) do
               Y := liste[i]+a;
               if 25 < Y then Y := Y-26 fi;
               crypto := [op(crypto),Y]
           od;
```

```
        crypto
    end:
```

• pluslettre :=

```
  proc(a,liste)
  local ll;
#cette procedure travaille sur les lettres
      ll := let_chif(liste); ll := plus(a,ll); ll := chif_let(ll); ll
  end:
```

• clair1 := RENDEZVOUSVENDREDISOIR:
clair2 := BONJOURCHERSAMISPOITEVINS:

• crypto1:= pluslettre(2,clair1);
crypto2:= pluslettre(4,clair2);

crypto1 := TGPFGBXQWUXGPFTGFKUQKT

crypto2 := FSRNSYVGLIVWEQMWTSMXIZMRW

• mult := proc(a,l)
```
       local Y,crypto;
#cette procedure travaille sur les chiffres
            crypto := [];
            for i to nops(l) do
                Y := l[i]*a;
                if 25 < Y then Y := Y mod 26 fi;
                crypto := [op(crypto),Y]
            od;
            crypto
        end:
```

• multlettre :=

```
  proc(a,l)
  local ll;
#cette procedure travaille sur les lettres
      ll := let_chif(l); ll := mult(a,ll); ll := chif_let(ll); ll
  end:
```

• crypto3:=multlettre(3,clair1);
crypto4:=multlettre(5,clair2);

crypto3 := ZMNJMXLQICLMNJZMJYCQYZ

crypto4 := FSNTSWHKJUHMAIOMXSORUBONM

```
crypto:= UFWKFNYATZXFAJEHTRUWNX:
```

```
• pluslettre(21,crypto);
                              PARFAITVOUSAVEZCOMPRIS
```

```
• crypto:= KUNUMRXAMUNKSHURHSXPOZZOKODU:
```

```
• multlettre(21,crypto);
                           CENESTPASENCORETROPDIFFICILE
```

```
•  # Algorithmes du chapitre 3:
```

```
• euclide := proc(a,b)
 # donne le pgcd de deux entiers
          local A,B,R,d;
              A := a;
              B := b;
              while B <> 0 do  R := A mod B; A := B; B := R od;
              d := A
          end:
```

```
• euclide(1256,2520);
                                        8
```

```
•euclidebezout := proc(a,b)
                  local s0,s1,t0,t1,A,B,S,T,Q;
#calcule le pgcd de deux entiers a et b et un couple d'entiers (x,y) tel
que ax + by = d
                        s0 := 0;
                        s1 := 1;
                        t0 := 1;
                        t1 := 0;
                        A := a;
                        B := b;
                        if b = 0 then d := a; x := 1; y := 0
                        else
                            while 0 < B do
                                R := A mod B;
                                Q := (A-R)/B;
                                S := s0-Q*s1;
                                T := t0-Q*t1;
                                A := B;
                                B := R;
                                s0 := s1;
                                s1 := S;
                                t0 := t1;
                                t1 := T
```

```
            od
        fi;
        d := A;
        x := t0;
        y := s0;
        d,x,y
    end:
```

```
• euclidebezout(128,37);
                                1, -13, 45
```

```
• euclidebezout(261,42);
                                3, 5, -31
```

```
• euclidebezout(1234567890,167);
                            1, -73, 539661413
```

• # Algorithmes du chapitre 4:

```
• erato := proc(N)
        local i,j,q,k,t,suite;
 # cette procedure est le crible d'Erathostene
            i := 2;
            while i < N+1 do  t[i] := 1; i := i+1 od;
            j := 2;
            q := evalf(sqrt(N));
            while j < q do
                k := 2*j;
                while k < N+1 do  t[k] := 0; k := k+j od;
                j := j+1;
                while t[j] = 0 do  j := j+1 od
            od;
            suite := [];
            i := 2;
            while i < N+1 do
                if t[i] <> 0 then suite := [op(suite),i] fi; i := i+1
            od;
            suite
        end:
```

```
• premier := erato(300);
 premier := [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, 199, 211, 223, 227, 229, 233, 239, 241, 251, 257, 263, 269,

271, 277, 281, 283, 293]
```

```
•primalite :=

    proc(N)
    local i,p, booleen;
 # cette procedure teste si un nombre est premier a l'aide
 # des divisions successives
        i := 1;
        p := 2;
        booleen := faux;
        while (p <= evalf(sqrt(N))) and (booleen = faux) do
            if N mod p = 0 then print(p,divise,N); booleen := vrai
            else i := i+1; p := premier[i]
            fi
        od;
        if booleen = faux then print(N,` est premier`) fi
    end:
```

```
• primalite(76);primalite(151);
                              2, divise, 76

                            151,  est premier
```

```
•factorisation :=

    proc(n)
    local i,N,A,p,j;
 # cette procedure founit la factorisation d'un nombre
        i := 1;
        N := n;
        A[i] := 0;
        while N <> 1 do
            p := premier[i];
            while N mod p = 0 do  A[i] := A[i]+1; N := N/p od;
            i := i+1;
            A[i] := 0
        od;
        j := 1;
        while j < i do
            if A[j] <> 0 then print(premier[j],A[j]) fi; j := j+1
        od;
        print(fin)
    end:
```

```
• factorisation (2^5*3^7*5^8*11);
                                    2, 5

                                    3, 7

                                    5, 8

                                   11, 1

                                    fin
•  # Algorithmes du chapitre 5;
:
```

```
• Ch_B10 :=

    proc(Base,Liste)
    local i,X,n,long,b;
#cette procedure permet d'ecrire en base 10 un nombre ecrit
# dans une autre base
        i := 1;
        long := nops(Liste);
        n := 0;
        b := 1;
        while i <= long do  n := n+Liste[i]*b; b := b*Base; i := i+1 od;
        n
    end:
```

```
• nombre16 := [5,7,3,11]:
```

```
• N:=Ch_B10(16,nombre16);
                                N := 45941
```

```
•Ch1_10B := proc(Base,a)
          local i,X,n,long,liste,b;
#cette procedure permet d'ecrire un nombre donne en base 10
# dans une autre base
              i := 1;
              n := a;
              liste := [];
              while 0 < n do
                  b := n mod Base;
                  liste := [op(liste),b];
                  i := i+1;
                  n := (n-b)/Base
              od;
              liste
          end:
```

```
• Ch1_10B(16, N);
                                [5, 7, 3, 11]
```

```
• Ch2_10B := proc(Base,a)
          local i,X,n,liste,c,b,k;
#C'est la deuxieme procedure qui  permet d'ecrire un nombre donne en base
10
# dans une autre base
               k := 0;
               n := a;
               X := Base;
               while X < n do  k := k+1; X := X*Base od;
               X := X/Base;
               i := k;
               liste := [];
               while 1 <= i do
                   c := n mod X;
                   b := (n-c)/X;
                   liste := [op(liste),b];
                   i := i-1;
                   n := c;
                   X := X/Base
               od;
               liste := [op(liste),n];
               liste
          end:
```

```
• Ch2_10B(16,N);
                                [11, 3, 7, 5]

•  # Algorithmes du chapitre 6
;
```

```
• euclidebezout := proc(a,b)
                 local s0,s1,t0,t1,A,B,S,T,Q;
#donne le pgcd de deux entiers a et b et un couple d'entiers (x, y) tel
*que ax + by = d
                     s0 := 0;
                     s1 := 1;
                     t0 := 1;
                     t1 := 0;
                     A := a;
                     B := b;
                     if b = 0 then d := a; x := 1; y := 0
                     else
                         while 0 < B do
                             R := A mod B;
                             Q := (A-R)/B;
                             S := s0-Q*s1;
                             T := t0-Q*t1;
```

```
                    A := B;
                    B := R;
                    s0 := s1;
                    s1 := S;
                    t0 := t1;
                    t1 := T
                od
            fi;
            d := A;
            x := t0;
            y := s0;
            d,x,y
        end:
```

```
• inverse := proc(a,n)
        local b,x;
# calcule l'nverse de a modulo n
          b := euclidebezout(a,n);
            if 1 < b[1] then print(impossible)
            else x := b[2]; if x < 0 then x := x+n fi; print(x)
            fi
        end:
```

```
• inverse(1256, 181489561);
                                96958197
```

```
• inverse (1356,356931);
                                impossible
```

```
•  # Algorithmes du chapitre 7
;
```

```
• vernam := proc(cle,clair)
        local a,q,k,i,c,x;
# cete procedure realise l'algoritme de Vernam
            longcle := nops(cle);
            longclair := nops(clair);
            a := longclair mod longcle;
            q := (longclair-a)/longcle;
            k := 0;
            crypto := [];
            while k < q do
                c := k*longcle;
                i := 1;
                while i <= longcle do
                    x := 1;
                    if clair[c+i] = cle[i] then x := 0 fi;
                    crypto := [op(crypto),x];
```

```
                i := i+1
            od;
            k := k+1
        od;
        c := k*longcle;
        i := 1;
        while i <= a do
            x := 1;
            if clair[c+i] = cle[i] then x := 0 fi;
            crypto := [op(crypto),x];
            i := i+1
        od;
        crypto
    end:
```

```
• cle:= [0,1,1,0,1,0,1,0,1,1,0,0,1,0,1,0]:
```

```
• clair:= [1,0,1,0,0,0,1,0,0,0,1,1,0,1,0,1]:
```

```
• crypto:=vernam(cle,clair);
       crypto := [1, 1, 0, 0, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1]
```

```
• M1:= vernam(cle, crypto);
         M1 := [1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 1, 0, 1, 0, 1]
```

```
•
clair2:=[1,0,1,0,0,0,1,0,0,0,1,1,0,1,0,1,0,0,0,1,1,0,1,1,0,0,0,1,0,0,0,0,1,
1,1,1,1,1,1,0,1,0,0]:
```

```
• crypto2:= vernam(cle,clair2);
 crypto2 := [1, 1, 0, 0, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1,

     0, 0, 0, 1, 1, 1, 0, 1, 1, 0, 1, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0]
```

```
• vernam(cle,crypto2);
[1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 1, 0, 1, 0, 1, 0, 0, 0, 1, 1, 0, 1, 1,

    0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0, 0]
```

```
•rsa :=

   proc(n,e,clair)
   local i,long,a;
# C'est l'algoritme RSA
       long := nops(clair);
       i := 1;
       crypto := [];
```

```
    while i <= long do
        a := (clair[i] &^ e) mod n; crypto := [op(crypto),a]; i := i+1
    od;
    crypto
end:
```

```
• clair :=[170413,030425,211420,182104,130317,040308,181408,179999]:
```

```
• crypto:=rsa(948667,9,clair);
crypto := [947841, 909875, 535838, 573617, 905809, 658606, 32768, 890073]
```

```
• cc:= rsa(948667,841529,crypto);
   cc := [170413, 30425, 211420, 182104, 130317, 40308, 181408, 179999]
```

```
• n:= 1234567891*987654323;
                          n := 1219326314583142793
```

```
• e:=163:
```

```
• d:=920105131413455407:
```

```
• clair2:=[170413030425211420,182104130317040308,181408171234567890]:
```

```
• crypto2:=rsa(n,e,clair2);
 crypto2 := [944990357097553312, 164883429666048921, 611524223493407257]
```

```
• ccc:= rsa(n,d,crypto2);
   ccc := [170413030425211420, 182104130317040308, 181408171234567890]
```

I. Auteur

ROBIN Guy

II. Titre

Apprenons l'arithmétique élémentaire pour comprendre la cryptographie moderne

III. Caratéristique

Édité par l'Institut de Recherche sur l'Enseignement des Mathématiques (IREM) de LIMOGES en 1998
Format : B5, 87 p.
ISBN : 2-910 165-07-8

IV. Type de documents et support

Type : vulgarisation
Support : papier

V. Matériel utilisé dans l'ouvrage

Logiciel de calcul formel sur ordinateur MAPLE

VI. Public visé

élève, étudiant, enseignant
NIV : Terminale scientifique, enseignement supérieur
AGE : 17, 18

VII. Contenus

RÉSUMÉ : Le but de cet ouvrage est de présenter les premiers éléments de cryptographie moderne et parallèlement d'introduire l'arithmétique nécessaire à la compréhension des protocoles construits.
Pour aborder cet ouvrage il suffit de savoir compter et lire, il reste à comprendre.
Après avoir expliqué ce qu'est la cryptographie nous décrivons l'outil mathématique :

- la notion de diviseur, de pgcd et de ppcm ;
- la division euclidienne ;
- les nombres premiers ;
- le calcul modulo un entier ;
- le petit théorème de Fermat et ses applications.

Nous avons particulièrement insisté sur l'aspect effectif et algorithmique car dans le dernier chapitre où nous revenons à la cryptographie, les protocoles de chiffrement et de signature sont décrits et mis en œuvre.
En annexe nous donnons des programmes rédigés en MAPLE.

Bibliogr. p.79

MCL : algorithmique, arithmétique, Bézout, calcul modulaire, chiffrement, cryptographie, diviseurs, division euclidienne, Euclide, Fermat, Gauss, nombres premiers, numération, pgcd, ppcm, protocole, signature.