Capítulo 5

Tipos estructurados: secuencias

> *Primero llegaron diez soldados portando bastos: tenían la misma forma que los tres jardineros, plana y rectangular, con las manos y los pies en las esquinas; luego venían los diez cortesanos, todos adornados de diamantes, y caminaban de dos en dos, como los soldados. Seguían los Infantes: eran diez en total y era encantador verlos venir cogidos de la mano, en parejas, dando alegres saltos: estaban adornados con corazones.*
>
> Lewis Carroll, *Alicia en el país de las maravillas*.

Hasta el momento hemos tratado con datos de tres tipos distintos: enteros, flotantes y cadenas. Los dos primeros son tipos de datos *escalares*. Las cadenas, por contra, son tipos de datos *secuenciales*. Un dato de tipo escalar es un elemento único, atómico. Por contra, un dato de tipo secuencial se compone de una *sucesión* de elementos y una cadena es una sucesión de caracteres. Los datos de tipo secuencial son *datos estructurados*. En Python es posible manipular los datos secuenciales de diferentes modos, facilitando así la escritura de programas que manejan conjuntos o series de valores.

En algunos puntos de la exposición nos desviaremos hacia cuestiones relativas al modelo de memoria de Python. Aunque se trata de un material que debes comprender y dominar, no pierdas de vista que lo realmente importante es que aprendas a diseñar e implementar algoritmos que trabajan con secuencias.

En este tema empezaremos aprendiendo más de lo que ya sabemos sobre *cadenas*. Después, te presentaremos las *listas*. Una lista es una sucesión de elementos de cualquier tipo. Finalmente, aprenderás a definir y manejar *matrices*: disposiciones bidimensionales de elementos. Python no incorpora un tipo de datos nativo para matrices, así que las construiremos como *listas de listas*.

5.1. Cadenas

5.1.1. Lo que ya sabemos

Ya vimos en temas anteriores que una *cadena* es una sucesión de caracteres encerrada entre comillas (simples o dobles). Python ofrece una serie de operadores y funciones predefinidos que manipulan cadenas o devuelven cadenas como resultado. Repasemos brevemente las que ya conocemos de temas anteriores:

- Operador **+** (concatenación de cadenas): acepta dos cadenas como operandos y devuelve la cadena que resulta de unir la segunda a la primera.
- Operador ***** (repetición de cadena): acepta una cadena y un entero y devuelve la concatenación de la cadena consigo misma tantas veces como indica el entero.

- Operador `%` (sustitución de marcas de formato): acepta una cadena y una o más expresiones (entre paréntesis y separadas por comas) y devuelve una cadena en la que las marcas de formato (secuencias como `%d`, `%f`, etc.) se sustituyen por el resultado de evaluar las expresiones.
- *int*: recibe una cadena cuyo contenido es una secuencia de dígitos y devuelve el número entero que describe.
- *float*: acepta una cadena cuyo contenido describe un flotante y devuelve el flotante en cuestión.
- *str*: se le pasa un entero o flotante y devuelve una cadena con una representación del valor como secuencia de caracteres.
- *ord*: acepta una cadena compuesta por un único carácter y devuelve su código ASCII (un entero).
- *chr*: recibe un entero (entre 0 y 255) y devuelve una cadena con el carácter que tiene a dicho entero como código ASCII.

Podemos manipular cadenas, además, mediante métodos que les son propios:

- *a*`.lower()` (paso a minúsculas): devuelve una cadena con los caracteres de *a* convertidos en minúsculas.
- *a*`.upper()` (paso a mayúsculas): devuelve una cadena con los caracteres de *a* convertidos en mayúsculas.
- *a*`.capwords()` (paso a palabras con inicial mayúscula): devuelve una cadena en la que toda palabra de *a* empieza por mayúscula.

Aprenderemos ahora a utilizar nuevas herramientas. Pero antes, estudiemos algunas peculiaridades de la codificación de los caracteres en las cadenas.