DiU EIL - Algorithmique

Bruno Darid

1er août 2019

1 Terminaison des algorithmes

Recherche séquentielle - ex.4

Algorithme 1 : Ex.4 Recherche séquentielle

Données : $L[0 \ldots n-1]$ tableau de n entiers, n_0 : un entier
Résultat : un booléen

$r : booléen \leftarrow Faux$
$n : entier \leftarrow len(L)$
$dernier : entier \leftarrow L[n-1]$
$i : entier \leftarrow 0$
$L[n-1] \leftarrow n_0$ // Placement d'une valeur sentinelle dans le tableau
tant que $L[i] \neq n_0$ **faire**
  $i \leftarrow i+1$
$L[n-1] \leftarrow dernier$
si $i < n-1$ **alors**
  $r \leftarrow Vrai$ // n_0 présent avant la fin de tableau
sinon si $L[n-1] = n_0$ **alors**
  $r \leftarrow Vrai$ // n_0 présent en dernière position.
retourner r

Somme double - ex. 5

On donne la somme double $\sum_{1 \leq i,j \leq n} min(i,j)$. On présente un algorithme naïf correspondant à cette somme (*qui se calcule aisément*).

$\sum\limits_{1\leq i,j\leq n} min(i,j) = \sum\limits_{i=1}^{n}\left(\sum\limits_{j=1}^{n} min(i,j)\right)$. Or, $min(i,j) = \begin{cases} j & \text{pour } 1 \leq j \leq i \\ i & \text{pour } i+1 \leq j \leq n \end{cases}$

$$
\begin{aligned}
\sum_{1\leq i,j\leq n} min(i,j) &= \sum_{i=1}^{n}\left(\sum_{j=1}^{i} j + \sum_{j=i+1}^{n} i\right) \\
&= \sum_{i=1}^{n}\left[\frac{i(i+1)}{2} + i(n-i)\right] \\
&= \sum_{i=1}^{n}\left[\frac{i(i+1)+2i(n-i)}{2}\right] \\
&= \sum_{i=1}^{n}\left[\frac{i[i+1+2(n-i)]}{2}\right] \\
&= \sum_{i=1}^{n}\left(i \times \frac{2n-i+1}{2}\right) \\
&= n \times \sum_{i=1}^{n} i + \sum_{i=1}^{n} \frac{i(1-i)}{2} \\
&= n \times \sum_{i=1}^{n} i + \frac{1}{2} \times \left[\sum_{i=1}^{n} i - \sum_{i=1}^{n} i^2\right] \\
&= n \times \frac{n(n+1)}{2} + \frac{1}{2}\left[\frac{n(n+1)}{2} - \frac{n(n+1)(2n+1)}{6}\right] \\
&= \frac{n(n+1)(2n+1)}{6}
\end{aligned}
$$

Algorithme 2 : Ex.5 Somme double - algorithme naïf

Données : n : un entier
Résultat : un entier

```
S : entier ← 0
s1 : entier
s2 : entier
pour i ← 1 à n faire
    s1 ← 0
    s2 ← 0
    pour j ← 1 à i faire
        s1 ← s1 + j
    pour j ← i + 1 à n faire
        s2 ← s2 + i
    S ← S + s1 + s2
retourner S
```

Terminaison et type d'instruction - ex. 6

La boucle `for` termine toujours (*attention aux modifications du compteur de boucle*), le nombre de tour de boucle étant défini. L'instruction problématique est la boucle `tant que`.

Algorithme 3 : Ex.6 Terminaison *Programme Terminator*

Données : n : un programme P
Résultat : un booléen
r : *booléen*
si P *termine* **alors**
  $r \leftarrow Vrai$
sinon
  $r \leftarrow Faux$
retourner r

Problème de l'arrêt - ex. 7

Algorithme 4 : Ex. Arrêt - SarahConnor

tant que $Terminator(SarahConnor)$ **faire**
  rien

Supposons que $SarahConnor$ termine. Dans ce cas, $Terminator(SarahConnor)$ renvoie **Vrai** et $SarahConnor$ boucle : on a une contradiction.

Fonction de Syracuse - ex. 8 et 9

Au point (*), pour $n = 15$ on a (15 46 23 70 35 106 53 160 80 40 20 10 5 16 8 4 2 1) et pour $n = 16$, on a (16 8 4 2 1).
L'algorithme retourne 1.

Exercice 10

i est un entier qui vaut initialement 0 et qui est incrémenté à chaque tour de boucle. Il est évident que la condition $i < 5$ ne sera plus satisfaite après 5 tours de boucle.

Exercice 11

On pose $f(i) = 5 - i$. La suite $(f(i_n))_{n\in\mathbb{N}}$ est décroissante.

Exercice 12

Un minorant de f tant qu'on est dans la boucle ($i < 5$) est 0.

Exercice 13

La fonction $\rho(i) = i,\ \forall i \in \mathbb{N}$ convient. Vérification :
— $\rho(i)$ est entier avant l'entrée dans la boucle ;
— $\rho(i)$ est entier dans la boucle ;
— $\rho(i)$ est strictement décroissante.

Exercice 14

Si n est pair : $\exists q \in \mathbb{N}^*$ tel que $n = 2q$. On a donc :

$$n = 2q > q = n \text{ div } 2$$

Si n est impair : $\exists q \in \mathbb{N}^*$ tel que $n = 2q + 1$. On a donc :

$$n = 2q + 1 > q = n \text{ div } 2$$

La fonction $\rho(i) = i$ div 2 convient. C'est une fonction à valeur entière à l'entrée et dans la boucle. De plus $\rho(i)$ est strictement décroissante.

Exercice 15

Le problème de terminaison se pose pour les valeurs de $0 < n \leq 5$. En effet, si $n > 5$ on ne passe qu'une fois dans la boucle, où x devient négatif.

$n = 1$	$n = 2$	$n = 3$	$n = 4$	$n = 5$
$\rho(1) = 2$	$\rho(2) = 2$	$\rho(3) = 4$	$\rho(4) = 3$	$\rho(5) = 1$
$\rho(8) = 1$	$\rho(6) = 1$	$\rho(4) = 3$	$\rho(2) = 2$	
		$\rho(2) = 2$	$\rho(6) = 1$	
		$\rho(6) = 1$		

La fonction $\rho(x)$ est entière avant et dans la boucle. De plus, elle est strictement décroissante. Par ailleurs, $\rho(n)$ donne le nombre de passage dans la boucle quand x est initialisé avec la valeur n.