Actividad Entregable: Analizador de Texto

Objetivo: Diseñar y construir un programa complejo aplicando los principios de la programación modular (Unidad 3), haciendo un uso intensivo de funciones y procedimientos, y aplicando las técnicas de manipulación de `String` (Unidad 2).

Resultados de Aprendizaje (RAs) Trabajados:

- **RA2: Escribe y depura código, analizando y utilizando las estructuras de control del lenguaje.**
- **RA3: Desarrolla programas aplicando la programación estructurada e introduciendo el tratamiento de datos.**

Enunciado

Has sido contratado para crear una herramienta de análisis de texto. El programa principal (`main`) debe ser simple: pedirá al usuario que introduzca un párrafo de texto y luego llamará a una única función que se encargará de analizarlo y mostrar un informe completo.

El desafío es **delegar cada cálculo individual a una función especializada**.

Estructura del Programa

Tu programa `AnalizadorTexto.java` debe contener las siguientes funciones. Presta atención a lo que debe hacer cada una y a lo que debe devolver

1. `main`

- Debe ser **simple**.
- Su única responsabilidad es:
  1. Crear un `Scanner`.
  2. Llamar a `mostrarBienvenida()`.
  3. Llamar a `pedirTexto()` para obtener el texto del usuario.
  4. Llamar a `analizarYMostrarResultados()` pasándole el texto.
  5. Cerrar el `Scanner`.

2. Funciones a Implementar

Aquí es donde ocurre la funcionalidad principal. Deberás crear:

A. `public static void mostrarBienvenida()`

- **Tipo:** Procedimiento.
- **Tarea:** Muestra un mensaje de bienvenida claro y las instrucciones del programa.

B. `public static String pedirTexto(Scanner teclado)`

- **Tipo:** Función.
- **Tarea:** Pide al usuario que "Introduzca un texto para analizar:" y devuelve el `String` que el usuario escriba.
- **Concepto clave:** Devuelve un tipo `String`.

C. `public static void analizarYMostrarResultados(String texto)`

- **Tipo:** Procedimiento.
- **Tarea:** Es la función que coordina el análisis. No calcula nada directamente, sino que **llama a todas las funciones auxiliares** (las siguientes) y usa sus resultados para imprimir un informe final.
- **Concepto clave:** Composición de funciones (una función que llama a

otras).

D. `public static int contarPalabras(String texto)`

- **Tipo:** Función.
- **Tarea:** Recibe el texto y devuelve el número total de palabras.
- **Pista:** Recorre el `String` carácter a carácter con un `for` (`charAt()`). Una palabra nueva comienza cuando el carácter actual **no** es un espacio (`' '`) y el carácter anterior **sí** lo era. (¡No olvides el caso especial de la primera palabra y ten cuidado con los espacios dobles!).
- **Concepto clave:** Manipulación de `String` (UD2) y retorno de `int`.

E. `public static int contarVocales(String texto)`

- **Tipo:** Función.
- **Tarea:** Recibe el texto y devuelve el número total de vocales (a, e, i, o, u), sin importar si son mayúsculas o minúsculas.
- **Pista:** Recorre el `String` con un `for`, usa `toLowerCase()` y `charAt()`, y comprueba cada carácter con un `switch`.
- **Concepto clave:** Recorrido de `String`, `switch`, acumulador.

F. `public static String encontrarPalabraMasLarga(String texto)`

- **Tipo:** Función.
- **Tarea:** Recibe el texto y encuentra y devuelve la palabra más larga. Si hay varias con la misma longitud máxima, devuelve la primera que encontró.
- **Pista:** Recorre el `String` carácter a carácter. Usa un `String` auxiliar (ej: `palabraActual`) para "construir" la palabra que estás leyendo. Cuando encuentres un espacio, significa que la palabra ha terminado. En ese momento, compara su longitud (`.length()`) con la longitud máxima que hayas encontrado hasta ahora. ¡No olvides comprobar la última palabra después de que termine el bucle!

- **Concepto clave:** Algorítmica de búsqueda, recorrido de `String`.

G. `public static int contarOcurrencias(String texto, char letra)`

- **Tipo:** Función.
- **Tarea:** Recibe el texto y una letra. Devuelve cuántas veces aparece esa letra en el texto (sin distinguir mayúsculas/minúsculas).
- **Concepto clave:** Paso de parámetros por valor (el `char`).

Ejemplo de Salida

```
--- BIENVENIDO AL ANALIZADOR DE TEXTO ---
Por favor, introduce el párrafo que deseas analizar.
El programa calculará el total de palabras, vocales,
la palabra más larga y cuántas veces aparece la letra 'a'.
-------------------------------------------------

Introduce un texto para analizar:
Java es un lenguaje de programacion potente y versatil

--- INFORME DEL TEXTO ---
Texto Analizado: "Java es un lenguaje de programacion potente
y versatil"
1. Total de Palabras: 8
2. Total de Vocales: 19
3. Palabra más Larga: "programacion"
4. Ocurrencias de la letra 'a': 5
---------------------------
```

Entrega

- Genera un fichero **Java** con el nombre `ud3_actividad_texto_[tu_nomb`

`re].java`.

- Asegúrate de que tu `main` esté limpio y que cada función cumpla con su responsabilidad única.