
Default writeup

speed(3)

Headorteil



Table des matières

Introduction	3
Solution	3
Le flag	5

Introduction

Dans ce challenge, on dispose d'un site qui nous demande de lui renvoyer l'étape $n + \text{<étapes>}$ de l'état actuel d'une matrice composée de cases blanches et de cases noires. On peut lire qu'il s'agit du jeu de la vie, on peut donc deviner qu'on doit itérer selon les règles de « Conway's game of life ». On nous demande de répéter l'opération, j'ai donc mis mon code de résolution dans une boucle while.

Solution

```
1 import time
2 import math
3 import requests
4 import numpy
5
6 def voisins(i,j, taille) :
7     """retourne la liste des voisins de matrice[i][j]"""
8     T = []
9     if i != 0 :
10         T.append([i-1, j])
11         if j != 0 :
12             T.append([i-1, j-1])
13         if j != taille-1 :
14             T.append([i-1, j+1])
15     if i != taille-1 :
16         T.append([i+1, j])
17         if j != 0 :
18             T.append([i+1, j-1])
19         if j != taille-1 :
20             T.append([i+1, j+1])
21     if j != 0 :
22         T.append([i, j-1])
23     if j != taille-1 :
24         T.append([i, j+1])
25     return T
26
27 def viv(T) :
28     """retourne le nombre de voisins vivants parmi T qui est la liste
29         des voisins"""
30     nb=0
31     for c in T :
```

```
31         nb += M[c[0]][c[1]]
32     return nbr
33
34 r = requests.get("http://10.22.6.197/speed/3_rEivf0Kq-Dueb-xgzG-
    sKXdDJeahGWaPHnu/server.php", headers={"Cookie": "PHPSESSID=
    pjc4h1440bdp7384e9cjdkkup7"})
35
36 while 1 :
37     print(r.content)
38
39     data = r.json()
40     R=data["cases"]
41
42     taille = math.sqrt(len(R))
43
44     M = [[0]*taille]*taille
45     M=numpy.array(M)
46
47     #On transforme les données en matrice parce que en ligne c'est pas
    très pratique
48     for i in range(len(R)) :
49         M[i//taille][i%taille] = R[i]
50
51     N=[[0]*taille]*taille
52     N=numpy.array(N)
53
54     for Z in range(data["etapes"]) :
55         #on fait une boucle pour arriver au rang demandé
56
57         for i in range(taille) :
58             for j in range(taille) :
59                 a = viv(voisins(i, j, taille))
60                 if M[i][j] == 0 and a == 3 :
61                     N[i][j] = 1
62                 elif M[i][j] == 1 and a < 2 :
63                     N[i][j] = 0
64                 elif M[i][j] == 1 and (a == 2 or a == 3) :
65                     N[i][j] = 1
66                 elif M[i][j] == 1 and a > 3 :
67                     N[i][j] = 0
68             M = numpy.copy(N)
69
70     R=[0]*(taille**2)
```

```
71
72     #On retransforme notre matrice en liste pour renvoyer au bon format
73     for i in range(taille) :
74         for j in range(taille) :
75             R[i*taille+j]=N[i][j]
76
77     # On met un sleep, sinon on nous reproche de flood
78     time.sleep(2)
79
80     data = {}
81     for i in range(taille**2) :
82         data["cases[" + str(i) + "]" ] = R[i]
83
84     r = requests.post('http://10.22.6.197/speed/3_rEivf0Kq-Dueb-xgzG-
        sKXdDJeahGWaPHnu/server.php',headers={"Cookie" : "PHPSESSID=
        pjC4h1440bdp7384e9cjdkkup7"},data=data)
```

Le flag

On obtient alors le code suivant : 3G5w26l1-xtBL-TBS7-jZs1Kpsf2pmjFgfR

On peut alors rentrer le code dans le champ dédié sur le site et on obtient alors le flag suivant :
URCACTF{o6tV6KFi-WdPZ-c9g1-aBOCiv9gsDDeAvjd}