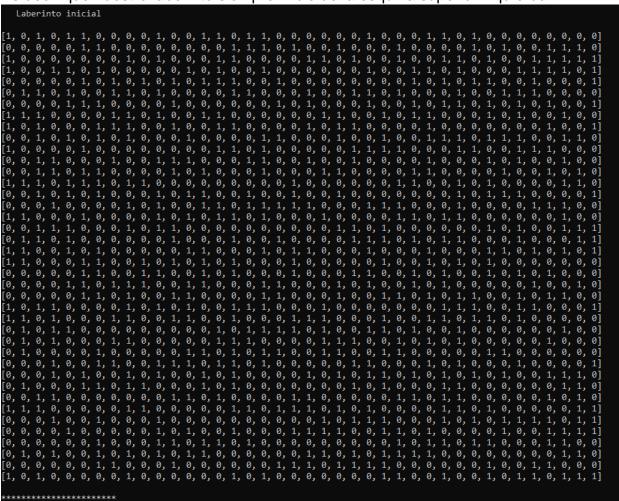
PROGRAMA LABERINTO INTELIGENTE EN PYTHON

 El primer paso del programa es generar una matriz de bits aleatorios de 40*40 a la cual vamos a forzar para que su primer valor es decir el valor [0][0] de la matriz siempre sea 1.

Es decir que nuestro laberinto siempre inicie de la esquina superior izquierda.



- 2. Después de generar la matriz aleatoria con nuestra puerta utilizamos el método A* para buscar si en este ya se encuentra un camino que una la primera fila con la última. En otras palabras, que recorra de nuestra puerta a alguna salida (1) de la ultima fila. De ser que no continuamos y mutamos.
- 3. Mutamos para ir generando un camino desde nuestra puerta hasta la última fila, para esto localizamos el último punto donde se genera un camino y a partir de esto mutamos un valor aleatorio inferior o derecho, para que de esta forma forcemos a que el camino se acerque a la ultima fila. Guardamos las posiciones de los (0) mutados para compensar en la fila según el porcentaje que se haya seleccionado.

4. Posteriormente volvemos a aplicar A* para localizar que nuestro camino sea el mejor y obtener al mismo tiempo el peso de este.

5. Imprimimos las coordenadas del camino desarrollado:

Caminito

- [1,0]
- [2,0]
- [3,0]
- [3,1]
- [3,2]
- [3,3]
- [3,4]
- [3,5]
- [3,6]
- [3,7]
- [4,7]
- [5,7]
- [۲٫۰]
- [5,8]
- [5,9]
- [6,9]
- [6,10]
- [6,11]
- [7,11]

- [8,11]
- [9,11]
- [9,12]
- [9,13]
- [10,13]
- [10,14]
- [10,15]
- [10,16]
- [10,17]
- [11,17]
- [12,17]
- [13,17]
- [14,17]
- [15,17]
- [15,18]
- [15,19]
- [15,20]
- [16,20]
- [16,21]
- [17,21]
- [17,22]
- [17,23]
- [18,23]
- [18,24]
- [18,25]
- [18,26]
- [18,27]
- [19,27]
- [19,28]
- [20,28]
- [20,29]
- [20,30]
- [21,30]
- [22,30]
- [23,30]
- [24,30]
- [24,31]
- [25,31]
- [25,32]
- [25,33]
- [25,34]
- [25,35]
- [26,35]
- [26,36]

[27,36] [27,37] [27,38] [28,38] [29,38] [30,38] [31,38] [32,38] [33,38] [34,38] [35,38] [35,37] [35,36] [34,36] [34,35] [34,34] [35,34] [36,34] [37,34] [37,35] [38,35] [39,35]

- 6. Guardamos las coordenadas de la salida y repetimos el proceso de manera inversa con la condicional de que la salida de este nuevo camino sea la entrada[0][0] de la matriz.
- 7. De esta forma finalmente obtenemos un segundo camino.

```
Laberinto de vuelta
54, 0, 0, 0,
              0, 1, 1, 0,
                  0, 0, 1, 0,
                           1, 0, 0,
 0, 0]
                                    0,
                                  0, 0,
 1, 0, 1, 0,
0, 0, 0, 0,
                                        0]
            0, 0,
      0, 0, 1,
          0, 0, 0, 0,
        0, 0, 1, 0,
    0, 0, 1, 0,
  0, 0, 0, 0,
        0, 0,
          0, 1,
1, 0,
  0, 0,
        0, 0,
  0,
    0,
    0,
          0, 0,
                          0, 0,
                                        0]
                0, 1,
                  0, 0, 0, 0, 0, 0, 0, 1,
*******
```

8. Finalmente volvemos a emplear el A* para determinar el mejor camino encontrado e imprimirlo

- De esta forma ya tenemos un laberinto con dos caminos y un camino seleccionado como el mejor.
- Finalmente, solo imprimimos el laberinto resúltate de manera grafica para poder identificar todo con mayor claridad.

