Algorithme itératif
du
DRAPEAU HOLLANDAIS

(E.W.DIJKSTRA)

Introduction.

Face à un algorithme, nous sommes toujours amenés à nous poser ces questions fondamentales :

- S'il y a une solution la trouve-t-il toujours ?
- Est-on sûr qu'il se finira ?
- Comment peut-on le prouver ?

Le problème du Drapeau Tricolore.

Cet algorithme du néerlandais E.W.DIJKSTRA répond au problème suivant :

On remplit aléatoirement toutes les cellules d'un tableau de [1..n] cases avec des boules *bleues*, *blanches* et *rouges*.
On trie ensuite ce tableau afin d'obtenir le drapeau hollandais.

Pour cela on doit :

- utiliser ce seul tableau et le parcourir une seule fois.
- évaluer pour chaque boule, chaque couleur une seule fois au maximum.
- s'efforcer aussi de minimiser le nombre de permutations.

La Démarche.

On choisit 3 indices :
b et r serviront de bornes et indiqueront la position que devra prendre la prochaine boule correspondant à leur couleur.
i pointera le boule en cours d'évaluation.

1 b i r n

V : O O O O O O ? ? ? ? ? ? O O O O O

On initialise les 3 indices i := 1; b := 1 et r := n

```
 i
 b                r
 1                n
V : ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ?
```

Suivant la couleur dans les cellules du Tableau V[i], on effectue l'un des blocs d'instructions suivants :

Si V[i] est Blanc alors on augmente i

Si V[i] est Bleu   alors Permuter V[i] et V[b] puis incrémenter b et i

Si V[i] est Rouge c'est plus compliqué:

Il faut reculer l'indice r tant qu'il pointe sur une case rouge et qu'il ne devient pas plus petit que i.

tant que V[r]=Red) et (i<r) faire r := r - 1

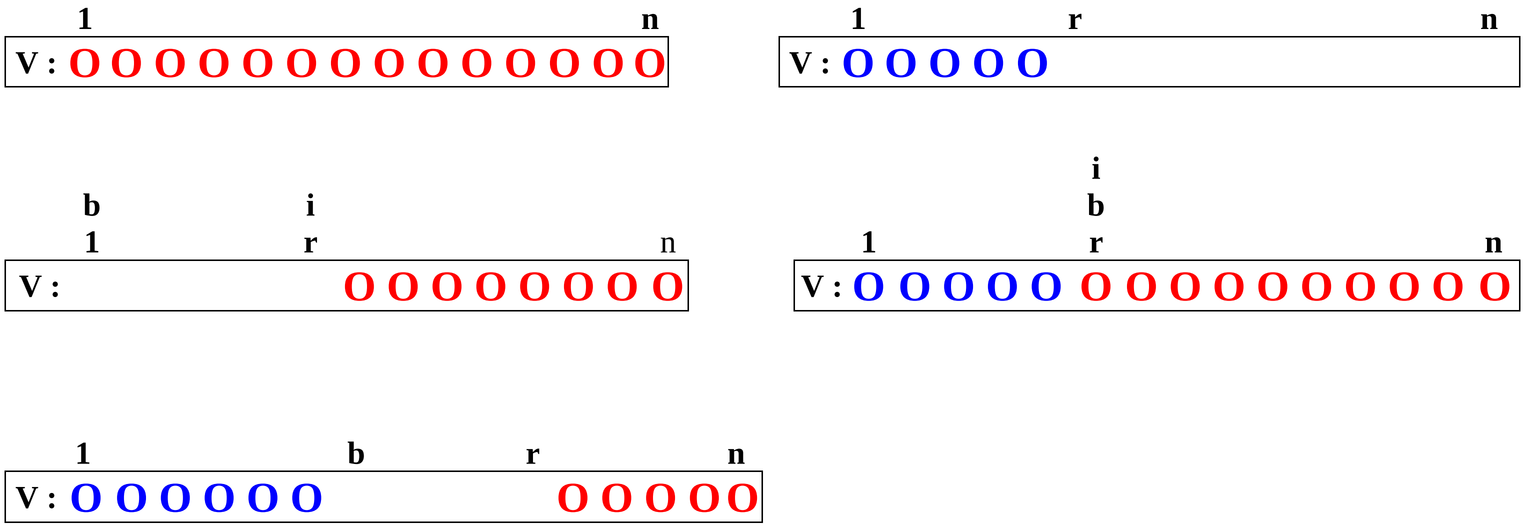
et ensuite Permuter V[i] et V[r] puis reculer r.

C'est terminé lorsque $i > r$, car soit i augmente, soit r diminue.

Tous les cas de fin de l'algorithme :

```
                         i
                         b                                     i
                         r       b                             r
 1                       n       1                             n
V : O O O O O O O O O O O O O O  V :
```

```
 i
 b                                     i                i
 r                                     b                r
```



Exemple de fonctionnement.

Initialisation

i := 1

b := 1

r := 9

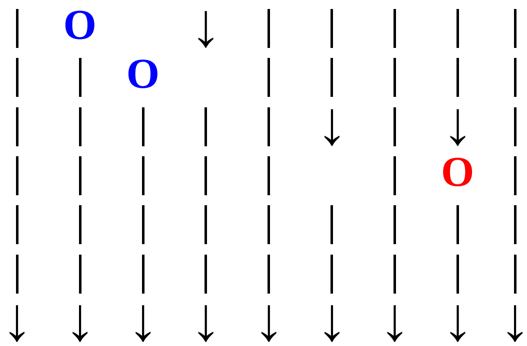
Nota :

i_x , b_x , r_x , *l'indice « x » indique l'état de la variable en sortie de l'itération « x »*

	i_0 b_0	i_2 b_2	i_3 b_4	i_4 b_5	i_5	i_6 r_9	i_8 r_7	r_1	r_0
V[i]	1	2	3	4	5	6	7	8	9
Aléatoire	O		O	O		O	O		O

		$i \leq r$?	Branchement	PermutCellule()	Affectation des variables
ITERATION	1	$i_0 \leq r_0$ = Vrai	Rouge(V[i_0]) = Vrai		$r_1 = r_0 - 1$
	2	$i_0 \leq r_1$ = Vrai	Bleu(V[i_0]) = Vrai		$b_2 = b_0+1$; $i_2 = i_0+1$
	3	$i_2 \leq r_1$ = Vrai	Blanc(V[i_2]) = Vrai		$i_3 = i_2+1$

4	$i_3 \leq r_1$ = Vrai	Bleu(V[i_3]) = Vrai	$b_4 = b_2+1$; $i_4 = i_3+1$
5	$i_4 \leq r_1$ = Vrai	Bleu (V[i_4]) = Vrai	$B_5 = b_4+1$; $i_5 = i_4+1$
6	$I_5 \leq r_1$ = Vrai	Blanc(V[i_5]) = Vrai	$i_6 = i_5+1$
7	$i_6 \leq r_1$ = Vrai	Rouge(V[i_6]) = Vrai	$r_7 = r_1$ -1
8	$i_6 \leq r_7$ = Vrai	Blanc (V[i_6]) = Vrai	$i_8 = i_6+1$
9	$i_8 \leq r_7$ = Vrai	Rouge(V[i_8]) = Vrai	$r_9 = r_7$ -1
10	$i_8 \leq r_9$ = Faux	FinTantQue	



RESULTAT

O	O	O				O	O	O