Examen de Programación Java

Instrucciones:
El examen consta de tres ejercicios: uno de cadenas (Strings), otro de arrays y uno de matrices. No se permite el uso de estructuras dinámicas como Listas ni métodos avanzados de la API de Java. Comenta el código donde creas que debes hacer aclaraciones, usa métodos para modular el código, y controla las posibles excepciones cuando lo consideres necesario.

Ejercicio 1: Strings

Cifrado de Vocales

Realiza una función que reciba como parámetro una cadena de texto y devuelva esa cadena con las vocales cifradas según el siguiente patrón:

Vocal	Sustituto
a/A	4
e/E	3
i/I	1
o/O	0
u/U	9

Además, debe invertir el orden de cada palabra (pero no el orden de las palabras en la frase).
Ejemplo:
Entrada: `Hola mundo`
Transformación 1: `H0l4 m9nd0`
Salida final: `4l0H 0dn9m`

Restricciones:

- No se pueden usar métodos `replace`, `replaceAll` de String ni `StringBuffer`.
- Se recomienda usar una estructura mutable (`StringBuilder` o similar) para hacer las transformaciones de manera eficiente.

Ejemplo adicional:
Entrada: `Buenos dias Eva`

Salida esperada: `s3n39B s41d 4v3`

Ejercicio 2: Arrays

Simulador de Ruleta

Escriba un programa que simule una ruleta simplificada con números del 0 al 36. El programa debe:

1. Generar tiradas aleatorias de la ruleta hasta que salga el número 0 (la banca gana).
2. Mostrar cada número que sale y clasificarlo como:
   - **ROJO**: Números impares del 1 al 10 y del 19 al 28, y números pares del 11 al 18 y del 29 al 36
   - **NEGRO**: El resto de números excepto el 0
   - **VERDE**: El número 0
3. Contar cuántas tiradas se realizaron hasta que salió el 0.
4. Mostrar la suma total de todos los números que salieron.

Después, escribe una función llamada `int[] contarColores(int n)` que simule n tiradas de ruleta y devuelva un array con tres posiciones: `[cantidad_rojos, cantidad_negros, cantidad_verdes]`.
Prueba el método con `n = 50` y `n = 200`.

Ejercicio 3: Matrices

Batalla Naval Simplificada

Crea un juego de batalla naval para consola en un tablero de 8x8. El programa debe realizar lo siguiente:

Inicialización:

- Crear un tablero de 8x8 (matriz de `char`), inicialmente relleno de `'~'` (agua).
- Colocar aleatoriamente:
  - 3 barcos pequeños de tamaño 2 (dos casillas consecutivas en horizontal o vertical)
  - Los barcos estarán representados con '`B`'
  - No pueden superponerse ni salirse del tablero

Juego:

- El jugador tiene 20 disparos para hundir todos los barcos
- En cada turno, el jugador introduce dos coordenadas (fila y columna)
- El programa responde:
  - “¡TOCADO!” si hay barco en la celda (marca con '`X`')
  - “Agua” si no hay barco (marca con '`O`')
  - “Ya disparaste ahí” si esa casilla ya ha sido jugada
- Mostrar el tablero después de cada disparo (sin revelar los barcos no descubiertos)

Finalización:

- Si se hunden todos los barcos: victoria
- Si se agotan los 20 disparos: derrota

Requerimientos adicionales:

- Crea un método void `colocarBarco`(char[][] tablero, int tamano) que coloque un barco aleatorio de tamaño dado en el tablero.
- Crea un método `void mostrarTablero(char[][] tablero, boolean mostrarBarcos)` que muestre el tablero. Si `mostrarBarcos` es `true`, muestra los barcos con 'B'; si es `false`, muestra `'~'` en su lugar.
- Crea un método `boolean todosBarcosHundidos(char[][] tablero)` que compruebe si quedan barcos ('B') en el tablero.