Introduction à la Cryptologie

Chapitre 2 : Arithmétique des nombres entiers

Michael Eisermann (Institut Fourier, UJF Grenoble)

Année 2008-2009
IF / IMAG, Master 1, S1-S2

document mis à jour le 7 juillet 2009

www-fourier.ujf-grenoble.fr/~eiserm/cours # crypto

Objectifs

Questions de base :

1. Qu’est-ce que les nombres entiers ?
2. Comment les implémenter sur ordinateur ?

Objectifs

Questions de base :

1. Qu’est-ce que les nombres entiers ?
2. Comment les implémenter sur ordinateur ?

Développement mathématique :

- Retracer le fondement axiomatique.
- Définir les opérations arithmétiques et établir leurs propriétés.
- Introduire notamment la division euclidienne des entiers.

Objectifs

Questions de base :

1. Qu'est-ce que les nombres entiers ?
2. Comment les implémenter sur ordinateur ?

Développement mathématique :

- Retracer le fondement axiomatique.
- Définir les opérations arithmétiques et établir leurs propriétés.
- Introduire notamment la division euclidienne des entiers.

Développement algorithmique :

- La numération positionnelle est une représentation efficace.
- Établir des algorithmes pour les opérations arithmétiques.
- Estimer leur complexité : coût en temps et en mémoire.

Sommaire

Numération romaine utilisée jusqu'au moyen âge :

$$I, II, III, IV, V, VI, VII, VIII, IX, X, \ldots$$

Qu'est-ce que les nombres naturels ?

Numération romaine utilisée jusqu'au moyen âge :

$$I, II, III, IV, V, VI, VII, VIII, IX, X, \ldots$$

Numération indo-arabe utilisée de nos jours :

$$0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, \ldots$$

Numération romaine utilisée jusqu'au moyen âge :

$$I, II, III, IV, V, VI, VII, VIII, IX, X, \ldots$$

Numération indo-arabe utilisée de nos jours :

$$0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, \ldots$$

Quatre opérations arithmétiques :

$$\begin{array}{rr} \text{DXXXVII} & 537 \\ +\text{LXXIX} & +79 \\ \hline \text{DMXVI} & 616 \end{array}$$

Qu'est-ce que les nombres naturels ?

Numération romaine utilisée jusqu'au moyen âge :

$$I, II, III, IV, V, VI, VII, VIII, IX, X, \dots$$

Numération indo-arabe utilisée de nos jours :

$$0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, \dots$$

Quatre opérations arithmétiques :

$$\begin{array}{rr} \text{DXXXVII} & 537 \\ +\text{LXXIX} & +79 \\ \hline \text{DMXVI} & 616 \end{array} \qquad \begin{array}{rr} \text{MMVIII} & 2008 \\ -\text{LXXIX} & -79 \\ \hline \text{MCMXXIX} & 1929 \end{array}$$

Numération romaine utilisée jusqu'au moyen âge :

$$I, II, III, IV, V, VI, VII, VIII, IX, X, \ldots$$

Numération indo-arabe utilisée de nos jours :

$$0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, \ldots$$

Quatre opérations arithmétiques :

$$\begin{array}{rr} \text{DXXXVII} & 537 \\ +\text{LXXIX} & +79 \\ \hline \text{DMXVI} & 616 \end{array} \qquad \begin{array}{rr} \text{MMVIII} & 2008 \\ -\text{LXXIX} & -79 \\ \hline \text{MCMXXIX} & 1929 \end{array}$$

$$\begin{array}{rr} \text{XLIII} & 43 \\ \times\ \text{LXXIX} & \times 79 \\ \hline \text{MMMXXXIIIX} & 3397 \end{array}$$

Qu'est-ce que les nombres naturels ?

Numération romaine utilisée jusqu'au moyen âge :

$$I, II, III, IV, V, VI, VII, VIII, IX, X, \dots$$

Numération indo-arabe utilisée de nos jours :

$$0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, \dots$$

Quatre opérations arithmétiques :

$$\begin{array}{rr} \text{DXXXVII} & 537 \\ +\text{LXXIX} & +79 \\ \hline \text{DMXVI} & 616 \end{array} \qquad \begin{array}{rr} \text{MMVIII} & 2008 \\ -\text{LXXIX} & -79 \\ \hline \text{MCMXXIX} & 1929 \end{array}$$

$$\begin{array}{rr} \text{XLIII} & 43 \\ \times\ \text{LXXIX} & \times 79 \\ \hline \text{MMMXXXIIIX} & 3397 \end{array} \qquad \begin{array}{rr} \text{MMMXXXIIIX} & 3397 \\ \div\ \text{LXXIX} & \div 79 \\ \hline \text{XLIII} & 43 \end{array}$$

Qu'est-ce que les nombres naturels ?

Numération romaine utilisée jusqu'au moyen âge :

$$I, II, III, IV, V, VI, VII, VIII, IX, X, \dots$$

Numération indo-arabe utilisée de nos jours :

$$0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, \dots$$

Quatre opérations arithmétiques :

$$\begin{array}{rr} \text{DXXXVII} & 537 \\ +\text{LXXIX} & +79 \\ \hline \text{DMXVI} & 616 \end{array} \qquad \begin{array}{rr} \text{MMVIII} & 2008 \\ -\text{LXXIX} & -79 \\ \hline \text{MCMXXIX} & 1929 \end{array}$$

$$\begin{array}{rr} \text{XLIII} & 43 \\ \times\ \text{LXXIX} & \times 79 \\ \hline \text{MMMXXXIIIX} & 3397 \end{array} \qquad \begin{array}{rr} \text{MMMXXXIIIX} & 3397 \\ \div\ \text{LXXIX} & \div 79 \\ \hline \text{XLIII} & 43 \end{array}$$

Ceci n'est pas une *définition* : il ne s'agit que *d'exemples*.

Sommaire

Sommaire

Sommaire

Les nombres naturels sont l'abstraction du processus de *compter* :

Heuristique

Les nombres naturels sont l'abstraction du processus de *compter* :

- Le nombre initial est noté 0.

Heuristique

Les nombres naturels sont l'abstraction du processus de *compter* :

- Le nombre initial est noté 0.
- À chaque nombre naturel n on associe son successeur $\mathbf{S}n$.

Heuristique

Les nombres naturels sont l'abstraction du processus de *compter* :

- Le nombre initial est noté 0.
- À chaque nombre naturel n on associe son successeur $\mathbf{S}n$.

Ceci permet déjà de nommer quelques petits exemples :

$$\begin{array}{llllll} 1 := \mathbf{S}0, & 2 := \mathbf{S}1, & 3 := \mathbf{S}2, & 4 := \mathbf{S}3, & 5 := \mathbf{S}4, & \\ 6 := \mathbf{S}5, & 7 := \mathbf{S}6, & 8 := \mathbf{S}7, & 9 := \mathbf{S}8, & 10 := \mathbf{S}9, & \dots \end{array}$$

Heuristique

Les nombres naturels sont l'abstraction du processus de *compter* :

- Le nombre initial est noté 0.
- À chaque nombre naturel n on associe son successeur $\mathbf{S}n$.

Ceci permet déjà de nommer quelques petits exemples :

$$\begin{array}{cccccc} 1 := \mathbf{S}0, & 2 := \mathbf{S}1, & 3 := \mathbf{S}2, & 4 := \mathbf{S}3, & 5 := \mathbf{S}4, & \\ 6 := \mathbf{S}5, & 7 := \mathbf{S}6, & 8 := \mathbf{S}7, & 9 := \mathbf{S}8, & 10 := \mathbf{S}9, & \dots \end{array}$$

On sous-entend tacitement :

Heuristique

Les nombres naturels sont l'abstraction du processus de *compter* :

- Le nombre initial est noté 0.
- À chaque nombre naturel n on associe son successeur $\mathbf{S}n$.

Ceci permet déjà de nommer quelques petits exemples :

$$\begin{array}{cccccc} 1 := \mathbf{S}0, & 2 := \mathbf{S}1, & 3 := \mathbf{S}2, & 4 := \mathbf{S}3, & 5 := \mathbf{S}4, & \\ 6 := \mathbf{S}5, & 7 := \mathbf{S}6, & 8 := \mathbf{S}7, & 9 := \mathbf{S}8, & 10 := \mathbf{S}9, & \dots \end{array}$$

On sous-entend tacitement :

1. Le nombre 0 est le point de départ.

Heuristique

Les nombres naturels sont l'abstraction du processus de *compter* :

- Le nombre initial est noté 0.
- À chaque nombre naturel n on associe son successeur $\mathbf{S}n$.

Ceci permet déjà de nommer quelques petits exemples :

$$\begin{array}{llllll} 1 := \mathbf{S}0, & 2 := \mathbf{S}1, & 3 := \mathbf{S}2, & 4 := \mathbf{S}3, & 5 := \mathbf{S}4, & \\ 6 := \mathbf{S}5, & 7 := \mathbf{S}6, & 8 := \mathbf{S}7, & 9 := \mathbf{S}8, & 10 := \mathbf{S}9, & \dots \end{array}$$

On sous-entend tacitement :

1. Le nombre 0 est le point de départ.
2. Ce processus continue infiniment sans jamais boucler.

Les nombres naturels sont l'abstraction du processus de *compter* :

- Le nombre initial est noté 0.
- À chaque nombre naturel n on associe son successeur $\mathbf{S}n$.

Ceci permet déjà de nommer quelques petits exemples :

$$\begin{array}{cccccc} 1 := \mathbf{S}0, & 2 := \mathbf{S}1, & 3 := \mathbf{S}2, & 4 := \mathbf{S}3, & 5 := \mathbf{S}4, & \\ 6 := \mathbf{S}5, & 7 := \mathbf{S}6, & 8 := \mathbf{S}7, & 9 := \mathbf{S}8, & 10 := \mathbf{S}9, & \dots \end{array}$$

On sous-entend tacitement :

1. Le nombre 0 est le point de départ.
2. Ce processus continue infiniment sans jamais boucler.
3. Tout nombre naturel est ainsi atteint.

Heuristique

Les nombres naturels sont l'abstraction du processus de *compter* :

- Le nombre initial est noté 0.
- À chaque nombre naturel n on associe son successeur $\mathbf{S}n$.

Ceci permet déjà de nommer quelques petits exemples :

$$
\begin{array}{llllll}
1 := \mathbf{S}0, & 2 := \mathbf{S}1, & 3 := \mathbf{S}2, & 4 := \mathbf{S}3, & 5 := \mathbf{S}4, & \\
6 := \mathbf{S}5, & 7 := \mathbf{S}6, & 8 := \mathbf{S}7, & 9 := \mathbf{S}8, & 10 := \mathbf{S}9, & \dots
\end{array}
$$

On sous-entend tacitement :

1. Le nombre 0 est le point de départ.
2. Ce processus continue infiniment sans jamais boucler.
3. Tout nombre naturel est ainsi atteint.

Toute la subtilité est donc de préciser les trois petits points . . .

Définition (nombres naturels)

Les nombres naturels forment un ensemble $\mathbb{N}$ muni d'un élément initial $0 \in \mathbb{N}$ (zéro) et d'une application $\mathbf{S}\colon \mathbb{N} \to \mathbb{N}$ (successeur).

Définition (nombres naturels)

Les nombres naturels forment un ensemble $\mathbb{N}$ muni d'un élément initial $0 \in \mathbb{N}$ (zéro) et d'une application $\mathbf{S} \colon \mathbb{N} \to \mathbb{N}$ (successeur).

Ce triplet $(\mathbb{N}, 0, \mathbf{S})$ satisfait aux axiomes suivants :

Définition (nombres naturels)

Les nombres naturels forment un ensemble $\mathbb{N}$ muni d'un élément initial $0 \in \mathbb{N}$ (zéro) et d'une application $\mathbf{S} \colon \mathbb{N} \to \mathbb{N}$ (successeur).

Ce triplet $(\mathbb{N}, 0, \mathbf{S})$ satisfait aux axiomes suivants :

1. Pour tout n dans $\mathbb{N}$ on a $\mathbf{S}n \neq 0$.
   Autrement dit, 0 n'appartient pas à l'image de $\mathbf{S}$.

Définition (nombres naturels)

Les nombres naturels forment un ensemble $\mathbb{N}$ muni d'un élément initial $0 \in \mathbb{N}$ (zéro) et d'une application $\mathbf{S}\colon \mathbb{N} \to \mathbb{N}$ (successeur).

Ce triplet $(\mathbb{N}, 0, \mathbf{S})$ satisfait aux axiomes suivants :

1. Pour tout n dans $\mathbb{N}$ on a $\mathbf{S}n \neq 0$.
   Autrement dit, 0 n'appartient pas à l'image de $\mathbf{S}$.
2. Pour tout $n \neq m$ dans $\mathbb{N}$ on a $\mathbf{S}n \neq \mathbf{S}m$.
   Autrement dit, $\mathbf{S}$ est injectif.

Définition (nombres naturels)

Les nombres naturels forment un ensemble $\mathbb{N}$ muni d'un élément initial $0 \in \mathbb{N}$ (zéro) et d'une application $\mathbf{S}\colon \mathbb{N} \to \mathbb{N}$ (successeur).

Ce triplet $(\mathbb{N}, 0, \mathbf{S})$ satisfait aux axiomes suivants :

1. Pour tout n dans $\mathbb{N}$ on a $\mathbf{S}n \neq 0$.
   Autrement dit, 0 n'appartient pas à l'image de $\mathbf{S}$.
2. Pour tout $n \neq m$ dans $\mathbb{N}$ on a $\mathbf{S}n \neq \mathbf{S}m$.
   Autrement dit, $\mathbf{S}$ est injectif.
3. Si $E \subset \mathbb{N}$ vérifie $0 \in E$ et $\mathbf{S}(E) \subset E$, alors $E = \mathbb{N}$.
   C'est l'axiome de récurrence.

La construction par récurrence

Soient X un ensemble, $x_0 \in X$ un élément, $f\colon X \to X$ une fonction.

La construction par récurrence

Soient X un ensemble, $x_0 \in X$ un élément, $f\colon X \to X$ une fonction.

On souhaite construire la suite récurrente

$$x_0, \quad x_1 = f(x_0), \quad x_2 = f(x_1), \quad x_3 = f(x_2), \quad \ldots$$

Soient X un ensemble, $x_0 \in X$ un élément, $f\colon X \to X$ une fonction.

On souhaite construire la suite récurrente

$$x_0, \quad x_1 = f(x_0), \quad x_2 = f(x_1), \quad x_3 = f(x_2), \quad \ldots$$

On cherche donc à construire une application $\mathbb{N} \to X$ comme suit :

$$\begin{array}{c|ccccccccccccc}
\mathbb{N} & 0 & \stackrel{\mathbf{s}}{\mapsto} & 1 & \stackrel{\mathbf{s}}{\mapsto} & 2 & \stackrel{\mathbf{s}}{\mapsto} & 3 & \stackrel{\mathbf{s}}{\mapsto} & 4 & \stackrel{\mathbf{s}}{\mapsto} & 5 & \stackrel{\mathbf{s}}{\mapsto} & \ldots \\
 & \downarrow & & \downarrow & & \downarrow & & \downarrow & & \downarrow & & \downarrow & & \\
X & x_0 & \stackrel{f}{\mapsto} & x_1 & \stackrel{f}{\mapsto} & x_2 & \stackrel{f}{\mapsto} & x_3 & \stackrel{f}{\mapsto} & x_4 & \stackrel{f}{\mapsto} & x_5 & \stackrel{f}{\mapsto} & \ldots
\end{array}$$

La construction par récurrence

Soient X un ensemble, $x_0 \in X$ un élément, $f\colon X \to X$ une fonction.

On souhaite construire la suite récurrente

$$x_0, \quad x_1 = f(x_0), \quad x_2 = f(x_1), \quad x_3 = f(x_2), \quad \ldots$$

On cherche donc à construire une application $\mathbb{N} \to X$ comme suit :

$$\begin{array}{c|ccccccccccccc}
\mathbb{N} & 0 & \overset{\mathbf{S}}{\mapsto} & 1 & \overset{\mathbf{S}}{\mapsto} & 2 & \overset{\mathbf{S}}{\mapsto} & 3 & \overset{\mathbf{S}}{\mapsto} & 4 & \overset{\mathbf{S}}{\mapsto} & 5 & \overset{\mathbf{S}}{\mapsto} & \ldots \\
 & \downarrow & & \downarrow & & \downarrow & & \downarrow & & \downarrow & & \downarrow & & \\
X & x_0 & \overset{f}{\mapsto} & x_1 & \overset{f}{\mapsto} & x_2 & \overset{f}{\mapsto} & x_3 & \overset{f}{\mapsto} & x_4 & \overset{f}{\mapsto} & x_5 & \overset{f}{\mapsto} & \ldots
\end{array}$$

Une telle construction est toujours possible et même unique :

Théorème (Dedekind, 1888)

Soient X un ensemble, $x_0 \in X$ un élément, $f\colon X \to X$ une fonction. Alors il existe une unique fonction $g\colon \mathbb{N} \to X$ vérifiant $g(0) = x_0$ et $g \circ \mathbf{S} = f \circ g$, c'est-à-dire que $g(\mathbf{S}n) = f(g(n))$ pour tout $n \in \mathbb{N}$.

Définition (addition)

Il existe une unique application $+\colon \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ définie par $m + 0 := m$ puis par récurrence par $m + \mathbf{S}n := \mathbf{S}(m + n)$ pour tout $m, n \in \mathbb{N}$.

L’addition des nombres naturels

Définition (addition)

Il existe une unique application $+\colon \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ définie par $m + 0 := m$ puis par récurrence par $m + \mathbf{S}n := \mathbf{S}(m + n)$ pour tout $m, n \in \mathbb{N}$.

Remarquons que $n + 1 = n + \mathbf{S}0 = \mathbf{S}(n + 0) = \mathbf{S}n$.
Ceci permet de remplacer $\mathbf{S}$ par la notation $n \mapsto n + 1$.

Définition (addition)

Il existe une unique application $+ \colon \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ définie par $m + 0 := m$ puis par récurrence par $m + \mathbf{S}n := \mathbf{S}(m + n)$ pour tout $m, n \in \mathbb{N}$.

Remarquons que $n + 1 = n + \mathbf{S}0 = \mathbf{S}(n + 0) = \mathbf{S}n$.
Ceci permet de remplacer $\mathbf{S}$ par la notation $n \mapsto n + 1$.

Proposition

L'opération $+$, appelée addition, jouit des propriétés suivantes :

(A1 : associativité)	$\forall a, b, c \in \mathbb{N}$:	$(a + b) + c = a + (b + c)$
(A2 : commutativité)	$\forall a, b \in \mathbb{N}$:	$a + b = b + a$
(A3 : élément neutre)	$\forall a \in \mathbb{N}$:	$0 + a = a$

Définition (addition)

Il existe une unique application $+ \colon \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ définie par $m + 0 := m$ puis par récurrence par $m + \mathbf{S}n := \mathbf{S}(m + n)$ pour tout $m, n \in \mathbb{N}$.

Remarquons que $n + 1 = n + \mathbf{S}0 = \mathbf{S}(n + 0) = \mathbf{S}n$.
Ceci permet de remplacer $\mathbf{S}$ par la notation $n \mapsto n + 1$.

Proposition

L'opération $+$, appelée addition, jouit des propriétés suivantes :

(A1 : associativité)	$\forall a, b, c \in \mathbb{N}$:	$(a + b) + c = a + (b + c)$
(A2 : commutativité)	$\forall a, b \in \mathbb{N}$:	$a + b = b + a$
(A3 : élément neutre)	$\forall a \in \mathbb{N}$:	$0 + a = a$

L'addition est cancellative : $m + k = n + k$ implique $m = n$.

Définition (multiplication)

Il existe une unique application $\cdot\colon \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ définie par $m \cdot 0 := 0$ puis par récurrence $m \cdot \mathbf{S}n := m \cdot n + m$ pour tout $m, n \in \mathbb{N}$.

La multiplication des nombres naturels

Définition (multiplication)

Il existe une unique application $\cdot \colon \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ définie par $m \cdot 0 := 0$ puis par récurrence $m \cdot \mathbf{S}n := m \cdot n + m$ pour tout $m, n \in \mathbb{N}$.

Proposition

L'opération $\cdot$, appelée multiplication, jouit des propriétés suivantes :

(M1 : associativité)	$\forall a, b, c \in \mathbb{N}$:	$(a \cdot b) \cdot c = a \cdot (b \cdot c)$
(M2 : commutativité)	$\forall a, b \in \mathbb{N}$:	$a \cdot b = b \cdot a$
(M3 : élément neutre)	$\forall a \in \mathbb{N}$:	$1 \cdot a = a \cdot 1 = a$

La multiplication des nombres naturels

Définition (multiplication)

Il existe une unique application $\cdot\colon \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ définie par $m \cdot 0 := 0$ puis par récurrence $m \cdot \mathbf{S}n := m \cdot n + m$ pour tout $m, n \in \mathbb{N}$.

Proposition

L'opération $\cdot$, appelée multiplication, jouit des propriétés suivantes :

(M1 : associativité)	$\forall a, b, c \in \mathbb{N}$:	$(a \cdot b) \cdot c = a \cdot (b \cdot c)$
(M2 : commutativité)	$\forall a, b \in \mathbb{N}$:	$a \cdot b = b \cdot a$
(M3 : élément neutre)	$\forall a \in \mathbb{N}$:	$1 \cdot a = a \cdot 1 = a$

La multiplication est cancellative :
si $m \cdot k = n \cdot k$ et $k \neq 0$, alors $m = n$.

Définition (multiplication)

Il existe une unique application $\cdot : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ définie par $m \cdot 0 := 0$ puis par récurrence $m \cdot \mathbf{S}n := m \cdot n + m$ pour tout $m, n \in \mathbb{N}$.

Proposition

L'opération $\cdot$, appelée multiplication, jouit des propriétés suivantes :

(M1 : associativité)	$\forall a, b, c \in \mathbb{N}$:	$(a \cdot b) \cdot c = a \cdot (b \cdot c)$
(M2 : commutativité)	$\forall a, b \in \mathbb{N}$:	$a \cdot b = b \cdot a$
(M3 : élément neutre)	$\forall a \in \mathbb{N}$:	$1 \cdot a = a \cdot 1 = a$

La multiplication est cancellative :
si $m \cdot k = n \cdot k$ et $k \neq 0$, alors $m = n$.

La multiplication est distributive sur l'addition :
$k \cdot (m + n) = (k \cdot m) + (k \cdot n)$.

Définition (exponentiation)

Il existe une unique application $\hat{}\colon \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ définie par $a^0 := 1$ puis par récurrence $a^{\mathbf{S}n} := a^n \cdot a$ pour tout $a, n \in \mathbb{N}$.

Définition (exponentiation)

Il existe une unique application $\hat{}\colon \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ définie par $a^0 := 1$ puis par récurrence $a^{\mathbf{S}n} := a^n \cdot a$ pour tout $a, n \in \mathbb{N}$.

Proposition

L'opération $\hat{}$, appelée exponentiation, jouit des propriétés suivantes :

$$a^1 = a \quad \textit{et} \quad a^{m+n} = a^m \cdot a^n \quad \textit{et} \quad (a^m)^n = a^{m \cdot n}.$$

Définition (exponentiation)

Il existe une unique application $\hat{}\colon \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ définie par $a^0 := 1$ puis par récurrence $a^{\mathbf{S}n} := a^n \cdot a$ pour tout $a, n \in \mathbb{N}$.

Proposition

L'opération $\hat{}$, appelée exponentiation, jouit des propriétés suivantes :

$$a^1 = a \quad \textit{et} \quad a^{m+n} = a^m \cdot a^n \quad \textit{et} \quad (a^m)^n = a^{m \cdot n}.$$

Si $n \neq 0$, alors $a^n = b^n$ implique $a = b$.

Définition (exponentiation)

Il existe une unique application $\hat{}\colon \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ définie par $a^0 := 1$ puis par récurrence $a^{\mathbf{S}n} := a^n \cdot a$ pour tout $a, n \in \mathbb{N}$.

Proposition

L'opération $\hat{}$, appelée exponentiation, jouit des propriétés suivantes :

$$a^1 = a \quad \textit{et} \quad a^{m+n} = a^m \cdot a^n \quad \textit{et} \quad (a^m)^n = a^{m \cdot n}.$$

Si $n \neq 0$, alors $a^n = b^n$ implique $a = b$.

Si $a \notin \{0, 1\}$, alors $a^m = a^n$ implique $m = n$.

L'ordre des nombres naturels

Définition

On définit la relation $\leq$ sur $\mathbb{N}$ en posant $m \leq n$ si et seulement s'il existe $k \in \mathbb{N}$ tel que $m + k = n$.

Définition

On définit la relation $\leq$ sur $\mathbb{N}$ en posant $m \leq n$ si et seulement s'il existe $k \in \mathbb{N}$ tel que $m + k = n$.

Proposition

La relation $\leq$ est un ordre total sur $\mathbb{N}$.

Définition

On définit la relation $\leq$ sur $\mathbb{N}$ en posant $m \leq n$ si et seulement s'il existe $k \in \mathbb{N}$ tel que $m + k = n$.

Proposition

La relation $\leq$ est un ordre total sur $\mathbb{N}$.

C'est-à-dire que pour tout $a, b, c \in \mathbb{N}$ on a :

Définition

On définit la relation $\leq$ sur $\mathbb{N}$ en posant $m \leq n$ si et seulement s’il existe $k \in \mathbb{N}$ tel que $m + k = n$.

Proposition

La relation $\leq$ est un ordre total sur $\mathbb{N}$.

C’est-à-dire que pour tout $a, b, c \in \mathbb{N}$ on a :

- $a \leq a$ *(réflexivité)*

Définition

On définit la relation $\leq$ sur $\mathbb{N}$ en posant $m \leq n$ si et seulement s'il existe $k \in \mathbb{N}$ tel que $m + k = n$.

Proposition

La relation $\leq$ est un ordre total sur $\mathbb{N}$.

C'est-à-dire que pour tout $a, b, c \in \mathbb{N}$ on a :

- $a \leq a$ *(réflexivité)*
- $a \leq b$ *et* $b \leq a$ *impliquent* $a = b$ *(antisymétrie)*

L'ordre des nombres naturels

Définition

On définit la relation $\leq$ sur $\mathbb{N}$ en posant $m \leq n$ si et seulement s'il existe $k \in \mathbb{N}$ tel que $m + k = n$.

Proposition

La relation $\leq$ est un ordre total sur $\mathbb{N}$.

C'est-à-dire que pour tout $a, b, c \in \mathbb{N}$ on a :

- $a \leq a$ *(réflexivité)*
- $a \leq b$ *et* $b \leq a$ *impliquent* $a = b$ *(antisymétrie)*
- $a \leq b$ *et* $b \leq c$ *impliquent* $a \leq c$ *(transitivité)*

L'ordre des nombres naturels

Définition

On définit la relation $\leq$ sur $\mathbb{N}$ en posant $m \leq n$ si et seulement s'il existe $k \in \mathbb{N}$ tel que $m + k = n$.

Proposition

La relation $\leq$ est un ordre total sur $\mathbb{N}$.

C'est-à-dire que pour tout $a, b, c \in \mathbb{N}$ on a :

- $a \leq a$ *(réflexivité)*
- $a \leq b$ *et* $b \leq a$ *impliquent* $a = b$ *(antisymétrie)*
- $a \leq b$ *et* $b \leq c$ *impliquent* $a \leq c$ *(transitivité)*
- $a \leq b$ *ou* $b \leq a$ *(totalité)*

Définition

On définit la relation $\leq$ sur $\mathbb{N}$ en posant $m \leq n$ si et seulement s'il existe $k \in \mathbb{N}$ tel que $m + k = n$.

Proposition

La relation $\leq$ est un ordre total sur $\mathbb{N}$.

C'est-à-dire que pour tout $a, b, c \in \mathbb{N}$ on a :

- $a \leq a$ *(réflexivité)*
- $a \leq b$ *et* $b \leq a$ *impliquent* $a = b$ *(antisymétrie)*
- $a \leq b$ *et* $b \leq c$ *impliquent* $a \leq c$ *(transitivité)*
- $a \leq b$ *ou* $b \leq a$ *(totalité)*

Proposition

L'ensemble $\mathbb{N}$ muni de sa relation d'ordre $\leq$ est bien ordonné :

L’ordre des nombres naturels

Définition

On définit la relation $\leq$ sur $\mathbb{N}$ en posant $m \leq n$ si et seulement s’il existe $k \in \mathbb{N}$ tel que $m + k = n$.

Proposition

La relation $\leq$ est un ordre total sur $\mathbb{N}$.

C’est-à-dire que pour tout $a, b, c \in \mathbb{N}$ on a :

- $a \leq a$ *(réflexivité)*
- $a \leq b$ *et* $b \leq a$ *impliquent* $a = b$ *(antisymétrie)*
- $a \leq b$ *et* $b \leq c$ *impliquent* $a \leq c$ *(transitivité)*
- $a \leq b$ *ou* $b \leq a$ *(totalité)*

Proposition

L’ensemble $\mathbb{N}$ muni de sa relation d’ordre $\leq$ est bien ordonné :
Tout sous-ensemble non vide $X \subset \mathbb{N}$ admet un plus petit élément, c’est-à-dire qu’il existe $m \in X$ tel que $m \leq x$ pour tout $x \in X$.

Les opérations respectent l'ordre

Proposition

L'addition et la multiplication respectent l'ordre sur $\mathbb{N}$ *:*

Les opérations respectent l'ordre

Proposition

L'addition et la multiplication respectent l'ordre sur $\mathbb{N}$ *:*

- $m \leq n$ *implique* $m + k \leq n + k$.

Les opérations respectent l'ordre

Proposition

L'addition et la multiplication respectent l'ordre sur $\mathbb{N}$ *:*

- $m \leq n$ *implique* $m + k \leq n + k$.
- $m \leq n$ *implique* $mk \leq nk$.

Les opérations respectent l'ordre

Proposition

L'addition et la multiplication respectent l'ordre sur $\mathbb{N}$ *:*

- $m \leq n$ *implique* $m + k \leq n + k$.
- $m \leq n$ *implique* $mk \leq nk$.

Il en est de même pour l'inégalité stricte :

Proposition

L'addition et la multiplication respectent l'ordre sur $\mathbb{N}$:

- $m \leq n$ *implique* $m + k \leq n + k$.
- $m \leq n$ *implique* $mk \leq nk$.

Il en est de même pour l'inégalité stricte :

- $m < n$ *implique* $m + k < n + k$.

Les opérations respectent l'ordre

Proposition

L'addition et la multiplication respectent l'ordre sur $\mathbb{N}$ *:*

- $m \leq n$ *implique* $m + k \leq n + k$*.*
- $m \leq n$ *implique* $mk \leq nk$*.*

Il en est de même pour l'inégalité stricte :

- $m < n$ *implique* $m + k < n + k$*.*
- $m < n$ *implique* $mk < nk$ *pourvu que* $k \neq 0$*.*

Les opérations respectent l'ordre

Proposition

L'addition et la multiplication respectent l'ordre sur $\mathbb{N}$ *:*

- $m \leq n$ *implique* $m + k \leq n + k$.
- $m \leq n$ *implique* $mk \leq nk$.

Il en est de même pour l'inégalité stricte :

- $m < n$ *implique* $m + k < n + k$.
- $m < n$ *implique* $mk < nk$ *pourvu que* $k \neq 0$.

Pour l'exponentiation on obtient :

Les opérations respectent l'ordre

Proposition

L'addition et la multiplication respectent l'ordre sur $\mathbb{N}$ *:*

- $m \leq n$ *implique* $m + k \leq n + k$.
- $m \leq n$ *implique* $mk \leq nk$.

Il en est de même pour l'inégalité stricte :

- $m < n$ *implique* $m + k < n + k$.
- $m < n$ *implique* $mk < nk$ *pourvu que* $k \neq 0$.

Pour l'exponentiation on obtient :

- *Si* $m < n$ *et* $k \neq 0$*, alors* $m^k < n^k$.

Les opérations respectent l'ordre

Proposition

L'addition et la multiplication respectent l'ordre sur $\mathbb{N}$ *:*

- $m \leq n$ *implique* $m + k \leq n + k$.
- $m \leq n$ *implique* $mk \leq nk$.

Il en est de même pour l'inégalité stricte :

- $m < n$ *implique* $m + k < n + k$.
- $m < n$ *implique* $mk < nk$ *pourvu que* $k \neq 0$.

Pour l'exponentiation on obtient :

- *Si* $m < n$ *et* $k \neq 0$*, alors* $m^k < n^k$.
- *Si* $m < n$ *et* $a \notin \{0, 1\}$*, alors* $a^m < a^n$.

Les opérations respectent l’ordre

Proposition

L’addition et la multiplication respectent l’ordre sur $\mathbb{N}$ *:*

- $m \leq n$ *implique* $m + k \leq n + k$*.*
- $m \leq n$ *implique* $mk \leq nk$*.*

Il en est de même pour l’inégalité stricte :

- $m < n$ *implique* $m + k < n + k$*.*
- $m < n$ *implique* $mk < nk$ *pourvu que* $k \neq 0$*.*

Pour l’exponentiation on obtient :

- *Si* $m < n$ *et* $k \neq 0$*, alors* $m^k < n^k$*.*
- *Si* $m < n$ *et* $a \notin \{0, 1\}$*, alors* $a^m < a^n$*.*

Définition (soustraction)

Dans $\mathbb{N}$ la soustraction $n - m$ n’est définie que si $n \geq m$.

Les opérations respectent l'ordre

Proposition

L'addition et la multiplication respectent l'ordre sur $\mathbb{N}$ *:*

- $m \leq n$ *implique* $m + k \leq n + k$.
- $m \leq n$ *implique* $mk \leq nk$.

Il en est de même pour l'inégalité stricte :

- $m < n$ *implique* $m + k < n + k$.
- $m < n$ *implique* $mk < nk$ *pourvu que* $k \neq 0$.

Pour l'exponentiation on obtient :

- *Si* $m < n$ *et* $k \neq 0$, *alors* $m^k < n^k$.
- *Si* $m < n$ *et* $a \notin \{0, 1\}$, *alors* $a^m < a^n$.

Définition (soustraction)

Dans $\mathbb{N}$ la soustraction $n - m$ n'est définie que si $n \geq m$.
Dans ce cas il existe $k \in \mathbb{N}$ tel que $m + k = n$.

Les opérations respectent l’ordre

Proposition

L’addition et la multiplication respectent l’ordre sur $\mathbb{N}$:

- $m \leq n$ *implique* $m + k \leq n + k$.
- $m \leq n$ *implique* $mk \leq nk$.

Il en est de même pour l’inégalité stricte :

- $m < n$ *implique* $m + k < n + k$.
- $m < n$ *implique* $mk < nk$ *pourvu que* $k \neq 0$.

Pour l’exponentiation on obtient :

- *Si* $m < n$ *et* $k \neq 0$*, alors* $m^k < n^k$.
- *Si* $m < n$ *et* $a \notin \{0, 1\}$*, alors* $a^m < a^n$.

Définition (soustraction)

Dans $\mathbb{N}$ la soustraction $n - m$ n’est définie que si $n \geq m$.
Dans ce cas il existe $k \in \mathbb{N}$ tel que $m + k = n$.
Ce nombre est unique, on peut donc définir $n - m := k$.

Définition (divisibilité)

On définit la relation $\mid$ sur $\mathbb{N}$ en posant $m \mid n$ (dit « m divise n ») si et seulement s'il existe $k \in \mathbb{N}$ tel que $mk = n$.

Définition (divisibilité)

On définit la relation $\mid$ sur $\mathbb{N}$ en posant $m \mid n$ (dit « m divise n »)
si et seulement s'il existe $k \in \mathbb{N}$ tel que $mk = n$.

Proposition

La relation $\mid$ est un ordre partiel sur $\mathbb{N}$.

Définition (divisibilité)

On définit la relation $\mid$ sur $\mathbb{N}$ en posant $m \mid n$ (dit « m divise n »)
si et seulement s'il existe $k \in \mathbb{N}$ tel que $mk = n$.

Proposition

La relation $\mid$ est un ordre partiel sur $\mathbb{N}$.

C'est-à-dire que pour tout $a, b, c \in \mathbb{N}$ on a :

Définition (divisibilité)

On définit la relation $\mid$ sur $\mathbb{N}$ en posant $m \mid n$ (dit « m divise n »)
si et seulement s'il existe $k \in \mathbb{N}$ tel que $mk = n$.

Proposition

La relation $\mid$ est un ordre partiel sur $\mathbb{N}$.

C'est-à-dire que pour tout $a, b, c \in \mathbb{N}$ on a :

- $a \mid a$ *(réflexivité)*

Définition (divisibilité)

On définit la relation $\mid$ sur $\mathbb{N}$ en posant $m \mid n$ (dit « m divise n »)
si et seulement s'il existe $k \in \mathbb{N}$ tel que $mk = n$.

Proposition

La relation $\mid$ est un ordre partiel sur $\mathbb{N}$.

C'est-à-dire que pour tout $a, b, c \in \mathbb{N}$ on a :

- $a \mid a$ *(réflexivité)*
- $a \mid b$ *et* $b \mid a$ *impliquent* $a = b$ *(antisymétrie)*

Définition (divisibilité)

On définit la relation $|$ sur $\mathbb{N}$ en posant $m \mid n$ (dit « m divise n ») si et seulement s'il existe $k \in \mathbb{N}$ tel que $mk = n$.

Proposition

La relation $|$ est un ordre partiel sur $\mathbb{N}$.

C'est-à-dire que pour tout $a, b, c \in \mathbb{N}$ on a :

- $a \mid a$ *(réflexivité)*
- $a \mid b$ *et* $b \mid a$ *impliquent* $a = b$ *(antisymétrie)*
- $a \mid b$ *et* $b \mid c$ *impliquent* $a \mid c$ *(transitivité)*

Définition (divisibilité)

On définit la relation $\mid$ sur $\mathbb{N}$ en posant $m \mid n$ (dit « m divise n ») si et seulement s'il existe $k \in \mathbb{N}$ tel que $mk = n$.

Proposition

La relation $\mid$ est un ordre partiel sur $\mathbb{N}$.

C'est-à-dire que pour tout $a, b, c \in \mathbb{N}$ on a :

- $a \mid a$ *(réflexivité)*
- $a \mid b$ *et* $b \mid a$ *impliquent* $a = b$ *(antisymétrie)*
- $a \mid b$ *et* $b \mid c$ *impliquent* $a \mid c$ *(transitivité)*

Définition (division)

Dans $\mathbb{N}$ la division a/b n'est définie que si $b \mid a$.

Divisibilité et division

Définition (divisibilité)

On définit la relation $|$ sur $\mathbb{N}$ en posant $m \mid n$ (dit « m divise n »)
si et seulement s'il existe $k \in \mathbb{N}$ tel que $mk = n$.

Proposition

La relation $|$ est un ordre partiel sur $\mathbb{N}$.

C'est-à-dire que pour tout $a, b, c \in \mathbb{N}$ on a :

- $a \mid a$ *(réflexivité)*
- $a \mid b$ *et* $b \mid a$ *impliquent* $a = b$ *(antisymétrie)*
- $a \mid b$ *et* $b \mid c$ *impliquent* $a \mid c$ *(transitivité)*

Définition (division)

Dans $\mathbb{N}$ la division a/b n'est définie que si $b \mid a$.
Dans ce cas il existe $q \in \mathbb{N}$ tel que $bq = a$.

Divisibilité et division

Définition (divisibilité)

On définit la relation $|$ sur $\mathbb{N}$ en posant $m \mid n$ (dit « m divise n »)
si et seulement s'il existe $k \in \mathbb{N}$ tel que $mk = n$.

Proposition

La relation $|$ est un ordre partiel sur $\mathbb{N}$.

C'est-à-dire que pour tout $a, b, c \in \mathbb{N}$ on a :

- $a \mid a$ *(réflexivité)*
- $a \mid b$ *et* $b \mid a$ *impliquent* $a = b$ *(antisymétrie)*
- $a \mid b$ *et* $b \mid c$ *impliquent* $a \mid c$ *(transitivité)*

Définition (division)

Dans $\mathbb{N}$ la division a/b n'est définie que si $b \mid a$.
Dans ce cas il existe $q \in \mathbb{N}$ tel que $bq = a$.
À l'exception du cas $a = b = 0$ le nombre q est unique.

Définition (divisibilité)

On définit la relation $\mid$ sur $\mathbb{N}$ en posant $m \mid n$ (dit « m divise n »)
si et seulement s'il existe $k \in \mathbb{N}$ tel que $mk = n$.

Proposition

La relation $\mid$ est un ordre partiel sur $\mathbb{N}$.

C'est-à-dire que pour tout $a, b, c \in \mathbb{N}$ on a :

- $a \mid a$ *(réflexivité)*
- $a \mid b$ *et* $b \mid a$ *impliquent* $a = b$ *(antisymétrie)*
- $a \mid b$ *et* $b \mid c$ *impliquent* $a \mid c$ *(transitivité)*

Définition (division)

Dans $\mathbb{N}$ la division a/b n'est définie que si $b \mid a$.
Dans ce cas il existe $q \in \mathbb{N}$ tel que $bq = a$.
À l'exception du cas $a = b = 0$ le nombre q est unique.
Sous cette précaution on peut donc définir $a/b := q$.

Proposition (division euclidienne)

Soient $a, b \in \mathbb{N}$ *deux nombres naturels tels que* $b \neq 0$*.*

Proposition (division euclidienne)

Soient $a, b \in \mathbb{N}$ *deux nombres naturels tels que* $b \neq 0$.
Alors il existe une unique paire $(q, r) \in \mathbb{N} \times \mathbb{N}$
vérifiant $a = b \cdot q + r$ *et* $0 \leq r < b$.

Division euclidienne

Proposition (division euclidienne)

Soient $a, b \in \mathbb{N}$ *deux nombres naturels tels que* $b \neq 0$.
Alors il existe une unique paire $(q, r) \in \mathbb{N} \times \mathbb{N}$
vérifiant $a = b \cdot q + r$ *et* $0 \leq r < b$.

Notation

Dans la situation précédente on appelle

- $a \operatorname{quo} b := q$ le *quotient* et

Proposition (division euclidienne)

Soient $a, b \in \mathbb{N}$ *deux nombres naturels tels que* $b \neq 0$.
Alors il existe une unique paire $(q, r) \in \mathbb{N} \times \mathbb{N}$
vérifiant $a = b \cdot q + r$ *et* $0 \leq r < b$.

Notation

Dans la situation précédente on appelle

- $a \operatorname{quo} b := q$ le *quotient* et
- $a \operatorname{rem} b := r$ le *reste*

de la division euclidienne de a par b.

Division euclidienne

Proposition (division euclidienne)

Soient $a, b \in \mathbb{N}$ *deux nombres naturels tels que* $b \neq 0$*.*
Alors il existe une unique paire $(q, r) \in \mathbb{N} \times \mathbb{N}$
vérifiant $a = b \cdot q + r$ *et* $0 \leq r < b$*.*

Notation

Dans la situation précédente on appelle

- $a \operatorname{quo} b := q$ le *quotient* et
- $a \operatorname{rem} b := r$ le *reste*

de la division euclidienne de a par b.

Remarque

Si $b \mid a$ alors $a = bc$ pour un certain c noté $c = a/b$.

Proposition (division euclidienne)

Soient $a, b \in \mathbb{N}$ *deux nombres naturels tels que* $b \neq 0$*.*
Alors il existe une unique paire $(q, r) \in \mathbb{N} \times \mathbb{N}$
vérifiant $a = b \cdot q + r$ *et* $0 \leq r < b$*.*

Notation

Dans la situation précédente on appelle

- $a \operatorname{quo} b := q$ le *quotient* et
- $a \operatorname{rem} b := r$ le *reste*

de la division euclidienne de a par b.

Remarque

Si $b \mid a$ alors $a = bc$ pour un certain c noté $c = a/b$.
La division euclidienne nous donne $a = bq + r$.

Division euclidienne

Proposition (division euclidienne)

Soient $a, b \in \mathbb{N}$ *deux nombres naturels tels que* $b \neq 0$*.*
Alors il existe une unique paire $(q, r) \in \mathbb{N} \times \mathbb{N}$
vérifiant $a = b \cdot q + r$ *et* $0 \leq r < b$*.*

Notation

Dans la situation précédente on appelle

- $a \operatorname{quo} b := q$ le *quotient* et
- $a \operatorname{rem} b := r$ le *reste*

de la division euclidienne de a par b.

Remarque

Si $b \mid a$ alors $a = bc$ pour un certain c noté $c = a/b$.
La division euclidienne nous donne $a = bq + r$.
L'unicité assure que $q = a/b$ et $r = 0$.

Numération positionnelle

On fixe un nombre naturel $B \geq 2$ que l'on appelle *base* dans la suite.

Numération positionnelle

On fixe un nombre naturel $B \geq 2$ que l'on appelle *base* dans la suite.

Théorème (numération en base B)

Alors tout nombre naturel $a \geq 1$ s'écrit de manière unique comme

$$a = a_n B^n + a_{n-1} B^{n-1} + \cdots + a_1 B + a_0$$

Numération positionnelle

On fixe un nombre naturel $B \geq 2$ que l'on appelle *base* dans la suite.

Théorème (numération en base B)

Alors tout nombre naturel $a \geq 1$ s'écrit de manière unique comme

$$a = a_n B^n + a_{n-1} B^{n-1} + \cdots + a_1 B + a_0$$

où $a_0, a_1, \ldots, a_{n-1}, a_n$ sont des nombres naturels vérifiant $0 \leq a_k < B$ pour tout $k = 0, \ldots, n$ ainsi que $a_n \neq 0$ (normalisation).

Numération positionnelle

On fixe un nombre naturel $B \geq 2$ que l'on appelle *base* dans la suite.

Théorème (numération en base B)

Alors tout nombre naturel $a \geq 1$ *s'écrit de manière unique comme*

$$a = a_n B^n + a_{n-1} B^{n-1} + \cdots + a_1 B + a_0$$

où $a_0, a_1, \ldots, a_{n-1}, a_n$ *sont des nombres naturels vérifiant* $0 \leq a_k < B$ *pour tout* $k = 0, \ldots, n$ *ainsi que* $a_n \neq 0$ *(normalisation).*

Idée de la preuve : on applique la division euclidienne.

Numération positionnelle

On fixe un nombre naturel $B \geq 2$ que l'on appelle *base* dans la suite.

Théorème (numération en base B)

Alors tout nombre naturel $a \geq 1$ *s'écrit de manière unique comme*

$$a = a_n B^n + a_{n-1} B^{n-1} + \cdots + a_1 B + a_0$$

où $a_0, a_1, \ldots, a_{n-1}, a_n$ *sont des nombres naturels vérifiant* $0 \leq a_k < B$ *pour tout* $k = 0, \ldots, n$ *ainsi que* $a_n \neq 0$ *(normalisation).*

Idée de la preuve : on applique la division euclidienne.

$$a = \underbrace{\left(a_n B^{n-1} + a_{n-1} B^{n-2} + \cdots + a_1 B^0\right)}_{a \operatorname{quo} B} B + \underbrace{a_0}_{a \operatorname{rem} B}$$

Numération positionnelle

On fixe un nombre naturel $B \geq 2$ que l'on appelle *base* dans la suite.

Théorème (numération en base B)

Alors tout nombre naturel $a \geq 1$ s'écrit de manière unique comme

$$a = a_n B^n + a_{n-1} B^{n-1} + \cdots + a_1 B + a_0$$

où $a_0, a_1, \ldots, a_{n-1}, a_n$ sont des nombres naturels vérifiant $0 \leq a_k < B$ pour tout $k = 0, \ldots, n$ ainsi que $a_n \neq 0$ (normalisation).

Idée de la preuve : on applique la division euclidienne.

$$a = \underbrace{(a_n B^{n-1} + a_{n-1} B^{n-2} + \cdots + a_1 B^0)}_{a \operatorname{quo} B} B + \underbrace{a_0}_{a \operatorname{rem} B}$$

Ceci permet de trouver $a_0 = a \operatorname{rem} B$. On conclut par récurrence.

De manière abrégée on écrit aussi

$$\langle a_n, a_{n-1}, \ldots, a_1, a_0 \rangle_B \ := \ a_n B^n + a_{n-1} B^{n-1} + \cdots + a_1 B + a_0.$$

Numération positionnelle : notation

De manière abrégée on écrit aussi

$$\langle a_n, a_{n-1}, \ldots, a_1, a_0 \rangle_B \; := \; a_n B^n + a_{n-1} B^{n-1} + \cdots + a_1 B + a_0.$$

Définition

On appelle $\mathbf{a} = (a_0, a_1, \ldots, a_n)$ la *représentation* de a en base B.

Numération positionnelle : notation

De manière abrégée on écrit aussi

$$\langle a_n, a_{n-1}, \ldots, a_1, a_0 \rangle_B \;:=\; a_n B^n + a_{n-1} B^{n-1} + \cdots + a_1 B + a_0.$$

Définition

On appelle $\mathbf{a} = (a_0, a_1, \ldots, a_n)$ la *représentation* de a en base B.

On appelle $\operatorname{len}_B(a) := n + 1$ la *longueur* de a en base B.

Numération positionnelle : notation

De manière abrégée on écrit aussi

$$\langle a_n, a_{n-1}, \dots, a_1, a_0 \rangle_B \; := \; a_n B^n + a_{n-1} B^{n-1} + \cdots + a_1 B + a_0.$$

Définition

On appelle $\mathbf{a} = (a_0, a_1, \dots, a_n)$ la *représentation* de a en base B.

On appelle $\operatorname{len}_B(a) := n + 1$ la *longueur* de a en base B.

Lorsque B est assez petit, on peut représenter chaque nombre a_i vérifiant $0 \leq a_i < B$ par un symbole distinctif, appelé *chiffre*.

Numération positionnelle : notation

De manière abrégée on écrit aussi

$$\langle a_n, a_{n-1}, \dots, a_1, a_0 \rangle_B \; := \; a_n B^n + a_{n-1} B^{n-1} + \cdots + a_1 B + a_0.$$

Définition

On appelle $\mathbf{a} = (a_0, a_1, \dots, a_n)$ la *représentation* de a en base B.

On appelle $\operatorname{len}_B(a) := n + 1$ la *longueur* de a en base B.

Lorsque B est assez petit, on peut représenter chaque nombre a_i vérifiant $0 \le a_i < B$ par un symbole distinctif, appelé *chiffre*.

En base fixée (disons $B = 10$) ceci mène à une notation comme

$$1729 := \langle 1, 7, 2, 9 \rangle_{\text{dec}} := 1 \cdot 10^3 + 7 \cdot 10^2 + 2 \cdot 10^1 + 9 \cdot 10^0.$$

C'est simple et efficace, mais il fallait y penser !

Sommaire

Représentation des nombres naturels

Pour une implémentation concrète sur ordinateur il nous faut

- une représentation des données et
- une implémentation des opérations.

Pour une implémentation concrète sur ordinateur il nous faut

- une représentation des données et
- une implémentation des opérations.

Question préparatoire : comment faire ?

Représentation des nombres naturels

Pour une implémentation concrète sur ordinateur il nous faut

- une représentation des données et
- une implémentation des opérations.

Question préparatoire : comment faire ?

Ici on représente $a = \sum_{i=0}^{m} a_i B^i$ par la suite $\mathbf{a} = (a_0, \ldots, a_m)$.

Représentation des nombres naturels

Pour une implémentation concrète sur ordinateur il nous faut

- une représentation des données et
- une implémentation des opérations.

Question préparatoire : comment faire ?

Ici on représente $a = \sum_{i=0}^{m} a_i B^i$ par la suite $\mathbf{a} = (a_0, \dots, a_m)$.

⚠ On exigera que cette représentation $\mathbf{a}$ soit normalisée :

$$0 \leq a_i < B \quad \text{et} \quad a_n \neq 0.$$

Représentation des nombres naturels

Pour une implémentation concrète sur ordinateur il nous faut

- une représentation des données et
- une implémentation des opérations.

Question préparatoire : comment faire ?

Ici on représente $a = \sum_{i=0}^{m} a_i B^i$ par la suite $\mathbf{a} = (a_0, \dots, a_m)$.

⚠ On exigera que cette représentation $\mathbf{a}$ soit normalisée :

$$0 \leq a_i < B \quad \text{et} \quad a_n \neq 0.$$

⚠ Si une opération risque de finir avec une suite non normalisée on effectue une *normalisation* pour propager les retenues.

Définition de la classe Naturel

```
typedef int Indice;           // type pour stocker un indice
typedef short int Chiffre;    // type pour stocker un chiffre (ou deux)
const Chiffre base= 10;       // base entre 2 et 16, ici on choisit 10
const char chiffre[]= "0123456789abcdef"; // chiffres pour l'affichage

class Naturel
{
public:
  vector<Chiffre> chiffres;   // suite des chiffres dans la base donnée
  Indice size() const { return chiffres.size(); };
  bool zero() const { return chiffres.empty(); };
  void clear() { chiffres.clear(); };
  void raccourcir();
  void normaliser();
};
```

Stockage des chiffres

Supprimer des zéros superflus :

```
void Naturel::raccourcir()
{
  while ( !chiffres.empty() && chiffres.back() == Chiffre(0) )
    chiffres.pop_back();
};
```

Supprimer des zéros superflus :

```
void Naturel::raccourcir()
{
  while ( !chiffres.empty() && chiffres.back() == Chiffre(0) )
    chiffres.pop_back();
};
```

Accéder au chiffre a_i par `a[i]` via une indexation sécurisée :

```
Chiffre Naturel::operator[] ( Indice i ) const
{
  if ( i<0 || i>=Indice(chiffres.size()) )
    return Chiffre(0);
  else
    return chiffres[i];
};
```

Propager les retenues

```
void Naturel::normaliser()
{
  Chiffre retenue= 0;
  for( Indice i=0; i<Indice(chiffres.size()); ++i )
  {
    chiffres[i]+= retenue;
    retenue     = chiffres[i]/base;
    chiffres[i]%= base;
  };

  while( retenue > 0 )
  {
    chiffres.push_back( retenue % base );
    retenue/= base;
  };

  raccourcir();
};
```

Incrémentation

L'opération la plus simple est l'incrémentation $a \mapsto a + 1$.

Incrémentation

L'opération la plus simple est l'incrémentation $a \mapsto a + 1$.

Voici une implémentation correcte mais peu efficace :

```
bool incrementer( Naturel& a )
{
  if ( a.zero() )
  {
    a.chiffres.push_back( Chiffre(1) );
  }
  else
  {
    ++(a.chiffres[0]);
    a.normaliser();
  }
  return true;
}
```

Voici une implémentation plus efficace :

Incrémentation

Voici une implémentation plus efficace :

```
bool incrementer( Naturel& a )
{
  for( Indice i=0; i<a.size(); ++i )
    if ( a.chiffres[i] == Chiffre(base-1) )
    {
      a.chiffres[i]= Chiffre(0);
    }
    else
    {
      ++(a.chiffres[i]);
      return true;
    };
  a.chiffres.push_back( Chiffre(1) );
  return true;
}
```

Decrémentation

La decrémentation $a \mapsto a - 1$ pour $a > 0$ est analogue :

Decrémentation

La decrémentation $a \mapsto a - 1$ pour $a > 0$ est analogue :

```
bool decrementer( Naturel& a )
{
  for( Indice i=0; i<a.size(); ++i )
    if ( a.chiffres[i] == Chiffre(0) )
    {
      a.chiffres[i]= Chiffre(base-1);
    }
    else
    {
      --(a.chiffres[i]);
      a.raccourcir();
      return true;
    };
  return false;
}
```

Addition « avec les doigts »

L'addition suivante traduit directement la définition axiomatique :

```
bool addition_lente( const Naturel& a, Naturel b, Naturel& somme )
{
  somme= a;
  while( !b.zero() )
  {
    incrementer(somme);
    decrementer(b);
  };
  return true;
}
```

Addition « avec les doigts »

L'addition suivante traduit directement la définition axiomatique :

```
bool addition_lente( const Naturel& a, Naturel b, Naturel& somme )
{
  somme= a;
  while( !b.zero() )
  {
    incrementer(somme);
    decrementer(b);
  };
  return true;
}
```

⚠ Si c'est encore acceptable pour les petits mombres, cette méthode est prohibitive pour des calculs sérieux.

Multiplication « sur les doigts »

La multiplication suivante traduit la définition axiomatique :

```
bool multiplication_lente( const Naturel& a, Naturel b, Naturel& produit )
{
  produit= 0;
  while( !b.zero() )
  {
    produit= produit + a,
    decrementer(b);
  }
  return true;
};
```

Multiplication « sur les doigts »

La multiplication suivante traduit la définition axiomatique :

```
bool multiplication_lente( const Naturel& a, Naturel b, Naturel& produit )
{
  produit= 0;
  while( !b.zero() )
  {
    produit= produit + a,
    decrementer(b);
  }
  return true;
};
```

⚠ Si c'est encore acceptable pour les petits mombres,
cette méthode est prohibitive pour des calculs sérieux.

Addition et multiplication plus efficaces

On représente $a = \sum_{i=0}^{m} a_i B^i$ par la suite $\mathbf{a} = (a_0, \dots, a_m)$.

Addition et multiplication plus efficaces

On représente $a = \sum_{i=0}^{m} a_i B^i$ par la suite $\mathbf{a} = (a_0, \ldots, a_m)$.
De même on représente $b = \sum_{j=0}^{n} b_j B^j$ par $\mathbf{b} = (b_0, \ldots, b_n)$.

Addition et multiplication plus efficaces

On représente $a = \sum_{i=0}^{m} a_i B^i$ par la suite $\mathbf{a} = (a_0, \ldots, a_m)$.
De même on représente $b = \sum_{j=0}^{n} b_j B^j$ par $\mathbf{b} = (b_0, \ldots, b_n)$.

Leur somme $s = a + b$ est donnée par

$$s = \sum_{k} s_k B^k \quad \text{où} \quad s_i = a_i + b_i.$$

Addition et multiplication plus efficaces

On représente $a = \sum_{i=0}^{m} a_i B^i$ par la suite $\mathbf{a} = (a_0, \ldots, a_m)$.
De même on représente $b = \sum_{j=0}^{n} b_j B^j$ par $\mathbf{b} = (b_0, \ldots, b_n)$.

Leur somme $s = a + b$ est donnée par

$$s = \sum_k s_k B^k \quad \text{où} \quad s_i = a_i + b_i.$$

Leur produit $p = a \cdot b$ est donné par

$$p = \sum_k p_k B^k \quad \text{où} \quad p_k = \sum_{i+j=k} a_i b_j$$

Addition et multiplication plus efficaces

On représente $a = \sum_{i=0}^{m} a_i B^i$ par la suite $\mathbf{a} = (a_0, \dots, a_m)$.
De même on représente $b = \sum_{j=0}^{n} b_j B^j$ par $\mathbf{b} = (b_0, \dots, b_n)$.

Leur somme $s = a + b$ est donnée par

$$s = \sum_k s_k B^k \quad \text{où} \quad s_i = a_i + b_i.$$

Leur produit $p = a \cdot b$ est donné par

$$p = \sum_k p_k B^k \quad \text{où} \quad p_k = \sum_{i+j=k} a_i b_j$$

Ces formules découlent des propriétés établies ci-dessus pour l'addition et la multiplication. (Les maths, ça sert.)

Addition et multiplication plus efficaces

On représente $a = \sum_{i=0}^{m} a_i B^i$ par la suite $\mathbf{a} = (a_0, \dots, a_m)$.
De même on représente $b = \sum_{j=0}^{n} b_j B^j$ par $\mathbf{b} = (b_0, \dots, b_n)$.

Leur somme $s = a + b$ est donnée par

$$s = \sum_k s_k B^k \quad \text{où} \quad s_i = a_i + b_i.$$

Leur produit $p = a \cdot b$ est donné par

$$p = \sum_k p_k B^k \quad \text{où} \quad p_k = \sum_{i+j=k} a_i b_j$$

Ces formules découlent des propriétés établies ci-dessus pour l'addition et la multiplication. (Les maths, ça sert.)

⚠ On exige que les représentations $\mathbf{a}$ et $\mathbf{b}$ soient normalisées :

$$0 \le a_i < B \quad \text{et} \quad a_n \neq 0.$$

Addition et multiplication plus efficaces

On représente $a = \sum_{i=0}^{m} a_i B^i$ par la suite $\mathbf{a} = (a_0, \dots, a_m)$.
De même on représente $b = \sum_{j=0}^{n} b_j B^j$ par $\mathbf{b} = (b_0, \dots, b_n)$.

Leur somme $s = a + b$ est donnée par

$$s = \sum_k s_k B^k \quad \text{où} \quad s_i = a_i + b_i.$$

Leur produit $p = a \cdot b$ est donné par

$$p = \sum_k p_k B^k \quad \text{où} \quad p_k = \sum_{i+j=k} a_i b_j$$

Ces formules découlent des propriétés établies ci-dessus
pour l'addition et la multiplication. (Les maths, ça sert.)

⚠ On exige que les représentations $\mathbf{a}$ et $\mathbf{b}$ soient normalisées :

$$0 \le a_i < B \quad \text{et} \quad a_n \neq 0.$$

Or, les suites $\mathbf{s} = (s_0, s_1, \dots)$ et $\mathbf{p} = (p_0, p_1, \dots)$ ne le seront plus.

Addition et multiplication plus efficaces

On représente $a = \sum_{i=0}^{m} a_i B^i$ par la suite $\mathbf{a} = (a_0, \dots, a_m)$.
De même on représente $b = \sum_{j=0}^{n} b_j B^j$ par $\mathbf{b} = (b_0, \dots, b_n)$.

Leur somme $s = a + b$ est donnée par

$$s = \sum_k s_k B^k \quad \text{où} \quad s_i = a_i + b_i.$$

Leur produit $p = a \cdot b$ est donné par

$$p = \sum_k p_k B^k \quad \text{où} \quad p_k = \sum_{i+j=k} a_i b_j$$

Ces formules découlent des propriétés établies ci-dessus
pour l'addition et la multiplication. (Les maths, ça sert.)

⚠ On exige que les représentations $\mathbf{a}$ et $\mathbf{b}$ soient normalisées :

$$0 \le a_i < B \quad \text{et} \quad a_n \neq 0.$$

Or, les suites $\mathbf{s} = (s_0, s_1, \dots)$ et $\mathbf{p} = (p_0, p_1, \dots)$ ne le seront plus.
On effectue donc une *normalisation* pour propager les retenues.

Addition scolaire

```
bool addition( const Naturel& a, const Naturel& b, Naturel& somme )
{
  // Réserver de la mémoire pour recevoir le résultat
  somme.clear();
  Indice taille= max( a.size(), b.size() );
  somme.chiffres.resize( taille, Chiffre(0) );

  // Calculer la somme chiffre par chiffre en tenant compte des retenues
  Chiffre retenue= 0;
  for( Indice i=0; i<taille; ++i )
    {
      Chiffre temp= a[i] + b[i] + retenue;   // indexation sécurisée
      if ( temp < base ) { somme.chiffres[i]= temp; retenue = 0; }
      else { somme.chiffres[i]= temp - base; retenue = 1; };
    };

  // Rajouter éventuellement un chiffre supplémentaire pour la retenue
  if ( retenue ) somme.chiffres.push_back(retenue);
  return true;
};
```

Multiplication scolaire

```
bool multiplication( const Naturel& a, const Naturel& b, Naturel& produit )
{
  // Attraper le cas particulier puis réserver de la mémoire
  produit.clear();
  if ( a.zero() || b.zero() ) return true;
  produit.chiffres.resize( a.size() + b.size(), Chiffre(0) );

  // Calculer le produit par (une variante de) la méthode scolaire
  for( Indice i=0; i<a.size(); ++i )
  {
    Chiffre aux, retenue= 0;
    for( Indice j=0; j<b.size(); ++j )
    {
      aux= produit.chiffres[i+j] + a.chiffres[i] * b.chiffres[j] + retenue;
      produit.chiffres[i+j]= aux % base;
      retenue= aux / base;
    };
    produit.chiffres[i+b.size()]= retenue;
  };
  produit.raccourcir();
  return true;
}
```

Comparaison

On considère $a, b \in \mathbb{N}$ représentés en base B par

$$\mathbf{a} = (a_0, a_1, \ldots, a_m) \qquad \text{tel que } a = \sum_{i=0}^{m} a_i B^i,\ 0 \leq a_i < B,\ a_m \neq 0$$

$$\mathbf{b} = (b_0, b_1, \ldots, b_m) \qquad \text{tel que } b = \sum_{j=0}^{n} b_j B^j,\ 0 \leq b_j < B,\ b_n \neq 0$$

Exercice

Expliciter un algorithme pour déterminer si $a < b$ ou $a = b$ ou $a > b$.
Prouver sa correction en utilisant les résultats précédents.
Esquisser une implémentation (en C++ disons).
Est-ce efficace ? Peut-on faire mieux ?

Implémentation de la comparaison

```
int comparaison_lente( Naturel a, Naturel b )
{
  while( !a.zero() && !b.zero() ) { decrementer(a); decrementer(b); }
  if ( a.zero() && b.zero() ) return 0;
  if ( a.zero() ) return -1; else return +1;
}
```

Implémentation de la comparaison

```
int comparaison_lente( Naturel a, Naturel b )
{
  while( !a.zero() && !b.zero() ) { decrementer(a); decrementer(b); }
  if ( a.zero() && b.zero() ) return 0;
  if ( a.zero() ) return -1; else return +1;
}
```

```
int comparaison_scolaire( const Naturel& a, const Naturel& b )
{
  if ( a.size() > b.size() ) return +1; // a > b
  if ( a.size() < b.size() ) return -1; // a < b
  for( Indice i= a.size()-1; i>=0; --i )
    {
      if ( a.chiffres[i] > b.chiffres[i] ) return +1; // a > b
      if ( a.chiffres[i] < b.chiffres[i] ) return -1; // a < b
    };
  return 0; // a == b
}
```

Opérations sur $a, b \in \mathbb{N}$ de longueur $\mathrm{len}(a), \mathrm{len}(b) \leq n$.

Opération	temps	mémoire
comparaison $a = b$	$O(n)$	$O(1)$
comparaison $a \leq b$	$O(n)$	$O(1)$

Opérations sur $a, b \in \mathbb{N}$ de longueur $\operatorname{len}(a), \operatorname{len}(b) \leq n$.

Opération	temps	mémoire
comparaison $a = b$	$O(n)$	$O(1)$
comparaison $a \leq b$	$O(n)$	$O(1)$
incrémentation $a + 1$	$O(n)$	$O(n)$
décrémentation $a - 1$	$O(n)$	$O(n)$

Opérations sur $a, b \in \mathbb{N}$ de longueur $\operatorname{len}(a), \operatorname{len}(b) \leq n$.

Opération	temps	mémoire
comparaison $a = b$	$O(n)$	$O(1)$
comparaison $a \leq b$	$O(n)$	$O(1)$
incrémentation $a + 1$	$O(n)$	$O(n)$
décrémentation $a - 1$	$O(n)$	$O(n)$
addition $a + b$	$O(n)$	$O(n)$
soustraction $a - b$	$O(n)$	$O(n)$

Opérations sur $a, b \in \mathbb{N}$ de longueur $\operatorname{len}(a), \operatorname{len}(b) \leq n$.

Opération	temps	mémoire
comparaison $a = b$	$O(n)$	$O(1)$
comparaison $a \leq b$	$O(n)$	$O(1)$
incrémentation $a + 1$	$O(n)$	$O(n)$
décrémentation $a - 1$	$O(n)$	$O(n)$
addition $a + b$	$O(n)$	$O(n)$
soustraction $a - b$	$O(n)$	$O(n)$
multiplication $a \cdot b$	$O(n^2)$	$O(n)$
division euclidienne $a \operatorname{quo} b$, $a \operatorname{rem} b$	$O(n^2)$	$O(n)$

Opérations sur $a, b \in \mathbb{N}$ de longueur $\operatorname{len}(a), \operatorname{len}(b) \leq n$.

Opération	temps	mémoire
comparaison $a = b$	$O(n)$	$O(1)$
comparaison $a \leq b$	$O(n)$	$O(1)$
incrémentation $a + 1$	$O(n)$	$O(n)$
décrémentation $a - 1$	$O(n)$	$O(n)$
addition $a + b$	$O(n)$	$O(n)$
soustraction $a - b$	$O(n)$	$O(n)$
multiplication $a \cdot b$	$O(n^2)$	$O(n)$
division euclidienne $a \operatorname{quo} b$, $a \operatorname{rem} b$	$O(n^2)$	$O(n)$

Dans nos algorithmes ces bornes sont génériquement atteintes.

Sommaire

La notion de semi-anneau

Les nombres naturels $(\mathbb{N}, +, \cdot)$ forment un *semi-anneau* :

La notion de semi-anneau

Les nombres naturels $(\mathbb{N}, +, \cdot)$ forment un *semi-anneau* :

Définition (semi-anneau)

Un *semi-anneau* $(A, +, \cdot)$ est un ensemble A muni d'une addition $+\colon A \times A \to A$ et d'une multiplication $\cdot\colon A \times A \to A$ satisfaisant aux axiomes suivants :

(A1 : associativité)	$\forall a, b, c :$	$(a + b) + c = a + (b + c)$
(A2 : commutativité)	$\forall a, b :$	$a + b = b + a$
(A3 : élément neutre)	$\exists 0 \ \forall a :$	$0 + a = a$
(M1 : associativité)	$\forall a, b, c :$	$(a \cdot b) \cdot c = a \cdot (b \cdot c)$
(M2 : commutativité)	$\forall a, b :$	$a \cdot b = b \cdot a$
(M3 : élément neutre)	$\exists 1 \neq 0 \ \forall a :$	$1 \cdot a = a$
(D : distributivité)	$\forall a, b, c :$	$a \cdot (b + c) = (a \cdot b) + (a \cdot c)$

La notion de semi-anneau

Les nombres naturels $(\mathbb{N}, +, \cdot)$ forment un *semi-anneau* :

Définition (semi-anneau)

Un *semi-anneau* $(A, +, \cdot)$ est un ensemble A muni d'une addition $+ \colon A \times A \to A$ et d'une multiplication $\cdot \colon A \times A \to A$ satisfaisant aux axiomes suivants :

(A1 : associativité)	$\forall a, b, c :$	$(a + b) + c = a + (b + c)$
(A2 : commutativité)	$\forall a, b :$	$a + b = b + a$
(A3 : élément neutre)	$\exists 0 \ \forall a :$	$0 + a = a$
(M1 : associativité)	$\forall a, b, c :$	$(a \cdot b) \cdot c = a \cdot (b \cdot c)$
(M2 : commutativité)	$\forall a, b :$	$a \cdot b = b \cdot a$
(M3 : élément neutre)	$\exists 1 \neq 0 \ \forall a :$	$1 \cdot a = a$
(D : distributivité)	$\forall a, b, c :$	$a \cdot (b + c) = (a \cdot b) + (a \cdot c)$

Dans $\mathbb{N}$ certaines équations comme $5 + x = 0$ n'ont pas de solution.

La notion d’anneau

On souhaite « compléter » $\mathbb{N}$ par des éléments nouveaux, que l’on appellera « nombres négatifs », afin de pouvoir résoudre toute équation $a = b + x$ avec $a, b \in \mathbb{N}$.

On souhaite « compléter » $\mathbb{N}$ par des éléments nouveaux, que l'on appellera « nombres négatifs », afin de pouvoir résoudre toute équation $a = b + x$ avec $a, b \in \mathbb{N}$.

Définition (anneau)

Un *anneau* est un ensemble A muni d'une addition $+\colon A \times A \to A$ et d'une multiplication $\cdot\colon A \times A \to A$ satisfaisant aux axiomes suivants :

(A1 : associativité)	$\forall a, b, c :$	$(a + b) + c = a + (b + c)$
(A2 : commutativité)	$\forall a, b :$	$a + b = b + a$
(A3 : élément neutre)	$\exists 0\ \forall a :$	$0 + a = a$
(A4 : élément opposé)	$\forall a\ \exists b :$	$a + b = 0$
(M1 : associativité)	$\forall a, b, c :$	$(a \cdot b) \cdot c = a \cdot (b \cdot c)$
(M2 : commutativité)	$\forall a, b :$	$a \cdot b = b \cdot a$
(M3 : élément neutre)	$\exists 1 \neq 0\ \forall a :$	$1 \cdot a = a$
(D : distributivité)	$\forall a, b, c :$	$a \cdot (b + c) = (a \cdot b) + (a \cdot c)$

Définition (nombres entiers)

Les nombres entiers $(\mathbb{Z}, +, \cdot)$ sont l'unique anneau
contenant le semi-anneau des nombres naturels $(\mathbb{N}, +, \cdot)$
de sorte que tout $a \in \mathbb{Z}$ vérifie $a \in \mathbb{N}$ ou $-a \in \mathbb{N}$.

Extension des nombres naturels aux nombres entiers

Définition (nombres entiers)

Les nombres entiers $(\mathbb{Z}, +, \cdot)$ sont l'unique anneau
contenant le semi-anneau des nombres naturels $(\mathbb{N}, +, \cdot)$
de sorte que tout $a \in \mathbb{Z}$ vérifie $a \in \mathbb{N}$ ou $-a \in \mathbb{N}$.

Questions naturelles :

- Est-ce qu'un tel anneau existe ?
- Pourquoi un tel anneau est-il unique ?

Définition (nombres entiers)

Les nombres entiers $(\mathbb{Z}, +, \cdot)$ sont l'unique anneau contenant le semi-anneau des nombres naturels $(\mathbb{N}, +, \cdot)$ de sorte que tout $a \in \mathbb{Z}$ vérifie $a \in \mathbb{N}$ ou $-a \in \mathbb{N}$.

Questions naturelles :

- Est-ce qu'un tel anneau existe ?
- Pourquoi un tel anneau est-il unique ?

Remarque

On peut *construire* les nombres entiers à partir des nombres naturels, puis *vérifier* les propriétés exigées. (Exercice long mais bénéfique)

L'ordre sur $\mathbb{Z}$

Définition

On étend l'ordre de $\mathbb{N}$ à $\mathbb{Z}$ en posant $a \leq b$ si et seulement s'il existe $k \in \mathbb{N}$ tel que $a + k = b$. Ainsi $\mathbb{N} = \{a \in \mathbb{Z} \mid a \geq 0\}$.

L'ordre sur $\mathbb{Z}$

Définition

On étend l'ordre de $\mathbb{N}$ à $\mathbb{Z}$ en posant $a \leq b$ si et seulement s'il existe $k \in \mathbb{N}$ tel que $a + k = b$. Ainsi $\mathbb{N} = \{a \in \mathbb{Z} \mid a \geq 0\}$.

Proposition

Cet ordre sur $\mathbb{Z}$ est compatible avec l'addition et la multiplication :

(OA) $\forall a, b, c :$ $\quad a \leq b \Rightarrow a + c \leq b + c$

(OM) $\forall a, b, c :$ $\quad a \leq b, 0 \leq c \Rightarrow ac \leq bc$

L'ordre sur $\mathbb{Z}$

Définition

On étend l'ordre de $\mathbb{N}$ à $\mathbb{Z}$ en posant $a \leq b$ si et seulement s'il existe $k \in \mathbb{N}$ tel que $a + k = b$. Ainsi $\mathbb{N} = \{a \in \mathbb{Z} \mid a \geq 0\}$.

Proposition

Cet ordre sur $\mathbb{Z}$ est compatible avec l'addition et la multiplication :

(OA) $\forall a, b, c :$ $$a \leq b \Rightarrow a + c \leq b + c$$

(OM) $\forall a, b, c :$ $$a \leq b, 0 \leq c \Rightarrow ac \leq bc$$

Notation

Pour tout $a \in \mathbb{Z}$ on pose $|a| = a$ si $a \geq 0$, et $|a| = -a$ si $a < 0$.

La division euclidienne dans $\mathbb{Z}$

Proposition (division euclidienne)

Soient $a, b \in \mathbb{Z}$ deux nombres entiers tels que $b \neq 0$.

Proposition (division euclidienne)

Soient $a, b \in \mathbb{Z}$ *deux nombres entiers tels que* $b \neq 0$*.*
Alors il existe une unique paire $(q, r) \in \mathbb{Z} \times \mathbb{Z}$
vérifiant $a = b \cdot q + r$ *et* $0 \leq r < |b|$*.*

Proposition (division euclidienne)

Soient $a, b \in \mathbb{Z}$ *deux nombres entiers tels que* $b \neq 0$*.*
Alors il existe une unique paire $(q, r) \in \mathbb{Z} \times \mathbb{Z}$
vérifiant $a = b \cdot q + r$ *et* $0 \leq r < |b|$*.*

Notation

Dans la situation précédente on appelle

- $a \operatorname{quo} b := q$ le *quotient* et

Proposition (division euclidienne)

Soient $a, b \in \mathbb{Z}$ *deux nombres entiers tels que* $b \neq 0$*.*
Alors il existe une unique paire $(q, r) \in \mathbb{Z} \times \mathbb{Z}$
vérifiant $a = b \cdot q + r$ *et* $0 \leq r < |b|$*.*

Notation

Dans la situation précédente on appelle

- $a \operatorname{quo} b := q$ le *quotient* et
- $a \operatorname{rem} b := r$ le *reste*

de la division euclidienne de a par b.

La division euclidienne dans $\mathbb{Z}$

Proposition (division euclidienne)

Soient $a, b \in \mathbb{Z}$ *deux nombres entiers tels que* $b \neq 0$*.*
Alors il existe une unique paire $(q, r) \in \mathbb{Z} \times \mathbb{Z}$
vérifiant $a = b \cdot q + r$ *et* $0 \leq r < |b|$*.*

Notation

Dans la situation précédente on appelle

- $a \operatorname{quo} b := q$ le *quotient* et
- $a \operatorname{rem} b := r$ le *reste*

de la division euclidienne de a par b.

Dans certains langages de programmation ceci s'écrit `a/b` et `a%b`.

Proposition (division euclidienne)

Soient $a, b \in \mathbb{Z}$ *deux nombres entiers tels que* $b \neq 0$*.*
Alors il existe une unique paire $(q, r) \in \mathbb{Z} \times \mathbb{Z}$
vérifiant $a = b \cdot q + r$ *et* $0 \leq r < |b|$*.*

Notation

Dans la situation précédente on appelle

- $a \operatorname{quo} b := q$ le *quotient* et
- $a \operatorname{rem} b := r$ le *reste*

de la division euclidienne de a par b.

Dans certains langages de programmation ceci s'écrit `a/b` et `a%b`.

⚠ Pour les nombres négatifs les opérations `a/b` et `a%b` en C/C++ et en Java n'implémentent pas la convention ci-dessus :

Proposition (division euclidienne)

Soient $a, b \in \mathbb{Z}$ *deux nombres entiers tels que* $b \neq 0$*.*
Alors il existe une unique paire $(q, r) \in \mathbb{Z} \times \mathbb{Z}$
vérifiant $a = b \cdot q + r$ *et* $0 \leq r < |b|$*.*

Notation

Dans la situation précédente on appelle

- $a \operatorname{quo} b := q$ le *quotient* et
- $a \operatorname{rem} b := r$ le *reste*

de la division euclidienne de a par b.

Dans certains langages de programmation ceci s'écrit `a/b` et `a%b`.

⚠ Pour les nombres négatifs les opérations `a/b` et `a%b` en C/C++ et en Java n'implémentent pas la convention ci-dessus :
`16%9` donne `7`, mais `(-16)%9` donne `-7`.

Sommaire

Opérations sur $a, b \in \mathbb{Z}$ de longueur $\operatorname{len}|a|, \operatorname{len}|b| \leq n$.

Opération	temps	mémoire
comparaison $a = b$	$O(n)$	$O(1)$
comparaison $a \leq b$	$O(n)$	$O(1)$
incrémentation $a + 1$	$O(n)$	$O(n)$
decrémentation $a - 1$	$O(n)$	$O(n)$
addition $a + b$	$O(n)$	$O(n)$
soustraction $a - b$	$O(n)$	$O(n)$
multiplication $a \cdot b$	$O(n^2)$	$O(n)$
division euclidienne $a \operatorname{quo} b$, $a \operatorname{rem} b$	$O(n^2)$	$O(n)$

Complexité des algorithmes scolaires

Opérations sur $a, b \in \mathbb{Z}$ de longueur $\operatorname{len}|a|, \operatorname{len}|b| \leq n$.

Opération	temps	mémoire
comparaison $a = b$	$O(n)$	$O(1)$
comparaison $a \leq b$	$O(n)$	$O(1)$
incrémentation $a + 1$	$O(n)$	$O(n)$
decrémentation $a - 1$	$O(n)$	$O(n)$
addition $a + b$	$O(n)$	$O(n)$
soustraction $a - b$	$O(n)$	$O(n)$
multiplication $a \cdot b$	$O(n^2)$	$O(n)$
division euclidienne $a \operatorname{quo} b$, $a \operatorname{rem} b$	$O(n^2)$	$O(n)$

Peut-on calculer plus rapidement ? de manière significative ?

Multiplication rapide selon Karatsuba

On suppose donnés deux nombres naturels a et b en base B.

Multiplication rapide selon Karatsuba

On suppose donnés deux nombres naturels a et b en base B.
Chacun est de longueur $< 2n$, autrement dit, $0 \leq a, b < B^{2n}$.

Multiplication rapide selon Karatsuba

On suppose donnés deux nombres naturels a et b en base B.

Chacun est de longueur $< 2n$, autrement dit, $0 \leq a, b < B^{2n}$.

On les décompose comme

$$a = a_0 + a_1 B^n \qquad \text{et} \qquad b = b_0 + b_1 B^n.$$

On suppose donnés deux nombres naturels a et b en base B.

Chacun est de longueur $< 2n$, autrement dit, $0 \leq a, b < B^{2n}$.

On les décompose comme

$$a = a_0 + a_1 B^n \qquad \text{et} \qquad b = b_0 + b_1 B^n.$$

Le produit ab est égal à

$$ab = (a_0 b_0) + (a_0 b_1 + a_1 b_0) B^n + (a_1 b_1) B^{2n}.$$

Multiplication rapide selon Karatsuba

On suppose donnés deux nombres naturels a et b en base B.

Chacun est de longueur $< 2n$, autrement dit, $0 \leq a, b < B^{2n}$.

On les décompose comme

$$a = a_0 + a_1 B^n \qquad \text{et} \qquad b = b_0 + b_1 B^n.$$

Le produit ab est égal à

$$ab = (a_0 b_0) + (a_0 b_1 + a_1 b_0) B^n + (a_1 b_1) B^{2n}.$$

A priori ceci nécessite quatre multiplications de type $n \times n$.

On suppose donnés deux nombres naturels a et b en base B.

Chacun est de longueur $< 2n$, autrement dit, $0 \leq a, b < B^{2n}$.

On les décompose comme

$$a = a_0 + a_1 B^n \qquad \text{et} \qquad b = b_0 + b_1 B^n.$$

Le produit ab est égal à

$$ab = (a_0 b_0) + (a_0 b_1 + a_1 b_0) B^n + (a_1 b_1) B^{2n}.$$

A priori ceci nécessite quatre multiplications de type $n \times n$.

Or, on peut calculer le résultat avec trois multiplications seulement !

$$s \leftarrow a_0 b_0, \quad t \leftarrow a_1 b_1, \quad u \leftarrow (a_0 + a_1)(b_0 + b_1) - s - t.$$

Multiplication rapide selon Karatsuba

On suppose donnés deux nombres naturels a et b en base B.

Chacun est de longueur $< 2n$, autrement dit, $0 \leq a, b < B^{2n}$.

On les décompose comme

$$a = a_0 + a_1 B^n \qquad \text{et} \qquad b = b_0 + b_1 B^n.$$

Le produit ab est égal à

$$ab = (a_0 b_0) + (a_0 b_1 + a_1 b_0) B^n + (a_1 b_1) B^{2n}.$$

A priori ceci nécessite quatre multiplications de type $n \times n$.

Or, on peut calculer le résultat avec trois multiplications seulement !

$$s \leftarrow a_0 b_0, \quad t \leftarrow a_1 b_1, \quad u \leftarrow (a_0 + a_1)(b_0 + b_1) - s - t.$$

On constate que $u = a_0 b_1 + a_1 b_0$, donc le produit cherché est

$$ab = s + uB^n + tB^{2n}$$

Analyse de complexité

Soit $c(n)$ le coût de la multiplication selon Karatsuba,

Analyse de complexité

Soit $c(n)$ le coût de la multiplication selon Karatsuba, mesuré en nombre d'opérations sur les chiffres.

Analyse de complexité

Soit $c(n)$ le coût de la multiplication selon Karatsuba, mesuré en nombre d'opérations sur les chiffres.

Alors la fonction $c\colon \mathbb{N} \to \mathbb{N}$ vérifie la majoration

$$c(2n) \leq 3c(n) + \alpha n.$$

Analyse de complexité

Soit $c(n)$ le coût de la multiplication selon Karatsuba, mesuré en nombre d'opérations sur les chiffres.

Alors la fonction $c\colon \mathbb{N} \to \mathbb{N}$ vérifie la majoration

$$c(2n) \leq 3c(n) + \alpha n.$$

Ici $3c(n)$ est le coût des 3 multiplications de taille n, puis αn est le coût linéaire des additions/soustractions.

Analyse de complexité

Soit $c(n)$ le coût de la multiplication selon Karatsuba, mesuré en nombre d'opérations sur les chiffres.

Alors la fonction $c\colon \mathbb{N} \to \mathbb{N}$ vérifie la majoration

$$c(2n) \leq 3c(n) + \alpha n.$$

Ici $3c(n)$ est le coût des 3 multiplications de taille n, puis αn est le coût linéaire des additions/soustractions.

On suppose une majoration $c(1) \leq \beta$ pour le cas de base.

Analyse de complexité

Soit $c(n)$ le coût de la multiplication selon Karatsuba,
mesuré en nombre d'opérations sur les chiffres.

Alors la fonction $c \colon \mathbb{N} \to \mathbb{N}$ vérifie la majoration

$$c(2n) \leq 3c(n) + \alpha n.$$

Ici $3c(n)$ est le coût des 3 multiplications de taille n,
puis αn est le coût linéaire des additions/soustractions.

On suppose une majoration $c(1) \leq \beta$ pour le cas de base.

Proposition

On a $c(n) < 3(\alpha + \beta)n^{\varepsilon}$ *avec exposant* $\varepsilon = \log 3 / \log 2 \approx 1,585$.

Soit $c(n)$ le coût de la multiplication selon Karatsuba, mesuré en nombre d'opérations sur les chiffres.

Alors la fonction $c\colon \mathbb{N} \to \mathbb{N}$ vérifie la majoration

$$c(2n) \le 3c(n) + \alpha n.$$

Ici $3c(n)$ est le coût des 3 multiplications de taille n, puis αn est le coût linéaire des additions/soustractions.

On suppose une majoration $c(1) \le \beta$ pour le cas de base.

Proposition

On a $c(n) < 3(\alpha + \beta)n^{\varepsilon}$ avec exposant $\varepsilon = \log 3 / \log 2 \approx 1,585$.

Les constantes α et β dépendent des détails de l'implémentation.

Analyse de complexité

Soit $c(n)$ le coût de la multiplication selon Karatsuba, mesuré en nombre d'opérations sur les chiffres.

Alors la fonction $c\colon \mathbb{N} \to \mathbb{N}$ vérifie la majoration

$$c(2n) \le 3c(n) + \alpha n.$$

Ici $3c(n)$ est le coût des 3 multiplications de taille n, puis αn est le coût linéaire des additions/soustractions.

On suppose une majoration $c(1) \le \beta$ pour le cas de base.

Proposition

On a $c(n) < 3(\alpha + \beta) n^{\varepsilon}$ *avec exposant* $\varepsilon = \log 3 / \log 2 \approx 1,585$.

Les constantes α et β dépendent des détails de l'implémentation.

Quelques soient les détails, pour n grand la méthode de Karatsuba est une amélioration significative de la méthode quadratique.

Multiplication de $a, b \in \mathbb{Z}$ de longueur $\operatorname{len}|a|, \operatorname{len}|b| \leq n$.
(La complexité de la division euclidienne est la même.)

Multiplication de $a, b \in \mathbb{Z}$ de longueur $\operatorname{len}|a|, \operatorname{len}|b| \leq n$.
(La complexité de la division euclidienne est la même.)

Méthode	connue depuis	coût en temps
lente	préhistoire	$O(B^n)$

Multiplication de $a, b \in \mathbb{Z}$ de longueur $\operatorname{len}|a|, \operatorname{len}|b| \leq n$.
(La complexité de la division euclidienne est la même.)

Méthode	connue depuis	coût en temps
lente	préhistoire	$O(B^n)$
scolaire	moyen âge	$O(n^2)$

Multiplication de $a, b \in \mathbb{Z}$ de longueur $\operatorname{len}|a|, \operatorname{len}|b| \leq n$.
(La complexité de la division euclidienne est la même.)

Méthode	connue depuis	coût en temps
lente	préhistoire	$O(B^n)$
scolaire	moyen âge	$O(n^2)$
Karatsuba	1962	$O(n^{1,585})$

Multiplication de $a, b \in \mathbb{Z}$ de longueur $\operatorname{len}|a|, \operatorname{len}|b| \leq n$.
(La complexité de la division euclidienne est la même.)

Méthode	connue depuis	coût en temps
lente	préhistoire	$O(B^n)$
scolaire	moyen âge	$O(n^2)$
Karatsuba	1962	$O(n^{1,585})$
Toom-Cook	1963	$O(n^{1,465})$
...	...	

Complexité de la multiplication

Multiplication de $a, b \in \mathbb{Z}$ de longueur $\operatorname{len}|a|, \operatorname{len}|b| \leq n$.
(La complexité de la division euclidienne est la même.)

Méthode	connue depuis	coût en temps
lente	préhistoire	$O(B^n)$
scolaire	moyen âge	$O(n^2)$
Karatsuba	1962	$O(n^{1,585})$
Toom-Cook	1963	$O(n^{1,465})$
...	...	
Schönhage-Strassen FFT	1971	$O(n \cdot \log n \cdot \log \log n)$

Complexité de la multiplication

Multiplication de $a, b \in \mathbb{Z}$ de longueur $\operatorname{len}|a|, \operatorname{len}|b| \leq n$.
(La complexité de la division euclidienne est la même.)

Méthode	connue depuis	coût en temps
lente	préhistoire	$O(B^n)$
scolaire	moyen âge	$O(n^2)$
Karatsuba	1962	$O(n^{1,585})$
Toom-Cook	1963	$O(n^{1,465})$
...	...	
Schönhage-Strassen FFT	1971	$O(n \cdot \log n \cdot \log \log n)$

« The development of fast algorithms is slow ! »

Bibliothèques courantes

Pour les calculs sérieux il faut des algorithmes optimisés.

Bibliothèques courantes

Pour les calculs sérieux il faut des algorithmes optimisés.
Or, leur implémentation est longue est coûteuse,

Bibliothèques courantes

Pour les calculs sérieux il faut des algorithmes optimisés.
Or, leur implémentation est longue est coûteuse,
Donc, les bibliothèques c'est bon, mangez-en !

Bibliothèques courantes

Pour les calculs sérieux il faut des algorithmes optimisés.
Or, leur implémentation est longue est coûteuse,
Donc, les bibliothèques c'est bon, mangez-en !

Bibliothèques arithmétique courantes :

- GNU Multiple Precision Project (GMP), `http://gmplib.org/`
- Number Theory Library (NTL), `http://www.shoup.net/ntl/`