TRABAJO EN CLASE - TEMA 07

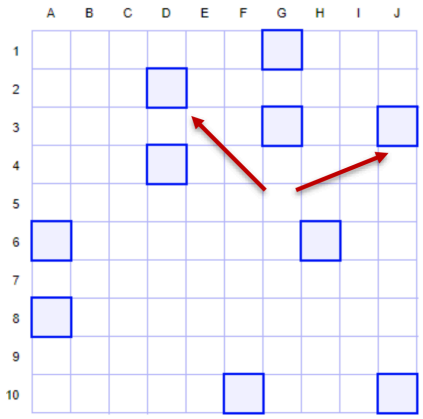
Ejercicio 1:

- Crear un diagrama de clases de UML (buscar herramientas libres para su elaboración)
- Un programa que cree 2 objetos de la clase Soldado. El usuario ingresa los valores por teclado y se muestran los datos en formato: "nombre, nivelVida => ColFila". Usando `toString()`

  NivelVida es aleatorio [1..5].

  `Soldado0, 5 => D2`

  `Soldado1, 3 => J3`



Clase Soldado.java:

```java
import java.util.*;
class Soldado {
    private String nombre;
    private int nivelVida;
    private char columna;
    private int fila;

    // Constructor
    public Soldado(String nombre, char columna, int fila) {
        this.nombre = nombre;
        this.columna = columna;
        this.fila = fila;
        this.nivelVida = generarNivelVida();
    }

    // Método para generar nivel de vida aleatorio entre 1 y 5
    private int generarNivelVida() {
        Random random = new Random();
        return random.nextInt(5) + 1;
    }

    // Método toString para mostrar los datos en el formato requerido
    @Override
    public String toString() {
        return nombre + ", " + nivelVida + " => " + columna + fila;
    }
}
```

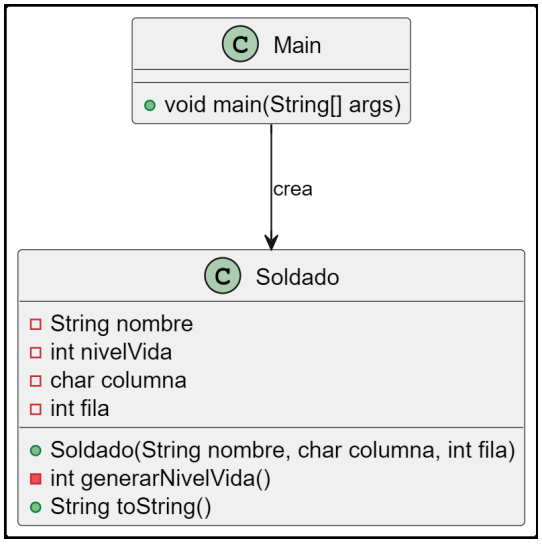
Clase Ejercicio1.java:

```java
import java.util.Scanner;

public class Ejercicio1 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        // Crear el primer soldado
        System.out.print("Ingrese el nombre del Soldado 1: ");
        String nombre1 = sc.nextLine();
        System.out.print("Ingrese la columna (A-J) del Soldado 1: ");
        char columna1 = sc.next().charAt(0);
        System.out.print("Ingrese la fila (1-10) del Soldado 1: ");
        int fila1 = sc.nextInt();
        Soldado soldado1 = new Soldado(nombre1, columna1, fila1);

        // Crear el segundo soldado
        System.out.print("Ingrese el nombre del Soldado 2: ");
        sc.nextLine(); // Esta linea limpia el buffer
        String nombre2 = sc.nextLine();
        System.out.print("Ingrese la columna (A-J) del Soldado 2: ");
        char columna2 = sc.next().charAt(0);
        System.out.print("Ingrese la fila (1-10) del Soldado 2: ");
        int fila2 = sc.nextInt();
        Soldado soldado2 = new Soldado(nombre2, columna2, fila2);

        // Mostrar los resultados
        System.out.println(soldado1);
        System.out.println(soldado2);
    }
}
```

Consola:

```
  '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\jhona\AppData\Roaming\Code\User\workspaceS
  Ejercicio1'
● Ingrese el nombre del Soldado 1: Benjamin
  Ingrese la columna (A-J) del Soldado 1: C
  Ingrese la fila (1-10) del Soldado 1: 3
  Ingrese el nombre del Soldado 2: José
  Ingrese la columna (A-J) del Soldado 2: J
  Ingrese la fila (1-10) del Soldado 2: 8
  Benjamin, 2 => C3
  Jos?, 4 => J8
```

Diagrama UML:



Ejercicio 2:

- Implementar un videojuego (iterativo) en modo gráfico.
- Crear diagrama de clases y código.
- Se debe simular una guerra entre 2 reinos (ingresar sus nombres por teclado).
- La guerra está compuesta de batallas (la cantidad será aleatoria entre 5 y 20).
- Cada batalla se da entre 2 soldados, uno por cada reino. Los tipos de soldado son Arquero, Espadachín y Caballero, los cuales tienen cierto nivel de vida. Para cada batalla se generará aleatoriamente el tipo de soldado y su nivel de vida según la siguiente tabla.
- Para determinar al ganador de la batalla se debe considerar las probabilidades (balance de poder) proporcionales al nivel de vida de cada soldado.

Ejemplo:

R1: Arquero 5 vs R2: Espadachín 5 -> probabilidades son 50% para ambos
R1: Arquero 4 vs R2: Caballero 8 -> probabilidades son 33,33% y 66,66%
La elección del ganador de la batalla es aleatoria pero de acuerdo a las probabilidades generadas. Cuando acabe la guerra se deberá mostrar el marcador final de las batallas ganadas y qué reino ganó la guerra (se permite empates).
Al acabar una guerra el videojuego deberá permitir empezar otra desde cero.