

Autres algorithmes simples (partie 2)

Sébastien Jean

IUT de Valence
Département Informatique

v1.0, 29 septembre 2025

Nombre parfait

Enoncé du problème

Un entier n est parfait s'il est égal à la somme de ses diviseurs propres (c'est-à-dire tous les diviseurs de n strictement inférieurs à n).

Spécification du problème

- Donnée d'entrée : n , entier (le nombre à tester)
- Donnée de sortie : *parfait*, booléen (caractère parfait de n)
- Pré-condition : $n \geq 0$
- Post-condition : *parfait* est VRAI si n est égal à la somme de ses diviseurs propres inférieurs stricts à n , FAUX sinon

Signature de la fonction

- *est_parfait* (n : entier) : booléen

Nombre parfait

Corps de la fonction



Nombre parfait

FONCTION est_parfait (n : entier) : booléen

VARIABLE i : entier

VARIABLE somme : entier

somme $\leftarrow$ 0

SI n = 0 ALORS

RETOURNER FAUX

FIN SI

POUR i DE 1 A n-1 PAR PAS DE 1

SI n mod i = 0 ALORS

somme $\leftarrow$ somme + i

FIN SI

FIN POUR

RETOURNER somme = n

FIN FONCTION

Nombre inversé

Enoncé du problème

On cherche un entier i dont les chiffres sont dans l'ordre inverse d'un entier n (ex : $123 \rightarrow 321$).

Spécification du problème

- Donnée d'entrée : n , entier (le nombre à inverser)
- Donnée de sortie : $inverse$, entier (le nombre inversé)
- Pré-condition : $n \geq 0$
- Post-condition : $inverse$ est égal à l'entier obtenu en inversant les chiffres de n .

Signature de la fonction

- $inverse(n : \text{entier}) : \text{entier}$

Nombre inversé

Corps de la fonction



Nombre inversé

```
FONCTION inverse (n : entier) : entier
```

```
VARIABLE inverse : entier
```

```
VARIABLE copie : entier
```

```
inverse ← 0
```

```
copie ← n
```

```
TANT QUE copie > 0
```

```
    inverse ← inverse * 10 + copie mod 10
```

```
    copie ← copie div 10
```

```
FIN TANT QUE
```

```
RETOURNER inverse
```

```
FIN FONCTION
```

Fin !

