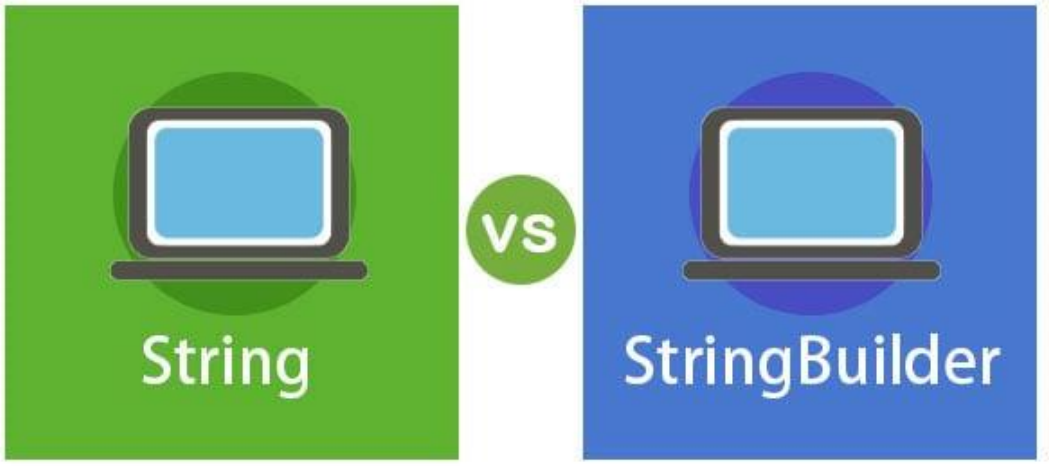
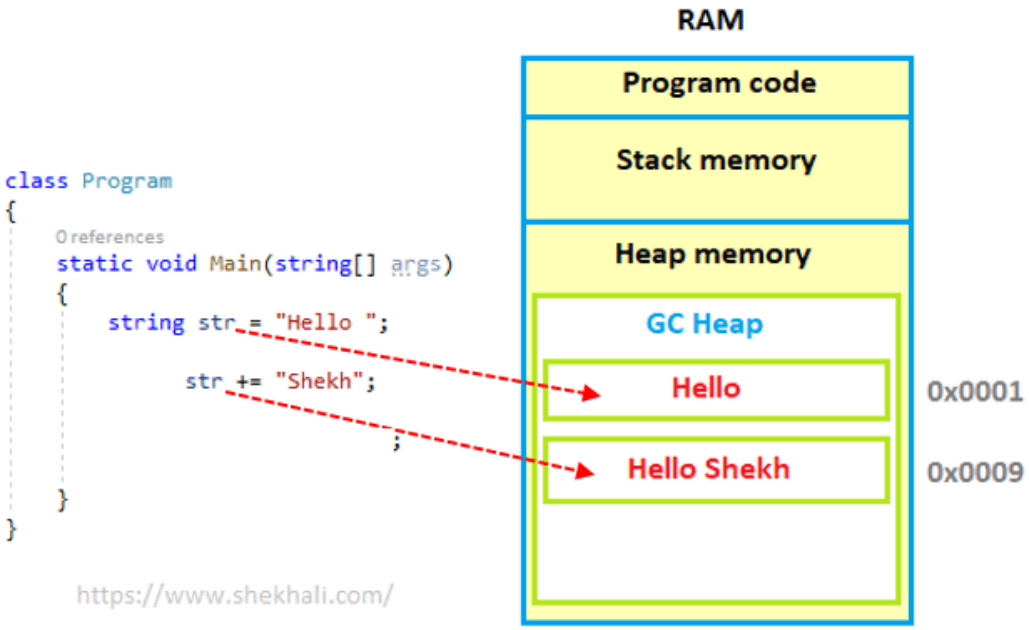
BONUS. MANIPULACIÓN DE CADENAS DE TEXTO CON StringBuilder



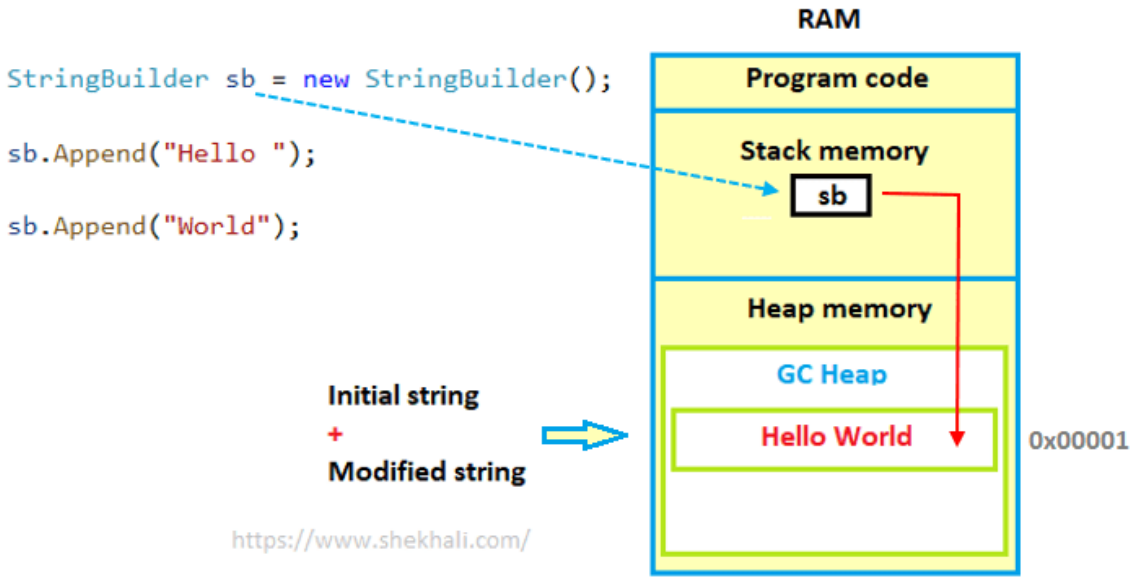
➔ ***String vs StringBuilder***

Un *StringBuilder* es una clase en algunos lenguajes de programación, como *Java* y *C#*, que se usa para manipular cadenas de texto de manera eficiente. A diferencia de los objetos *String*, que son inmutables (no pueden modificarse después de su creación), un *StringBuilder* permite modificar su contenido sin necesidad de crear nuevos objetos en memoria.

Cada vez que concatenamos un *String* con otro mediante el símbolo suma (+), lo que pasa realmente es que se crea un nuevo objeto en memoria, lo que puede ser ineficiente en términos de rendimiento si se hacen muchas modificaciones.



StringBuilder, en cambio, usa un buffer dinámico que permite modificar el texto sin crear nuevos objetos constantemente.



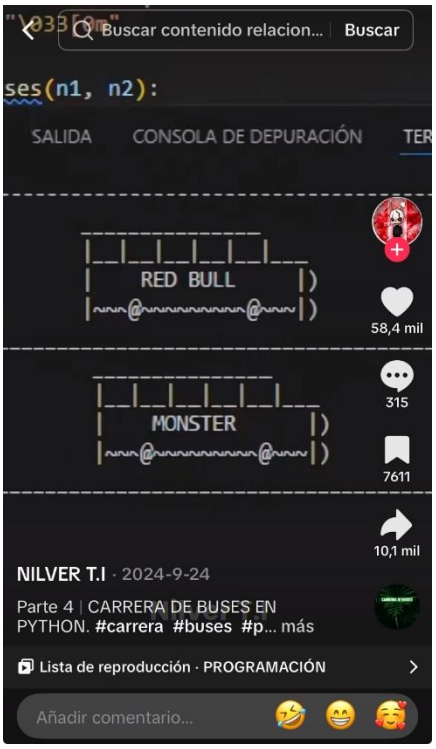
➔ **Métodos de *StringBuilder* para manipular cadenas**

- ***append(String s)***: inserta texto al final.
- ***insert(int index, String s)***: inserta texto en una posición específica.
- ***replace(int start, int end, String s)***: reemplaza una parte de la cadena.
- ***delete(int start, int end)***: elimina caracteres dentro de un rango.
- ***reverse()***: invierte la cadena.
- ***toString()***: convierte el StringBuilder en un String para poder imprimirlo.

Ejemplos de uso:

```
public class EjemploStringBuilder {
    public static void main(String[] args) {

        StringBuilder sb = new StringBuilder("Hola");
        sb.append(" Mundo");  // inserta " Mundo" al final
        sb.insert(5, " querido");  // inserta " querido" en la posición 5 (delante de
el que ya estaba en esa posición que es Mundo)
        sb.replace(5, 13, "gran "); // reemplaza por "gran" todo lo que haya desde la
posición 5 a la 13
        sb.delete(0, 5);  // borra los primeros 5 caracteres: "Hola "

        System.out.println(sb.toString());  // imprime "granoMundo"

        sb.reverse();
        System.out.println(sb.toString()); // imprime "odnuM narg"

    }
}
```

➔ **La famosa carrera de autobuses de *TikTok***



A continuación, se proporciona el código *Java* que implementa un *StringBuilder* para generar un autobús que avanza por la pantalla:

```
public class Bus {

    public static final int TAM = 97;

    public static void main(String[] args) throws InterruptedException {

        int a = 0; // POSICIÓN INICIAL DEL AUTOBÚS

        System.out.println("\n<<<<<<<<<< AUTOBUSITO >>>>>>>>>>");
        Thread.sleep(3000);

        while (a < TAM ) {

            a++; // avanzamos

            limpiarPantalla();

            if (a<TAM) {
                System.out.println(dibujarCarrera(a));
                Thread.sleep(70);
            }

        }


        System.out.println("\033[32m"+ "EL AUTOBUSITO HA LLEGADO A SU DESTINO!!" +
"\033[0m");


    }

    public static String dibujarCarrera(int n1) {
```

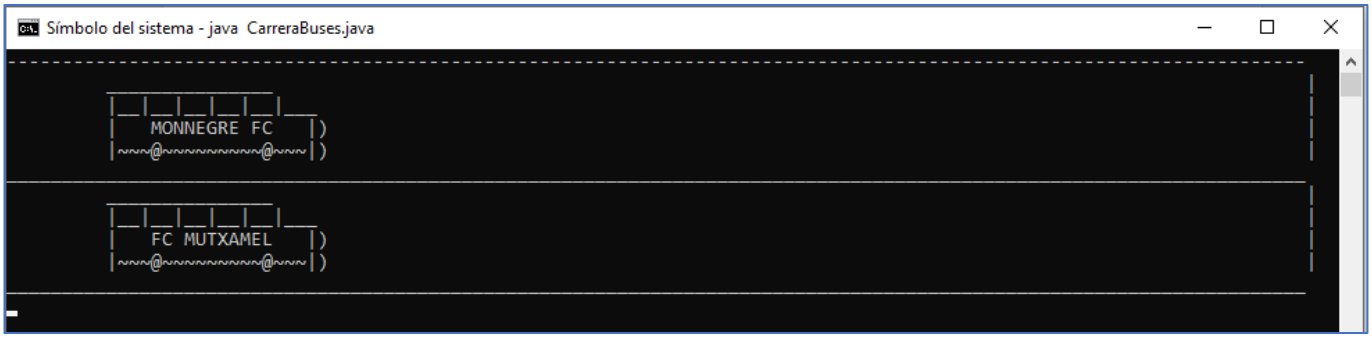
```
        StringBuilder sb = new StringBuilder();

        sb.append("-".repeat(117)).append("\n");
        sb.append(" ".repeat(n1)).append("_________________  ").append(" ".repeat(100 -
n1)).append("|\n");
        sb.append(" ".repeat(n1)).append("|__|__|__|__|__|___ ").append("
".repeat(TAM - n1)).append("|\n");
        sb.append(" ".repeat(n1)).append("|   IES MUTXAMEL  |)").append("
".repeat(TAM - n1)).append("|\n");
        sb.append(" ".repeat(n1)).append("|~~~@~~~~~~~~~@~~~|)").append("
".repeat(TAM - n1)).append("|\n");
        sb.append("_".repeat(117));

        return sb.toString();
    }

    public static void limpiarPantalla() {
        try {
            new ProcessBuilder("cmd", "/c", "cls").inheritIO().start().waitFor();
        } catch (Exception e) {
            System.out.print("\033[H\033[2J");
            System.out.flush();
        }
    }

}
```

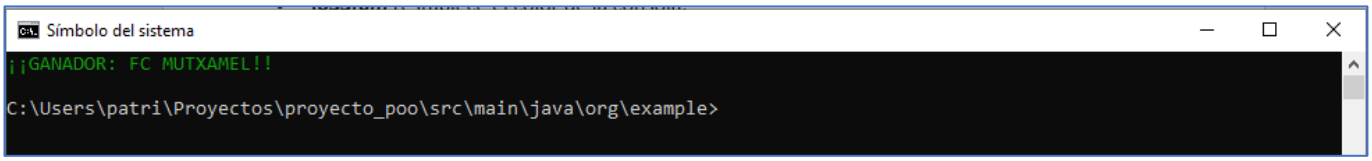
Explicación:

- ***a*** representa la posición del autobús.
- Se usa ***Thread.sleep(3000)*** para pausar 3 segundos, creando un efecto de animación.
- Mientras el autobús no llegue al final del tablero (***while (a < TAM)***), la ruta continúa avanzando.
- Se llama al método ***limpiarPantalla()***, que borra la pantalla de la consola para mostrar la nueva posición del autobús.
- Sólo dibuja la ruta (***if (a<TAM)***) si el autobús no ha llegado aún a su destino (*TAM*).
- El método ***dibujarCarrera(int n1)*** dibuja la pista y la posición del autobús.
    - « ***sb.append("-".repeat(117)).append("\n")*** dibuja una línea superior de 117 guiones.
    - « A la forma del autobús se añaden líneas con espacios ***(" ".repeat(n1))*** para posicionar el autobús según su avance.
    - « ***sb.append("_".repeat(117))*** dibuja la línea inferior con 117 barra bajas.
- Se imprime que ha llegado a su destino con color verde (***\033[32m***).
- ***\033[0m*** restablece el color de la consola.

ACTIVIDAD. Adapta el código dado para que se emule una carrera con dos autobuses como en el vídeo viral de *TikTok*.

```
C:\Users\patri\Proyectos\proyecto_poo\src\main\java\org\example>java CarreraBuses.java

<<<<<<<<<< CARRERA DE AUTOBUSES >>>>>>>>>>
    MONNEGRE FC vs FC MUTXAMEL
               FIGHT!
```





ENTREGA

PEGA EN AULES LA URL DE TU CLASE JAVA DISPONIBLE EN GITHUB.

EXPLICA TU CÓDIGO EN UN FICHERO MARKDOWN Y SÚBELO AL MISMO DIRECTORIO DE GITHUB QUE LA CLASE *JAVA*.