

Les Tuiles de Wang

Atelier de programmation pour enfants

Nicolas Seriot, avril 2023

But du projet

- Générer des images de pavages
 - aléatoires
 - sur des grilles de taille variables
 - avec des dessins différents (terrain, rails, tuyaux, etc)
- De l'idée au résultat: comment organiser un projet

Outils et techniques

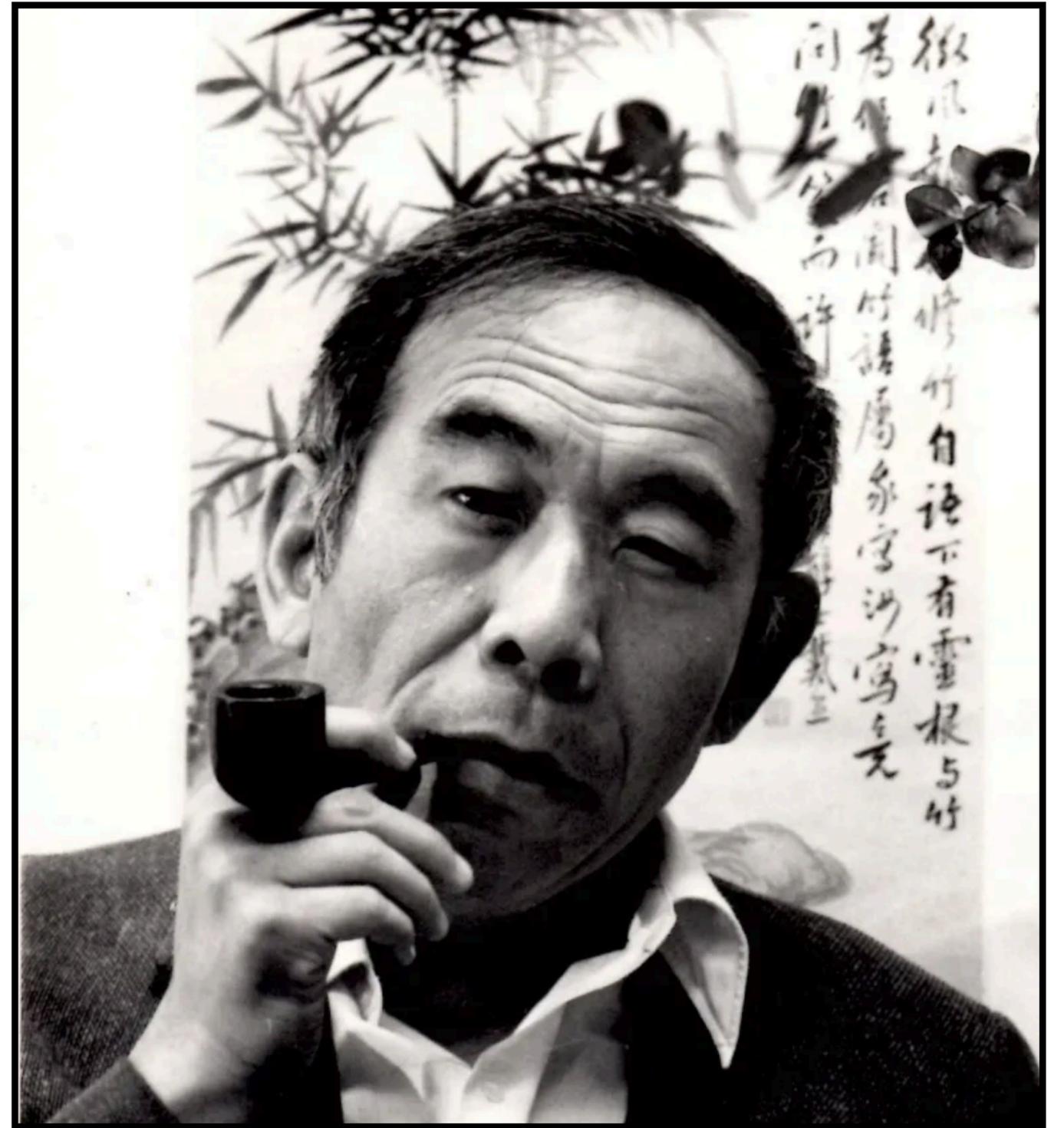
- Python
 - Listes et tableaux
 - Fonctions
 - Opérateurs binaires
- Pycairo
 - Translations
 - Crop

- 1. Introduction
 - Wang
 - Les tuiles
- 2. Créer le modèle
 - Listes et matrices
 - fill_board()
- 3. Créer la vue
 - draw_board()
 - draw_tile()
 - Sprites
- 4. Écrire l'algorithme
 - Opérateurs binaires
 - clean edges

1. Introduction

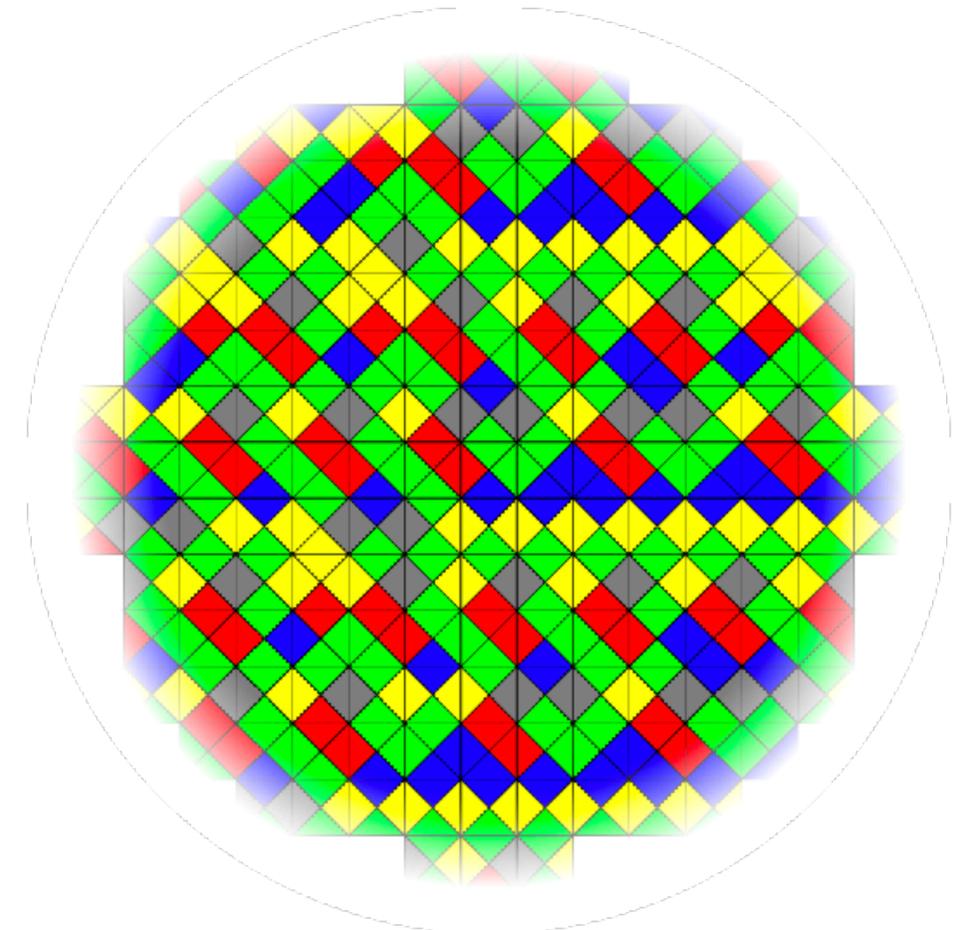
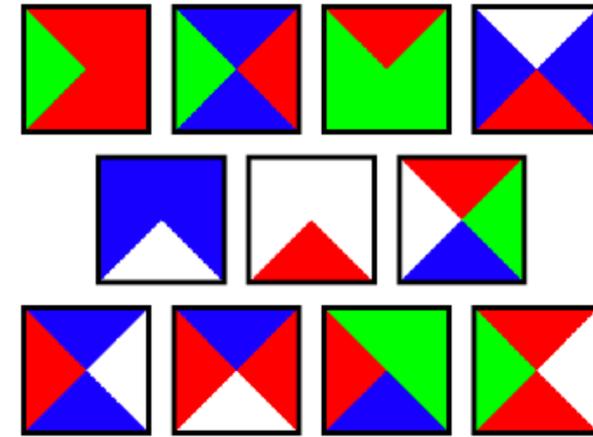
Hao Wang

- Mathématicien et philosophe
- Chinois et américain
- Dans les années 1960
- Intéressé par le pavage de surfaces



Les pavés (“tiles”)

- les tiles sont carrés
- chaque côté a une couleur
- l'orientation est fixe, pas de rotation
- les pavés côte à côte se correspondent
- on peut ainsi former des motifs infinis



Les listes et les tableaux

liste

```
l1 = [1,2,3]
```

```
l1[0] = 1
```

liste de listes

```
l2 = [ [1,2,3], [4,5,6] ]
```

```
l3 = [[1,2,3],  
      [4,5,6],  
      [7,8,9]]
```

```
l3[2][2]
```

Création de listes et tableaux

```
list(range(4)) # [0,1,2,3]
```

```
l4 = [0 for x in range(3)] # [0,0,0]
```

```
# initialiser un tableau avec des zéros
```

```
l5 = [[0 for x in range(3)] for x in range(3)] # [[0,0,0], [0,0,0], [0,0,0]]
```

Les fonctions

```
def mon_calcul(a, b, c):  
    x = a/b + c  
    return x
```

```
r1 = mon_calcul(4,2,1) # 3
```

```
r2 = mon_calcul(1,1,1) # 2
```

2. Remplir le tableau

Créer et remplir le tableau aléatoirement

```
def fill_board():  
    pass
```

3. Dessiner le tableau

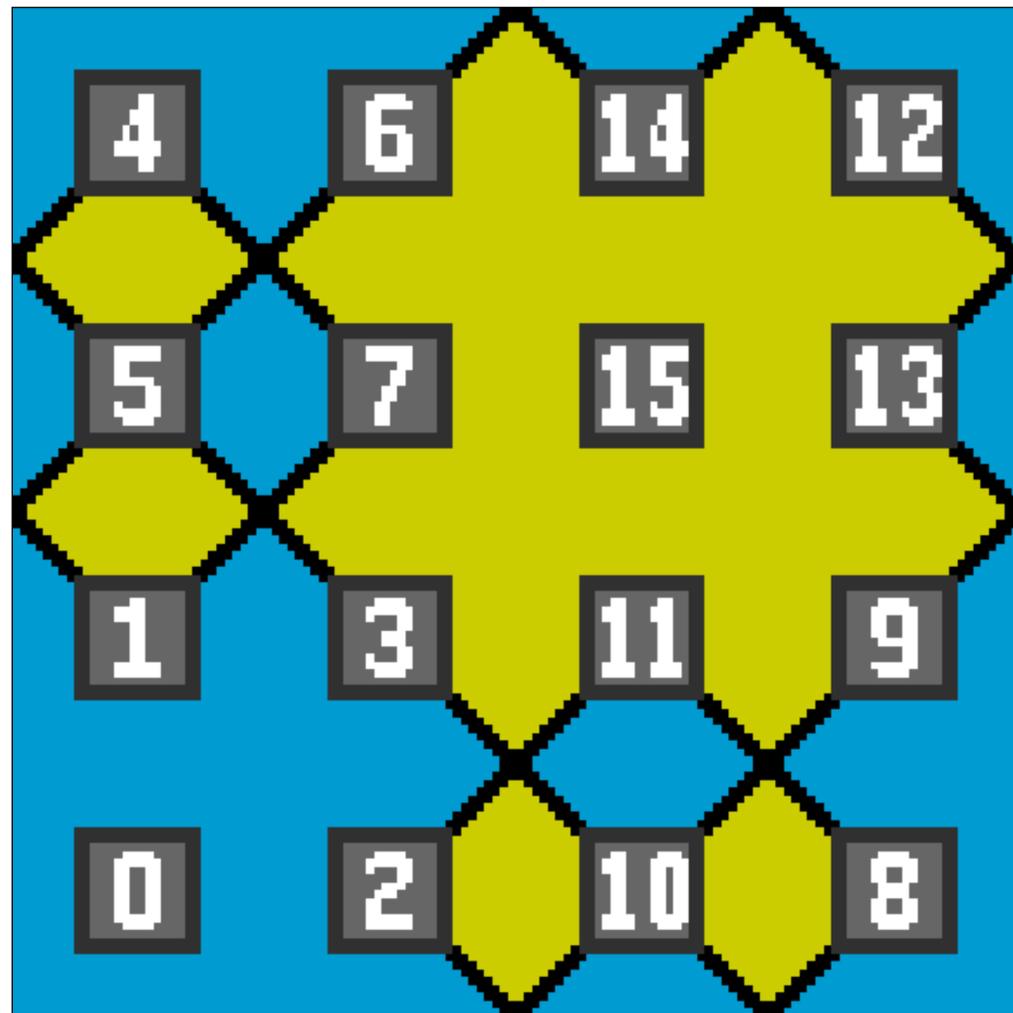
Dessiner le tableau

```
def draw_board():  
    pass
```

Les Sprites



Les Sprites



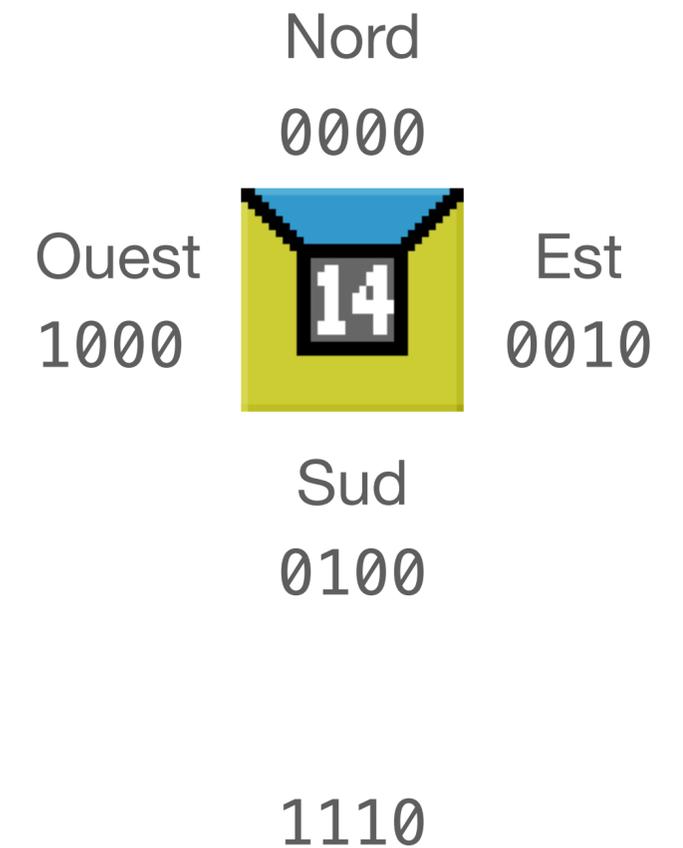
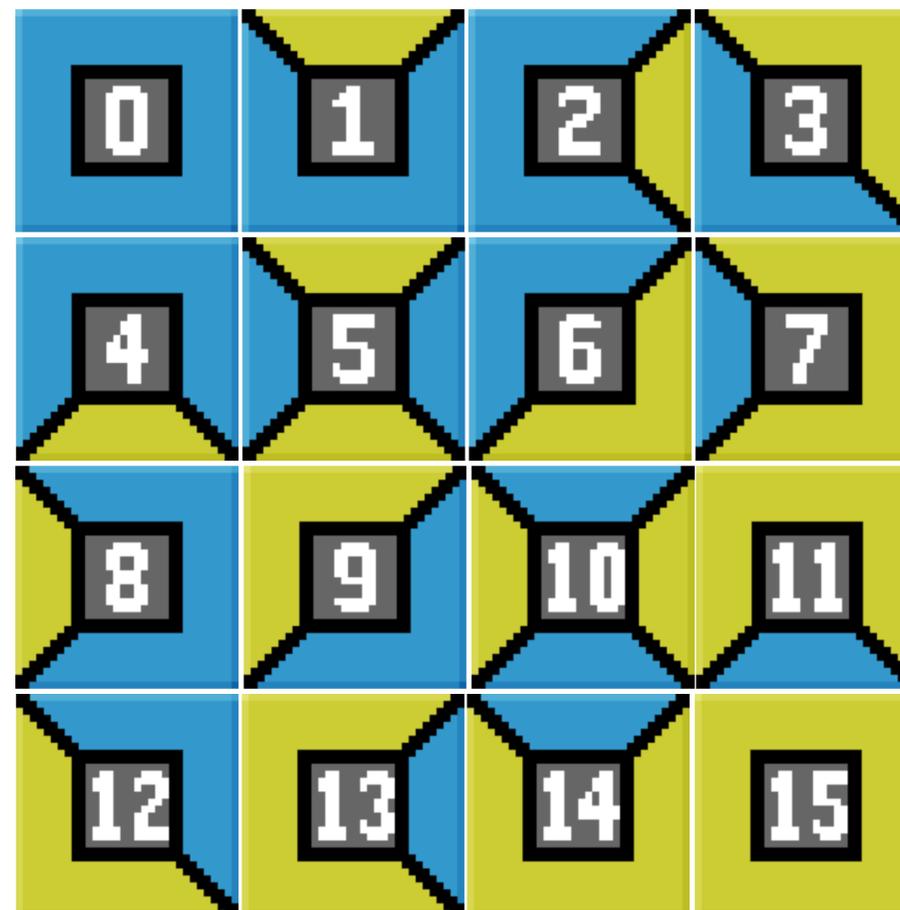
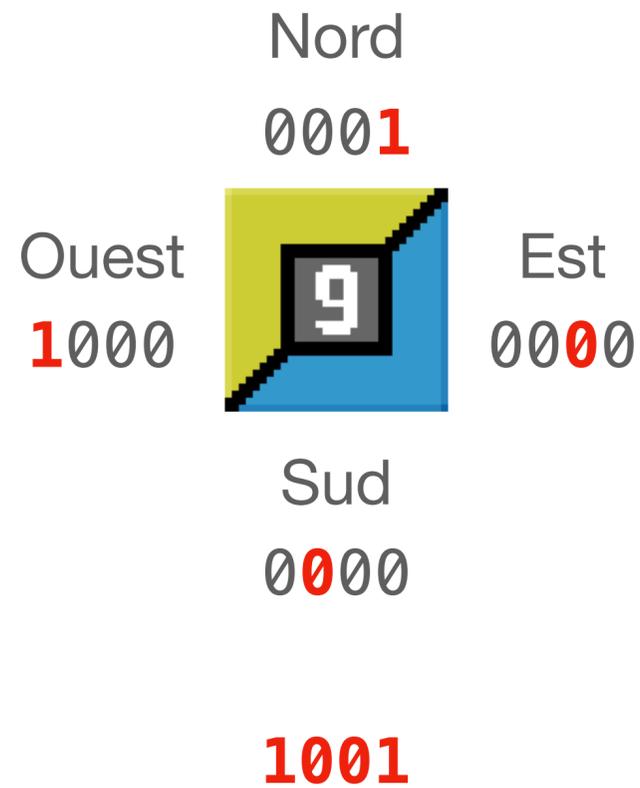
pos = [(0,3), (0,2), ...]

Dessiner une tile

```
def draw_tile():  
    pass  
  
def draw_board():  
    # ...  
    # draw_tile()  
    # ...
```

4. Faire que les côté se correspondent

Les tiles à 2 couleurs, codage binaire



$$1 \times 1 + 0 \times 2 + 0 \times 4 + 1 \times 8 = 9$$

Opérations sur les bits

ET logique
(si les deux sont vrais)

1 & 1 = 1
1 & 0 = 0
0 & 1 = 0
0 & 0 = 0

OU logique
(si au moins un des deux est vrai)

1 | 1 = 1
1 | 0 = 1
0 | 1 = 1
0 | 0 = 0

1100 & 1101 = 1100

0110 & 0001 = 0000

1110 & 1000 = 1000

1100 | 1101 = 1101

0110 | 0001 = 0111

1110 | 1000 = 1110

0010 | 0101 = 0110

L'algorithme

```
def fill_board():  
    # idée  
    # on parcourt toutes les cases de haut en bas et de gauche à droite  
    # on propage les couleurs d'est en ouest  
    # on propage les couleurs du sud au nord
```

Sources

- [https://en.wikipedia.org/wiki/Hao_Wang_\(academic\)](https://en.wikipedia.org/wiki/Hao_Wang_(academic))
- <http://www.cr31.co.uk/stagecast/wang/intro.html>
- https://en.wikipedia.org/wiki/Wang_tile

4	6	14	12
5	7	15	13
1	3	11	9
0	2	10	8

7	8	4	0	6	13	3	9	7	15	14	11	10	15	13	0	2	12	0	5	5	0	5	0	2	13	2	9	2	8	0	6
3	12	3	8	1	3	14	12	7	11	15	10	14	13	3	10	8	7	14	15	15	14	13	6	8	7	8	0	0	6	8	1
12	1	0	2	10	10	11	9	7	8	3	10	15	11	14	12	0	3	15	13	7	15	13	1	0	3	12	6	8	1	0	2
5	6	12	2	12	6	14	12	5	0	0	4	1	2	11	9	0	4	3	11	13	7	11	10	8	2	9	1	6	12	6	12
9	1	1	0	5	1	7	9	1	6	14	13	6	12	0	6	14	13	6	8	5	1	2	10	10	10	12	0	7	15	13	3
2	14	8	6	13	6	9	2	14	13	5	1	7	13	0	1	1	1	7	8	7	8	0	2	12	0	1	6	15	13	7	8
12	3	12	7	13	3	14	12	3	9	5	2	9	5	0	4	2	14	9	4	7	8	6	12	7	10	14	11	11	11	11	12
11	10	9	5	1	6	13	5	6	10	15	10	8	7	10	15	10	11	14	15	15	8	3	15	9	0	3	12	2	12	2	11
12	2	14	11	14	11	11	9	7	14	9	6	14	15	12	1	6	10	9	5	7	12	6	13	0	2	10	15	8	5	0	0
1	2	9	6	11	10	14	14	15	9	6	15	9	1	5	6	9	2	8	5	3	13	3	13	0	4	2	11	14	15	10	8
10	12	4	1	0	6	13	7	9	2	15	9	4	4	1	3	10	10	14	15	8	3	12	5	2	13	2	12	3	11	10	10
10	11	15	10	10	15	11	9	6	10	15	14	11	11	14	10	12	4	3	11	8	0	7	15	14	13	2	13	2	12	0	6
10	10	11	12	4	7	14	14	11	10	13	3	14	10	11	10	15	13	0	4	0	0	7	11	11	9	6	11	10	11	10	15
14	12	4	1	3	13	3	9	6	10	15	12	1	6	10	10	9	3	12	5	4	2	15	14	10	12	1	4	0	4	6	13
5	1	5	0	6	15	8	2	15	12	1	5	6	15	10	14	10	8	3	11	13	0	7	15	14	9	0	1	0	3	11	9
11	10	9	0	7	11	8	0	1	3	12	7	11	9	2	13	6	8	0	4	7	8	5	5	1	6	14	14	10	8	6	14
10	12	6	12	1	4	0	4	0	2	15	15	14	10	8	3	13	6	14	13	1	4	7	15	8	5	5	1	2	14	9	1
0	7	9	5	4	3	10	15	10	12	3	11	13	0	6	14	15	15	11	15	10	15	15	11	12	7	13	6	8	1	6	10
12	1	2	15	9	6	8	7	12	3	10	8	1	4	3	13	7	9	2	15	12	3	15	8	7	11	9	5	4	4	3	14
11	10	8	5	0	5	6	13	7	14	14	14	8	3	14	15	11	12	2	11	15	12	5	6	13	0	6	11	13	3	10	13
6	8	4	3	12	5	3	9	1	7	9	5	6	12	7	15	14	13	2	8	5	3	9	1	1	0	1	6	13	2	8	3
13	2	15	10	9	1	4	0	4	5	0	7	13	3	15	11	13	1	2	8	5	0	2	10	14	12	0	1	7	8	2	10
3	8	1	4	4	4	1	4	7	9	0	1	1	4	1	4	7	12	4	4	7	8	4	4	7	11	8	0	7	14	10	12
4	4	2	11	11	15	10	15	11	12	0	0	6	9	0	1	5	5	3	9	7	8	1	5	1	2	8	4	3	13	0	1
1	3	10	14	12	1	4	1	6	9	4	4	7	8	4	0	5	3	8	2	9	6	8	7	14	12	0	3	8	1	6	12
12	2	14	11	13	6	15	12	1	2	13	5	3	12	5	0	7	8	2	14	10	13	6	11	11	9	2	14	14	12	7	15
7	12	7	12	5	1	5	5	2	14	9	1	0	3	15	12	3	12	2	13	4	7	11	14	8	0	6	9	5	7	11	11
9	7	13	3	15	8	3	15	10	9	4	6	8	4	7	11	10	11	12	1	5	3	12	3	12	2	13	0	3	11	14	8
14	15	15	12	7	14	10	11	10	10	9	5	2	15	15	14	8	0	3	12	5	2	9	6	11	14	13	0	6	12	1	4
5	1	7	15	9	5	6	12	4	4	6	13	4	3	15	13	4	2	14	15	9	6	8	3	8	1	7	10	15	13	6	13
3	12	3	13	6	13	7	11	11	15	11	13	1	2	13	5	5	0	5	3	10	11	10	8	2	12	3	8	3	9	3	13
4	3	12	1	3	13	3	10	12	1	6	11	10	10	9	7	13	6	9	4	0	6	8	6	12	3	10	8	6	10	14	13

