6.B. Cadenas de caracteres.

1. Cadenas de caracteres.

1.1. Operaciones avanzadas con cadenas de caracteres (I).

¿Qué operaciones puedes hacer con una cadena? Muchas más de las que te imaginas. Empezaremos con la operación más sencilla: la concatenación. La concatenación es la unión de dos cadenas, para formar una sola. En Java es muy sencillo, pues sólo tienes que utilizar el operador de concatenación (signo de suma):

```
String cad = "¡Bien"+"venido!";
```

```
System.out.println(cad);
```

En la operación anterior se esta creando una nueva cadena, resultado de unir dos cadenas: una cadena con el texto "`¡Bien`", y otra cadena con el texto "`venido!`". La segunda línea de código muestra por la salida estándar el resultado de la concatenación. El resultado de su ejecución será que aparecerá el texto "`¡Bienvenido!`" por la pantalla.

Otra forma de usar la concatenación, que ilustra que cada literal de cadena es a su vez una instancia de la clase `String`, es usando el método `concat` del objeto `String`:

```
String cad="¡Bien".concat("venido!");
```

```
System.out.printf(cad);
```

Fíjate bien en la expresión anterior, pues genera el mismo resultado que la primera opción y en ambas participan tres instancias de la clase `String`. Una instancia que contiene el texto "`¡Bien`", otra instancia que contiene el texto "`venido!`", y otra que contiene el texto "`¡Bienvenido!`". La tercera cadena se crea nueva al realizar la operación de concatenación, sin que las otras dos hayan desaparecido. Pero no te preocupes por las otras dos cadenas, pues se borrarán de memoria cuando el recolector de basura detecte que ya no se usan.

Fíjate además, que se puede invocar directamente un método de la clase `String`, posponiendo el método al literal de cadena. Esto es una señal de que al escribir un literal de cadena, se crea una instancia del objeto inmutable `String`.

Pero no solo podemos concatenar una cadena a otra cadena. Gracias al método `toString()` podemos concatenar cadenas con literales numéricos e instancias de otros objetos sin problemas.

El método `toString()` es un método disponible en todas las clases de Java. Su objetivo es simple, permitir la conversión de una instancia de clase en cadena de texto, de forma que se pueda convertir a texto el contenido de la instancia. Lo de convertir, no siempre es posible, hay clases fácilmente convertibles a texto, como es la clase `Integer`, por ejemplo, y otras que no se pueden convertir, y que el resultado de invocar el método `toString()` es información relativa a la instancia.

La gran ventaja de la concatenación es que el método toString() se invocará automáticamente, sin que tengamos que especificarlo, por ejemplo:

```
Integer i1=new Integer (1223); // La instancia i1 de la clase Integer contiene el número 1223.
```

```
System.out.println("Número: " + i1); // Se mostrará por pantalla el texto "Número: 1223"
```

En el ejemplo anterior, se ha invocado automáticamente i1.toString(), para convertir el número a cadena. Esto se realizará para cualquier instancia de clase concatenada, pero cuidado, como se ha dicho antes, no todas las clases se pueden convertir a cadenas.

Autoevaluación

¿Qué se mostrará como resultado de ejecutar el siguiente código `System.out.println(4+1+"-"+4+1);` ?

- ○ Mostrará la cadena "5-41".
- ○ Mostrará la cadena "41-14".
- ○ Esa operación dará error.