

(https://recursospython.com/)

Inicio (https://recursospython.com/)

Códigos de fuente (https://www.recursospython.com/category/codigos-de-fuente/)

Guías y manuales (https://www.recursospython.com/category/guias-y-manuales/)

Foro (https://foro.recursospython.com/) Micro (https://micro.recursospython.com/)

Tutorial (https://tutorial.recursospython.com/)

Newsletter (https://recursospython.com/newsletter/)

Consultoría (https://recursospython.com/consultoria/)

Contacto (https://recursospython.com/contacto/) Donar ♥ (https://recursospython.com/donar/)

Solucionador de Sudoku

diciembre 26, 2016 (https://recursospython.com/codigos-de-fuente/solucionador-de-sudoku/) by Recursos Python (https://recursospython.com/author/admin/) (https://recursospython.com/codigos-de-fuente/solucionador-de-sudoku/#comments)Dejar un comentario (https://recursospython.com/codigos-de-fuente/solucionador-de-sudoku/#respond)

Descarga: <u>sudokusolver.zip (https://www.recursospython.com/wp-content/uploads/2016/12/sudokusolver.zip)</u>.

El siguiente código de alrededor de 150 líneas es un simple

solucionador del juego <u>Sudoku</u> (https://es.wikipedia.org/wiki/Sudoku) escrito en Python 3.

Basándose en las reglas del juego, el pequeño programa determina un conjunto de números posibles para cada uno de los casilleros vacíos, y luego aplica fuerza bruta (seleccionar números aleatoriamente hasta obtener el resultado correcto). La fuerza bruta es sin duda uno de los métodos más simples para resolver este tipo de juegos matemáticos, pero no por eso poco eficaz. No será un

(https://es.wikipedia.org/wiki/Aprendizaje_autom%C3%A1tico) pero ha logrado pasar todas las pruebas: desde las grillas más simples a las más avanzadas. El tiempo de resolución del script es generalmente de unos pocos segundos.

1.	#!/usr/bin/env python	
2.	# -*- coding: utf-8 -*-	

from random import choice

complejo modelo de *Machine Learning*

Búsqueda ...

Q

Últimas entradas

Reproducir inyección
SQL en sqlite3 y
PyMySQL
(https://recursospython.com/gui
y-manuales/reproducirinyeccion-sql-ensqlite3-y-pymysql/)

Bloc de notas simple con

Tk (tkinter)

(https://recursospython.com/co/
de-fuente/bloc-denotas-simple-contkinter/)

Examinar archivo o
carpeta en Tk (tkinter)
(https://recursospython.com/gui
y-manuales/examinararchivo-o-carpeta-en-tktkinter/)

```
5.
6.
      def number_in_row(grid, row, number):
7.
8.
9.
          Chequear si un número se encuentra en la fila
10.
11.
          return number in grid[row]
12.
13.
      def number in col(grid, col, number):
14.
15.
16.
          Chequear si un número se encuentra en la columna
      especificada.
17.
          return number in (row[col] for row in grid)
18.
19.
20.
21
      def number in box(grid, row, col, number):
22.
23.
          Chequear si un número se encuentra en la caja a la que
          corresponde la posición especificada.
24.
25.
26.
          # Obtener la caja a la que pertenece el número.
27.
          box row, box col = box by pos(row, col)
          # Construir una lista con los números en la caja.
28.
          numbers in box = unpack(
29.
              row[box col*3:box col*3 + 3]
30.
31.
              for row in grid[box row*3:box row*3 + 3]
32.
          return number in numbers in box
33.
34.
35.
36.
      def reduce(n):
37.
38.
          Reducir la posición 9x9 a 3x3.
39.
          n /= 3
40.
          if n == 0 or n != int(n):
41.
42.
          return int(n)
43.
44.
45.
46.
      def box_by_pos(row, col):
47.
          # Trabajar temporalmente con base 1.
          row += 1
48.
49.
          col += 1
50.
          # Obtener base 0 nuevamente.
51.
          return reduce(row) - 1, reduce(col) - 1
52.
53.
54.
      def unpack(iterable):
55.
56.
57.
          >>> list(unpack([[1, 2], [3, 4]]))
58.
          [1, 2, 3, 4]
          ,, ,, ,,
59.
60.
          for item in iterable:
              yield from item
61.
62.
63.
64.
      def get possible numbers(grid, row, col):
65.
          Retorna números posibles para una determinada posición.
66.
67.
68.
          for number in range(1, 10):
```

Múltiples
configuraciones
(desarrollo/producción)
en Django
(https://recursospython.com/gui
y-manuales/multiplesconfiguracionesdesarrollo-produccionen-django/)

Buscar el archivo de
mayor tamaño en una
ruta
(https://recursospython.com/co/
de-fuente/buscar-elarchivo-de-mayortamano-en-una-ruta/)

Comentarios recientes

Recursos Python en Generar código QR (https://recursospython.com/gui y-manuales/generarcodigo-qr/#comment-2586) Joaquín en Generar código QR (https://recursospython.com/gui y-manuales/generarcodigo-qr/#comment-2584) Recursos Python en pickle - Serialización de <u>objetos</u> (https://recursospython.com/gui <u>y-manuales/pickle-</u> serializacion-deobjetos/#comment-2435)

Recursos Python en Lista

desplegable (Combobox)

```
if (not number in row(grid, row, number) and
 69.
                  not number in col(grid, col, number) and
 70.
                  not number in box(grid, row, col, number)):
 71
 72.
                   yield number
 73.
 74
      def main():
 75.
 76.
          while True:
 77.
               # Los ceros representan casilleros vacíos.
 78.
                   [0, 0, 0, 8, 0, 3, 0, 0, 0],
 79.
                  [0, 1, 0, 9, 0, 0, 7, 0, 8],
 80.
                  [0, 0, 0, 0, 4, 0, 0, 0, 9],
 81.
                  [2, 0, 0, 0, 0, 4, 0, 6, 0],
 82.
                   [4, 0, 7, 0, 0, 0, 9, 0, 5],
 83.
                  [0, 5, 0, 7, 0, 0, 0, 0, 1],
 84.
                  [7, 0, 0, 0, 3, 0, 0, 0, 0],
 85.
                  [3, 0, 6, 0, 0, 8, 0, 7, 0],
 86.
                  [0, 0, 0, 2, 0, 7, 0, 0, 0],
 87.
 88.
 89.
               s = \
 90.
              """\
 92.
               | { } { } { } | { } | { } | { } | { } |
 93.
 94.
               | {} {} {} | {} | {} |
 95.
 96.
 97.
               | {} {} {} | {} | {} |
               | { } { } { } | { } | { } | { } | { } |
98.
99.
               | { } { } { } | { } | { } | { } | { } |
100.
101.
               | {} {} {} | {} | {} | | |
               | { } { } { } | { } | { } | { } | { } |
               | {} {} {} | {} | {} |
103.
104.
               +----+
               ,, ,, ,,
105.
106
107.
               while True:
108.
                  possible numbers = {
109.
                       (row, col): None for row in range(9) for
      col in range(9)
110.
111.
112.
                   # Obtener una lista de números posibles para
      cada una de
113.
                   # las posiciones vacías.
                   for row in range(9):
114.
115.
                       for col in range(9):
                           number = grid[row][col]
117.
                           if number == 0:
118.
                               options = list(
119.
                                   get possible numbers (grid, row,
      col)
120.
                               )
121.
                               if options:
122.
                                   possible numbers[(row, col)] =
      options
123.
124.
                   # Remover valores vacíos y ordenar por la
      cantidad de
                   # posibilidades.
125.
126.
                  possible numbers = sorted(
127.
                       (
128.
                           (k, v)
129.
                           for (k, v) in possible_numbers.items()
                           if v is not None
```

en Tcl/Tk (tkinter)
(https://recursospython.com/gui
y-manuales/listadesplegable-comboboxen-tkinter/#comment2434)
Herná en Lista
desplegable (Combobox)
en Tcl/Tk (tkinter)
(https://recursospython.com/gui
y-manuales/listadesplegable-comboboxen-tkinter/#comment2424)

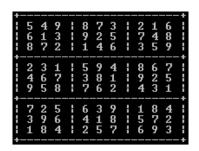
```
131.
132.
                       key=lambda kv: len(kv[1])
133.
                   if possible numbers:
135.
                       # Obtener el primer item.
136.
137.
                       (row, col), numbers = possible numbers[0]
                       # Fuerza bruta: obtener un número aleatorio
138.
      de la
139.
                       # lista de posibiilidades hasta que la
      grilla esté
140.
                       # completa.
                       grid[row][col] = choice(numbers)
141.
142.
                   else:
143.
                       break
144.
               # Chequear si la fuerza bruta dió resultado: si no
145.
      hay más
146.
               # ceros en la grilla entonces el Sudoku está
      resuelto.
147.
               if 0 not in unpack(grid):
                   print(s.format(*(unpack(grid))))
148.
149.
150.
151.
152.
      if __name__ == "__main__":
153.
          main()
```

Traté de mantener el código lo más funcional, comentado y legible posible. La grilla por defecto representa el siguiente juego de Sudoku:

			8		3			
	1		9			7		8
				4				9
2					4		6	
4		7				9		5
	5		7					1
7				3				
3		6			8		7	
			2		7			

(https://www.recursospython.com/wp-content/uploads/2016/12/sudoku.png)

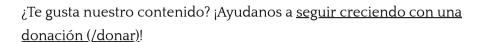
Y la solución provista por el programa es:



(https://www.recursospython.com/wp-content/uploads/2016/12/sudoku-solucion.png)

Para resolver tus propias partidas de Sudoku, simplemente debes editar los elementos de la lista grid.

Donar 💚



Entrada publicada en Códigos de fuente (https://recursospython.com/category/codigos-de-fuente/) CON las etiquetas fuerza-bruta (https://recursospython.com/tag/fuerza-bruta/) sudoku (https://recursospython.com/tag/sudoku/)

◆ Formulario de contacto vía CGI
(https://recursospython.com/codigos-de-fuente/formulario-decontacto-via-cgi/)

<u>Diferencia entre variables en Python y otros lenguajes</u> (https://recursospython.com/guias-y-manuales/diferencia-entre-variables-en-python-y-otros-lenguajes/)

Deja una respuesta



© 2013 - 2023

¡Suscríbete a nuestra newsletter! (https://www.recursospython.com/newsletter/)

Políticas de Uso y Privacidad (https://www.recursospython.com/politicas-de-uso-y-privacidad/)

En inglés: Python Assets (https://pythonassets.com/)

(https://creativecommons.org/licenses/by-nc/3.0/deed.es)