Programación II
Práctica 02c: Tipos Abstractos de Datos (TAD)
Monopoly

Esta práctica es parte de las actividades a distancia de la materia y se debe complementar con el video explicativo que se indica en la Hoja de Ruta de TADs.

Ejercicio de diseño de TADs - Abstracción: Juego Monopolio

Se desea modelar el clásico juego Monopolio, pero con la cantidad de casilleros definida por el usuario. Para ello se tiene un tablero de **n** posiciones y **2** jugadores.

Instancia: para n = 40

Sistema de juego simplificado:

Se pide por única vez el tamaño del tablero(n), y los precios de cada casilla.

En cada turno cada jugador tira un dado (un número de 1 a 5).

El tablero es circular, de manera que cuando se llega al final, se vuelva a dar otra vuelta.

Cada jugador comienza con $1000.

Cuando el jugador cae en una casilla, pueden pasar dos cosas:

1) Que la casilla no se haya comprado aun. En ese caso el jugador está obligado a comprarla y de ahí en más la casilla pertenecerá a dicho jugador
2) Que la casilla se encuentre comprada.
   Si la casilla pertenece al jugador que compro la casilla no se hace nada.
   Si la casilla pertenece al otro jugador, se debe pagar un alquiler, igual al 1% del monto de compra de dicha casilla.

El juego termina cuando alguno de los dos jugadores se queda sin dinero.

Esto puede suceder por alguno de los siguientes motivos:

1) El jugador no puede comprar una casilla.
2) El jugador no puede pagar el alquiler.

Diseño:

Diseñar e implementar el TAD Juego de “Monopolio” (Mono)
Es obligatorio que el TAD Mono utilice al menos otro TAD como TAD soporte.
Es decir:
-Que Mono no podrá estar implementado únicamente sobre los tipos primitivos de Java (ni java.util).
-Que Mono utilice varias clases para ser implementado.

Ayuda: Ver cuáles de las siguientes clases son necesarias para el TAD Mono, cuáles no, y por qué:
-Jugador
-Dado
-Tablero
-Reglamento
-Mono

Interfaz obligatoria:

```
Mono(n)                  //tamaño del tablero, donde n > 1
String ganador()         //devuelve el nro de jugador ganador o 0 si no hay
                         ganador por el momento
Void agregarCasilla(int casilla,double precio)
                         //Asigna el precio a la casilla
void jugar()             // tira los dos dados
String ver()             // muestra el estado del tablero y los jugadores
```

Ejemplo del código principal o código cliente de MONO

```
Mono mono = new Mono(6);            // Instancia: n == 6

Mono.agregarCasilla(0,100);   //La Casilla 0 vale $100
Mono.agregarCasilla(3,10);    //La Casilla 3 vale $10
while mono.ganador() == “”{         //como máximo hacerlo 1000 veces
      mono.jugar();                 // Los 2 jugadores “tiran” los dados
      System.out.println(mono.ver())  //Se muestra un resumen  del tablero
}
System.out.println(mono.ganador());
```

Implementación

Implementar un test que al menos pruebe el ejemplo del código principal.