6.B. Cadenas de caracteres.

1. Cadenas de caracteres.

Probablemente, una de las cosas que más utilizarás cuando estés programando en cualquier lenguaje de programación son las cadenas de caracteres. Las cadenas de caracteres son estructuras de almacenamiento que permiten almacenar una secuencia de caracteres de casi cualquier longitud. Y la pregunta ahora es, ¿qué es un carácter?

En Java y en todo lenguaje de programación, y por ende, en todo sistema informático, los caracteres se codifican como secuencias de bits que representan a los símbolos usados en la comunicación humana. Estos símbolos pueden ser letras, números, símbolos matemáticos e incluso ideogramas y pictogramas.

Para saber más

Si quieres puedes profundizar en la codificación de caracteres leyendo el siguiente artículo de la Wikipedia.

Codificación de caracteres.

La forma más habitual de ver escrita una cadena de caracteres es como un literal de cadena. Consiste simplemente en una secuencia de caracteres entre comillas dobles, por ejemplo:

```
"Ejemplo de cadena de caracteres".
```

En Java, los literales de cadena son en realidad instancias de la clase `String`, lo cual quiere decir que, por el mero hecho de escribir un literal, se creará una instancia de dicha clase. Esto da mucha flexibilidad, puesto que permite crear cadenas de muchas formas diferentes, pero obviamente consume mucha memoria. La forma más habitual es crear una cadena partiendo de un literal:

```
String cad="Ejemplo de cadena";
```

En este caso, el literal de cadena situado a la derecha del igual es en realidad una instancia de la clase `String`. Al realizar esta asignación hacemos que la variable `cad` se convierta en una referencia al objeto ya creado. Otra forma de crear una cadena es usando el operador `new` y un constructor, como por ejemplo:

```
String cad=new String ("Ejemplo de cadena");
```

Cuando se crean las cadenas de esta forma, se realiza una copia en memoria de la cadena pasada por parámetro. La nueva instancia de la clase `String` hará referencia por tanto a la copia de la cadena, y no al original.

Reflexiona

Fijate en el siguiente línea de código, ¿cuántas instancias de la clase `String` ves?

```
String cad="Ejemplo de cadena 1"; cad="Ejemplo de cadena 2"; cad=new String("Ejemplo de cadena 3");
```

Solución

Pues en realidad hay 4 instancias. La primera instancia es la que se crea con el literal de cadena "`Ejemplo de cadena 1`". El segundo literal, "`Ejemplo de cadena 2`", da lugar a otra instancia diferente a la anterior. El tercer literal, "`Ejemplo de cadena 3`", es también nuevamente otra instancia de `String` diferente. Y por último, al crear una nueva instancia de la clase `String` a través del operador `new`, se crea un nuevo objeto `String` copiando para ello el contenido de la cadena que se le pasa por parámetro, con lo que aquí tenemos la cuarta instancia del objeto `String` en solo una línea.