CURSO
BACKEND 1

Programación Orientada a Objetos

# Ejercicios extras

## CLASES DE UTILIDADES

egg | Argentina programa 4.0

# Ejercicios extras

Estos van a ser ejercicios para reforzar los conocimientos previamente vistos. Estos deben realizarse cuando hayas terminado la guía y tengas una buena base sobre lo que venimos trabajando. Además, si ya terminaste la guía y te queda tiempo libre, puedes continuar con estos ejercicios extra, recordando siempre que no es necesario que los termines para continuar con el tema siguiente. Por último, recuerda que la prioridad es ayudar a los compañeros de tu equipo y que cuando tengas que ayudar, lo más valioso es que puedas explicar el ejercicio con la intención de que tu compañero lo comprenda, y no sólo mostrarlo. ¡Muchas gracias!

1. Crea una clase en Java donde declares una variable de tipo array de Strings que contenga los doce meses del año, en minúsculas. A continuación, declara una variable mesSecreto de tipo String, y hazla igual a un elemento del array (por ejemplo, mesSecreto = mes[9]. El programa debe pedir al usuario que adivine el mes secreto. Si el usuario acierta mostrar un mensaje, y si no lo hace, pedir que vuelva a intentar adivinar el mes secreto.  Un ejemplo de ejecución del programa podría ser este:

   Adivine el mes secreto. Introduzca el nombre del mes en minúsculas: febrero

   No ha acertado. Intente adivinarlo introduciendo otro mes: agosto

   ¡Ha acertado!

2. Juego Ahorcado: Crear una clase Ahorcado (como el juego), la cual deberá contener como atributos, un vector con la palabra a buscar, la cantidad de letras encontradas y la cantidad jugadas máximas que puede realizar el usuario. Definir los siguientes métodos en AhorcadoService:

   - **Metodo crearJuego():** le pide la palabra al usuario y cantidad de jugadas máxima. Con la palabra del usuario, pone la longitud de la palabra, como la longitud del vector. Después ingresa la palabra en el vector, letra por letra, quedando cada letra de la palabra en un índice del vector. Y también, guarda la cantidad de jugadas máximas y el valor que ingresó el usuario.

- **Método longitud():** muestra la longitud de la palabra que se debe encontrar. Nota: buscar como se usa el vector.length.

- **Método buscar(letra):** este método recibe una letra dada por el usuario y busca si la letra ingresada es parte de la palabra o no. También informará si la letra estaba o no.

- **Método encontradas(letra):** que reciba una letra ingresada por el usuario y muestre cuantas letras han sido encontradas y cuántas le faltan. Este método además deberá devolver true si la letra estaba y false si la letra no estaba, ya que, cada vez que se busque una letra que no esté, se le restará uno a sus oportunidades.

- **Método intentos():** para mostrar cuántas oportunidades le queda al jugador.

- **Método juego():** el método juego se encargará de llamar todos los métodos previamente mencionados e informará cuando el usuario descubra toda la palabra o se quede sin intentos. Este método se llamará en el main.

  **Un ejemplo de salida puede ser así:**

  Ingrese una letra:

  a

  Longitud de la palabra: 6

  Mensaje: La letra pertenece a la palabra

  Número de letras (encontradas, faltantes): (3,4)

  Número de oportunidades restantes: 4

  ---

  Ingrese una letra:

  z

  Longitud de la palabra: 6

  Mensaje: La letra no pertenece a la palabra

  Número de letras (encontradas, faltantes): (3,4)

  Número de oportunidades restantes: 3

---

Ingrese una letra:

b

Longitud de la palabra: 6

Mensaje: La letra no pertenece a la palabra

Número de letras (encontradas, faltantes): (4,3)

Número de oportunidades restantes: 2

---

Ingrese una letra:

u

Longitud de la palabra: 6

Mensaje: La letra no pertenece a la palabra

Número de letras (encontradas, faltantes): (4,3)

Número de oportunidades restantes: 1

---

Ingrese una letra:

q

Longitud de la palabra: 6

Mensaje: La letra no pertenece a la palabra

Mensaje: Lo sentimos, no hay más oportunidades