ESCUELA COLOMBIANA DE INGENIERÍA
PROGRAMACIÓN ORIENTADA A OBJETOS
Parcial segundo tercio. Nota esperada: _____
S11: 2020-02

ECIJudge

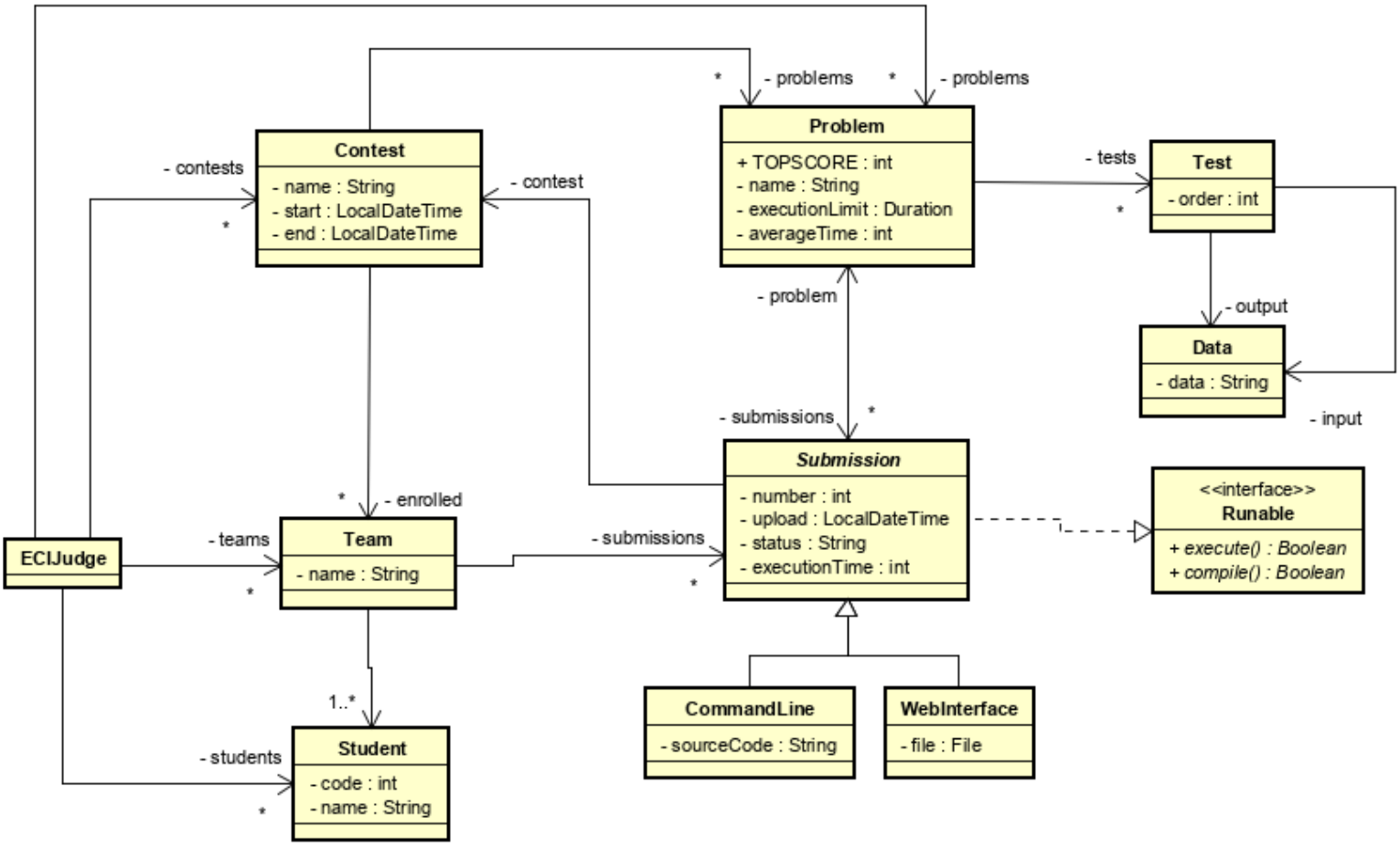
La decanatura de Ingeniería de Sistemas de la ECI quiere extender el sistema de competencias de programación. **EciJude** debe permitir ahora:

1. Los equipos pueden realizar sus envíos por alguno de los dos mecanismos: Línea de comandos o interfaz web. El envío por interfaz web tiene un tiempo de ejecución mayor al de línea de comandos. Para el envío de las soluciones se debe tener en cuenta:
   - Un equipo no puede enviar una solución para un problema si ya tiene un envío pendiente por revisar.
   - Si el tiempo de ejecución del envío es mayor al tiempo máximo de ejecución del problema, se genera una excepción por TIMEOUT y el envío queda rechazado.
2. El criterio de puntuación de los equipos cambia a la siguiente manera:
   - Si el equipo es de dos o más estudiantes:
     i. Máximo puntaje: Cada problema resuelto suma el máximo puntaje (TOPSCORE).
   - Si el equipo es de un estudiante:
     ii. Máximo puntaje: Cada problema resuelto suma el máximo puntaje (TOPSCORE) + 5 puntos.
3. La puntuación total de cada equipo en cada problema se calcula de la siguiente manera:

   $$puntajeTotal = TS + AT + CS$$

   Donde:
   - TS: Máximo puntaje (depende del número de integrantes de cada equipo).
   - AT: Si el tiempo de ejecución de la solución es menor al tiempo promedio del problema, suma 5 puntos.
   - CS: Si el problema es de categoría *complejos,* suma 5 puntos.

(Todos los contenedores son **HashMap**)



I. (25%) DISEÑANDO

Presente el diseño completo (No olvide MDD) para el método que evalúa (compilación y ejecución del código) el envío de una solución por medio de línea de comando. No olvide cambiar el estado del envío.

MDD
1. Escriba la especificación (documentación + encabezado) del método
2. Construya el(los) diagrama(s) de secuencia (adicione el manejo de excepciones con otro color)
3. Actualice el diagrama de clases con los nuevos elementos
4. Escriba el código de la clase responsable inicial (encabezado y atributos) y escriba únicamente el código del método especificado.

II. (30%) IMPLEMENTANDO

Implementen el siguiente método.

MDD
1. Estudie la especificación (documentación + encabezado) del método y las características de ECIJudge
2. Realice el diagrama de secuencia (adicione el manejo de excepciones con otro color)
3. Actualice el diagrama de clases con los nuevos elementos
4. Escriba el código completo. Adicione el manejo de excepciones con otro color.

En ECIJudge

public List<String> getFinalScoreBoard(**String** contestName) **throws** ECIJudgeException

Calculates the final contest scoreboard.

Parameters:

name – The contest's name

Throws:

ECIJudgeException - PENDING_SUBMMISIONS If there are some pending submissions.
ECIJudgeException - ALL_SUBMMISIONS_REJECTED If all submissions are rejected.
ECIJudgeException - CONTEST_IN_PROGRESS If the current date and time is before contest end.

Return:

Teams or students names ordered.

Precondiciones:

Los equipos se muestran teniendo en cuenta el siguiente orden

- Máximo puntaje (Punto 3 del enunciado).
- Si hay equipos empatados por máximo puntaje, se ordena según el número de problemas solucionados.
- Si hay equipos empatados por el número de problemas solucionados, se ordena por el menor tiempo de ejecución

III. (25%) EXTENDIENDO

Adicional a las modificaciones enunciadas previamente, la Decanatura ha solicitado las siguientes extensiones:

1. Los participantes pueden elegir el lenguaje de programación en los cuales presentarán sus soluciones. La decanatura ha decidido utilizar: Java, Python y C. De acuerdo al tipo de lenguaje, el tiempo límite para cada solución varía.
2. Los problemas estarán clasificados en dos categorías:
    a. Básico: Son problemas con temáticas consideradas como conocimientos requeridos para participar. Se espera que todos los participantes dominen el contenido.
    b. Complejos: Son problemas con temáticas más difíciles, los cuales requieren una documentación adicional al enunciado para ser comprendidos por los participantes.

Considerando las extensiones mencionadas, seleccione una de ellas teniendo en cuenta los siguientes criterios:

- Implica utilizar el concepto de herencia o interfaz
- Adición de mínimo 2 clases o puede mejorar el diagrama de clases que se muestra en el enunciado.

Una vez haya seleccionado la extensión, realice:

- Realice los cambios necesarios en el diagrama de clases.
- Proponga un método involucrando:
    - Excepciones: Generar mínimo 1 excepción.
    - Mínimo un nivel de secuencia.
    - BONO: Se dará bono extra si reutiliza alguno de los métodos de las partes I (Diseñando) o II (Implementando).
    - NO OLVIDE MDD.
- Realice el diagrama de secuencia del método propuesto (Adicione el manejo de excepciones con otro color).

IV. (20%) CONCEPTOS

1. ¿Cuáles son los tres momentos de una excepción? Explique cada uno.
2. ¿Qué es el bloque *finally* en excepciones?
3. ¿Qué es MDD?
4. ¿Qué indica que una clase sea *final?*
5. ¿Con cuál criterio podría decidir si utilizar una interfaz o una clase abstracta? (Puede explicar mediante un ejemplo)