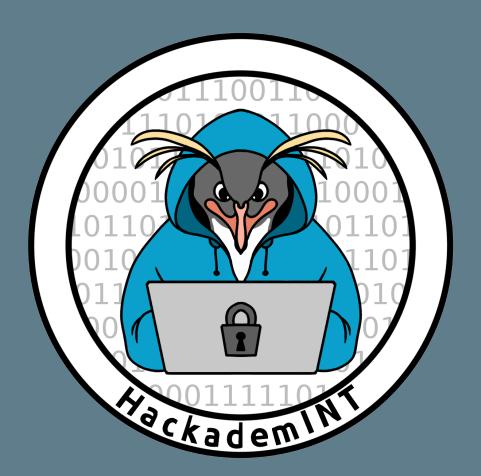
Default writeup

speed(3)

Headorteil



				• •
Tah	16	des	mat	ières

Introduction	з
Solution	3
Le flag	5

Introduction

Dans de challenge, on dispose d'un site qui nous demande de lui renvoyer l'étape n + <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 itérerer 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
   import requests
4
   import numpy
5
   def voisins(i,j, taille):
       """retourne la liste des voisins de matrice[i][j]"""
8
       T = []
       if i != 0 :
9
           T.append([i-1, j])
           if j != 0 :
11
12
                T.append([i-1, j-1])
            if j != taille-1 :
                T.append([i-1, j+1])
14
       if i != taille-1 :
           T.append([i+1, j])
           if j != 0 :
17
18
                T.append([i+1, j-1])
19
           if j != taille-1 :
20
                T.append([i+1, j+1])
21
       if j != 0 :
           T.append([i, j-1])
22
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
           des voisins"""
29
       nb=0
       for c in T:
```

```
31
           nb += M[c[0]][c[1]]
32
       return nbr
   r = requests.get("http://10.22.6.197/speed/3_rEivfOKq-Dueb-xgzG-
34
       sKXdDJeahGWaPHnu/server.php", headers={"Cookie":"PHPSESSID=
       pjc4h1440bdp7384e9cjdkkup7"})
35
   while 1:
       print(r.content)
37
38
       data = r.json()
       R=data["cases"]
40
41
       taille = math.sqrt(len(R))
42
43
44
       M = [[0]*taille]*taille
       M=numpy.array(M)
45
46
47
       #On transforme les données en matrice parce que en ligne c'est pas
           trés pratique
       for i in range(len(R)):
48
           M[i//taille][i%taille] = R[i]
49
51
       N=[[0]*taille]*taille
52
       N=numpy.array(N)
53
       for Z in range(data["etapes"]):
54
55
       #on fait une boucle pour arriver au rang demandé
57
            for i in range(taille):
                for j in range(taille):
                    a = viv(voisins(i, j, taille))
                    if M[i][i] == 0 and a == 3:
60
                        N[i][j] = 1
61
                    elif M[i][j] == 1 and a < 2:
62
63
                        N[i][j] = 0
                    elif M[i][j] == 1 and (a == 2 or a == 3):
64
                        N[i][j] = 1
65
                    elif M[i][j] == 1 and a > 3:
66
                        N[i][j] = 0
67
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):
           for j in range(taille):
74
               R[i*taille+j]=N[i][j]
75
76
       # 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
       r = requests.post('http://10.22.6.197/speed/3_rEivfOKq-Dueb-xgzG-
84
           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{06tV6KFi-WdPZ-c9g1-aBOCiv9gsDDeAvjd}