**Autómatas celulares elementales**

La [Teoría de la Computabilidad](https://es.wikipedia.org/wiki/Teor%C3%ADa_de_la_computabilidad) trata de estudiar los problemas que se pueden resolver con un algoritmo o, equivalentemente, con una [Máquina de Turing](https://es.wikipedia.org/wiki/M%C3%A1quina_de_Turing)

En esta práctica trabajaremos con uno de los formalismos propuestos durante la historia para formular matemáticamente el concepto de algoritmo: los [autómatas celulares elementales](https://en.wikipedia.org/wiki/Elementary_cellular_automaton).

Un autómata celular elemental es un *mecanismo*que transforma una secuencia de bits (ceros o unos) en otra aplicando reglas sencillas. Estas reglas indican como cambia un bit teniendo en cuenta el valor actual y los valores de los bits contíguos. Por ejemplo, consideremos el autómata dado por la siguiente tabla

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 111 | 110 | 101 | 100 | 011 | 010 | 001 | 000 |  |
| 0 | 1 | 1 | 0 | 1 | 1 | 1 | 0 |  |

En cada casilla de la fila superior se indican el estado de los bits en la cadena original, el del centro se corresponde con el bit a modificar y los otros los que están a derecha a izquierda. La fila de abajo representa el nuevo bit obtenido.

Las reglas se aplican a secuencias de bits. Los bits que están a los extremos de la secuencia se quedan inalterados. Entonces si tomamos la secuencia de bits 00010011011111, tendremos

1. El primer bit se queda inalterado: 0
2. Para el segundo tenemos que aplicar el resultado de 000: 0
3. Para el tercero tenemos que aplicar el resultado de 001: 1
4. Para el tercero tenemos que aplicar el resultado de 010: 1
5. Para el quinto tenemos que aplicar el resultado de 100: 0
6. Para el sexto tenemos que aplicar el resultado de 001: 1
7. Para el séptimo tenemos que aplicar el resultado de 011: 1
8. Para el octavo tenemos que aplicar el resultado de 110: 1
9. Para el noveno tenemos que aplicar el resultado de 101: 1
10. Para el décimo tenemos que aplicar el resultado de 011: 1
11. Para el undécimo en adelante tenemos que aplicar el resultado de 111: 0

Así obtenemos el resultado 00110111110001 en una iteración. El número de iteraciones se puede repetir en número que se desee.

Sólamente hay 28 autómatas diferentes que se pueden expresar en este formalismo. Se nombran por la secuencia de bits de la segunda fila de la tabla anterior. Por tanto este autómata es el autómata 01101110 (en base 2), más conocido por su representación en base diez: 110. Resulta que este autómata tiene la potencia de las [Máquinas de Turing](https://es.wikipedia.org/wiki/M%C3%A1quina_de_Turing).

En este problema se realizará un programa que realize los cómputos de un autómata de este tipo. Para ello realizaremos una función automaton(datafile). El parámetro datafile es una variable de tipo fichero (no una ruta) que previamente ha sido abierto con los datos que queremos ejecutar. En concreto el fichero tiene tres líneas:

* La primera tiene los datos del autómata. Sólamente necesitamos la segunda fila de la anterior tabla. Será una fila que contenga 8 ceros y unos separados por espacios.
* La segunda línea tiene la secuencia de ceros y unos que queremos procesar. En esta línea no hay separación entre los ceros y los unos.
* La tercera línea tiene número de pasos que queremos dar.

Por ejemplo, el contenido del fichero es el siguiente:

0 1 1 0 1 1 1 0

00010011011111

5

La función debe devolver en una cadena de caracteres la secuencia de unos y ceros resultante de aplicar el autómata a la cadena original, en el caso del ejemplo 5 pasos. El resultado en este caso es:

01011111000111