INSTITUTO de TECNOLOGIA O.R.T.
Carrera: **ANALISTA de SISTEMAS**

TALLER DE PROGRAMACIÓN 1 **Final** **Tema 1** **15/07/2019**

Leé el enunciado con cuidado y por lo menos dos veces para entender qué se pide y qué estructuras son necesarias para resolver lo pedido basàndote en el diseño UML incompleto que figura más abajo y el proyecto entregado junto con este enunciado.

Pensá bien la estrategia de resolución antes de comenzar el desarrollo. El objetivo de este examen es **evaluar la correcta aplicación de los conceptos y el dominio de las técnicas** vistos durante la cursada:

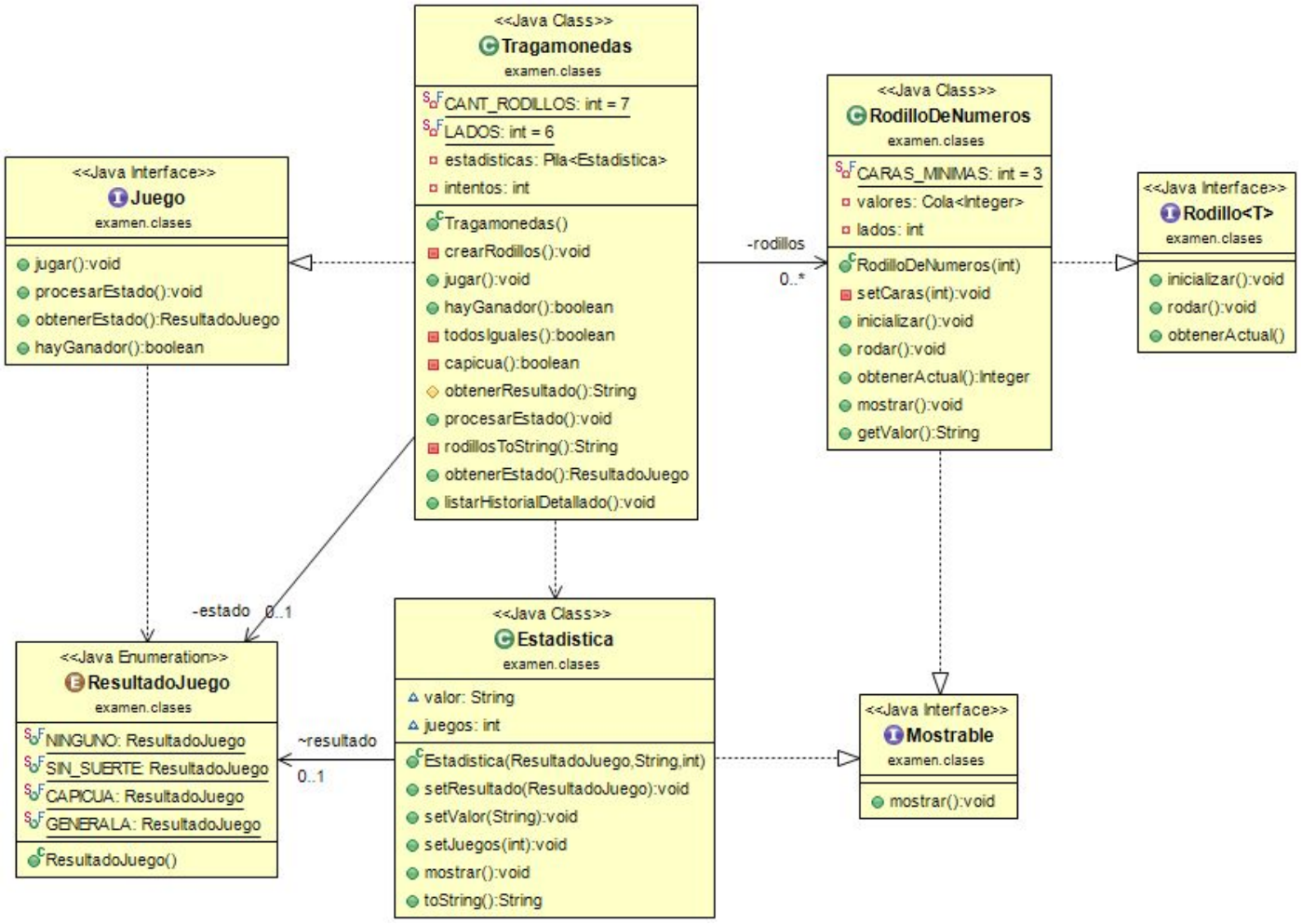
- Uso adecuado de excepciones (lanzamiento y captura) en la creación de objetos y sus validaciones.
- Correcto uso de Arrays, Pilas y/o Colas, siempre que se necesiten y aprovechando las interfaces e implementaciones provistas.
- Correcto uso de estructuras combinadas.
- Dominio de la herramienta utilizada durante el cuatrimestre para el desarrollo de proyectos Java (Eclipse), en especial lo referido a la importación y exportación de librerías y proyectos.

La mala aplicación de cualquiera de estos puntos hará que el examen sea desaprobado.
No deben modificarse las estructuras ni los métodos no mencionados en lo requerido. Sólo deben usarse.
La entrega del proyecto con warnings será penalizada.

Enunciado

La empresa para la que trabajamos está comenzando el desarrollo de una línea de máquinas tragamonedas de distinto tipo. Como todavía está en la etapa experimental nos solicitan la creación de un prototipo que permita probar estadísticamente cuántos lanzamientos son necesarios en promedio para obtener algún premio.

El diseño del prototipo es el que figura a continuación:



Cada tragamonedas tiene una serie de rodillos (un array cuyo tamaño se define al crear el tragamonedas), Cada rodillo está compuesto, principalmente, por una cola circular que puede ser de números, caracteres u otros símbolos, dependiendo del modelo de tragamonedas. La cantidad de caras de cada rodillo (la cantidad de elementos de la cola circular) se define al crear el rodillo.

Cada vez que se lanza una jugada, el tragamonedas le indica a cada rodillo que ruede, y cada uno de estos cambia su cara visible al azar (entre uno a la cantidad de lados del rodillo, inclusive). Una vez que todos los rodillos terminaron de rodar, el tragamonedas se fija si se da alguna combinación que dé premio. Como primera prueba se evalúa las condiciones: “todos iguales”, donde todos los rodillos tienen el mismo valor, y “capicúa”, donde el primero tiene el mismo valor que el último, el segundo tiene el mismo valor que el penúltimo y lo mismo con el resto si hubiese más rodillos que evaluar.

Por último, cada elemento de la estadística se forma ejecutando lanzamientos hasta que se obtenga alguna de las dos combinaciones “premiables” (“todos iguales” o “capicúa”). Además, esta información se apila para acceder automáticamente a la última serie procesada, pero al momento de listarla habrá que mostrarla en orden cronológico ascendente.

Tomando en cuenta esta información, completá la implementación desarrollando los siguientes puntos:

1. Implementar completa la clase **RodilloDeNumeros** de forma tal que coincida con el diseño del diagrama. Debe controlar que la cantidad de caras pedidas al menos alcance al mínimo posible (CARAS_MINIMAS). Recordá que el método rodar() mueve una cantidad de elementos al azar (entre 1 y la cantidad de caras del rodillo.
2. Completar la declaración e Implementar el constructor de la clase **Tragamonedas** (debe usar el método a implementar *crearRodillos()*).
3. Corregir la visibilidad de los atributos en la clase **Estadistica**.
4. Implementar o completar los siguientes métodos de **Tragamonedas**:
    a. *crearRodillos()*: Debe crear la cantidad de rodillos necesaria para este tragamonedas indicándole a cada uno la cantidad de caras que tendrá.
    b. *todosIguales()* y *capicua()*, cuyas reglas se explicaron más arriba.
    c. *procesarEstado()*: Resta completar lo referido al procesamiento de la estadística cuando se obtenga un resultado distinto a SIN_SUERTE: La muestra tiene el Resultado de la jugada ganadora, el valor combinado de los rodillos de la jugada ganadora (*rodillosToString()*) y la cantidad de lanzamientos necesarios para alcanzar esta jugada. Este contador debe ponerse en cero luego de guardar la muestra estadística. Debe tenerse en cuenta que la creación del registro de la muestra estadística (lo que se quiere apilar) puede generar una excepción.
    d. listarHistorialDetallado(): debe listar la información estadística en orden cronológico natural (ascendente)

Al terminar exportá el proyecto quitando los binarios y subilo al aula virtual. **Todo proyecto exportado que no pueda ser importado como se explicó en la materia será automáticamente reprobado**.