

Exemple d'état de jeu:

```
(  
  (A ((1 a) (2 a) (3 a)))  
  (B ((1 b) (2 b) (3 b)))  
  (C ())  
)
```

a a	b b	
aa aa	bb bb	
aaa aaa	bbb bbb	
=====		
A	B	C

```
(  
  (A ((1 a) (2 a) (3 a) (4 a)))  
  (B nil)  
  (C nil)  
  (D ((1 b) (2 b) (3 b) (4 b)))  
)
```

a a			b b
aa aa			bb bb
aaa aaa			bbb bbb
aaaa aaaa			bbbb bbbb
=====			
A	B	C	D

Manipulation du plateau de jeu:

Ajout du disque (1 b) à la tige 1:

```
(add-disc (1 b) 1 '(  
  (A ((1 a) (2 a) (3 a)))  
  (B ((2 b) (3 b)))  
  (C nil)  
))
```

Résultat:

```
(  
  (A ((1 a) (2 a) (3 a)))  
  (B ((1 b) (2 b) (3 b)))  
  (C nil)  
)
```

Déplacement du disque au sommet de la tige 1 sur la tige 2:

```
(move-disc 2 1 '(  
  (A ((1 a) (2 a) (3 a)))  
  (B ((1 b) (2 b) (3 b)))  
  (C nil)  
))
```

Résultat:

```
(  
  (A ((1 a) (2 a) (3 a)))  
  (B ((2 b) (3 b)))  
  (C ((1 b)))  
)
```

Vérification de la validité d'un mouvement effectué sur la tige 2:

```
(check-move 2 '(  
  (A ((1 a) (2 a) (3 a)))  
  (B ((2 b) (3 b)))  
  (C ((1 b)))  
))  
(check-move 2 '(  
  (A ((1 a) (2 a) (3 a)))  
  (B ((3 b)))  
  (C ((2 b) (1 b)))  
))
```

Résultat:

```
(  
  (A ((1 a) (2 a) (3 a)))  
  (B ((2 b) (3 b)))  
  (C ((1 b)))  
) nil
```



Génération de positions:

3 types de disques (a, b et c), 4 disques par types, à partir d'un jeu vide de trois tours:

```
(display-game  
  (generate-position '(a b c) 4 empty-state-3)  
)
```

```
      |           |           |  
      c|c         |           |  
      b|b         |           |  
      a|a         |           |  
      cc|cc       |           |  
      bb|bb       |           |  
      aa|aa       |           |  
      cccc|cccc   ccc|ccc     bbb|bbb  
      bbbb|bbbb   aaaa|aaaa   aaa|aaa  
=====
```

A	B	C
---	---	---

nil

Heuristique utilisant la distance de Belfort:

État de jeu donné:

```
(display-game state-4-4)
```

```
      |           |           |           |
      |           |           |           |
      |           |           |           |
      |           |           |           |
      |           |           |           |
      a|a         |         |         b|b
      aa|aa       |         |         bb|bb
      aaa|aaa     |         |         bbb|bbb
      aaaa|aaaa   |         |         bbbb|bbbb
=====
      A           B           C           D
nil
```

État de jeu final voulu:

```
(display-game state-4-5)
```

```
      |           |           |           |
      |           |           |           |
      |           |           |           |
      |           |           |           |
      |           |           |           |
      a|a         aa|aa       aaa|aaa     aaaa|aaaa
      b|b         bb|bb       bbb|bbb     bbbb|bbbb
=====
      A           B           C           D
nil
```

Valeur renvoyé par l'heuristique:

```
(heuristic state-4-4 state-4-5)
```

94

```
(heuristic state-4-4 state-4-4)
```

0

```
(heuristic state-4-5 state-4-5)
```

0

Génère les déplacements possibles d'un jeu en considérant qu'il n'y a pas eu de mouvement précédent.

```
(generate-moves '(
  (A ((1 a) (2 a) (3 a)))
  (B ((1 b) (2 b) (3 b)))
  (C ()))
) nil)
```

Résultat:

```
(
  ( (A B)
    (
      (A ((2 a) (3 a)))
      (B ((1 a) (1 b) (2 b) (3 b)))
      (C nil)
    )
  )
  ( (A C)
    (
      (A ((2 a) (3 a)))
      (B ((1 b) (2 b) (3 b)))
      (C ((1 a)))
    )
  )
  ( (B A)
    (
      (A ((1 b) (1 a) (2 a) (3 a)))
      (B ((2 b) (3 b)))
      (C nil)
    )
  )
  ( (B C)
    (
      (A ((1 a) (2 a) (3 a)))
      (B ((2 b) (3 b)))
      (C ((1 b)))
    )
  )
)
```