**Clase: Torneo**

Es la clase encargada de administrar los torneos teniendo en cuenta arbitro, participantes y podios. Tiene de 5 métodos.

**Torneo**: Este es el método constructor de torneos.

**getLanzadores**: Devuelve una lista de participantes.

**getPodioConsistencia**: Devuelve el podio por consistencia

**getPodioDistancia**: Devuelve el podio por distancia.

**generarPodio**: Para cada podio se clasifican los participantes según los lanzamientos y de acuerdo con las correcciones del árbitro. De esta forma se generan los podios en el torneo.

**Clase: Arbitro**

Es una de las clases principales del programa ya que cumple un papel fundamental para el torneo, evaluando los lanzamientos de acuerdo a un criterio.

Esta clase tiene 8 métodos, un método constructor **Arbitro** que recibe un criterio.

**esTiroBueno** y **esTiroMalo**: Ambos métodos clasifican a un lanzamiento de acuerdo con el ángulo que obtuvieron en bueno o malo cumpliendo con las especificaciones del enunciado.

**compare** y **calcular**: en este caso, delega la acción de calcular y comparar al criterio de evaluación.

**corregirLanzamientos**: Se encarga de modificar los valores de la distancia de los lanzamientos dependiendo del ángulo de estos.

**validar**: Se encarga de validar los lanzamientos del participante según el criterio de evaluación.

**cambiarCriterioDeEvaluacion:** se usa para cambiar de criterio de evaluación. Utiliza el patrón Strategy, con una cola circular, que cambia al siguiente criterio de evaluación.

**Clase: Podio**

Esta clase tiene un método constructor **Podio** que recibe la cantidad de escalones que va a tener el mismo.

Tiene dos métodos del tipo getter, **getGanadores** y **getEscalones** que devuelven en cada caso una lista de ganadores y los escalones que tiene el podio respectivamente.

Tiene también un método llamado **clasificarParticipante** que se encarga de cargar los escalones del podio de acuerdo con las comparaciones del árbitro.

**Clase: Participante**

Clase encargada de modelas a los participantes. Consta de un método constructor **Participante** el cual recibe el número de participante y la lista de lanzamientos.

Tiene su propia implementación del método **toString ()** y cuenta con dos métodos **getLanzamientos** y **getNumeroParticipante** que devuelven la lista de lanzamientos y el número de participante respectivamente.

**Clase: Lanzamiento**

Esta clase modela los lanzamientos de los participantes, consta de seis métodos.

El método **Lanzamiento** para construir un objeto de esta clase que recibe la distancia y el ángulo.

Tiene también los getters y setters de estos atributos que son **getDistancia**, **setDistancia**, **getAngulo**, **setAngulo**, **setValidez** y **getValidez**.

**Clase: CriterioDeEvaluacion**

Esta es una interface la cual tiene dos método:

**Calcular**: Básicamente es un método que permite realizar cálculos sobre una lista de lanzamientos.

**Validar:** se usa para validar si un participante puede o no clasificar para un podio.

Esta clase extiende de la interfaz Comparator, por lo cual las clases que la implementen deben sobre escribir el **compare**.

**Clase: Consistencia**

Implementa la interfaz CriterioDeEvaluacion realizando su propia implementación para los métodos **calcular, validar** y **compare.** En este caso calcula y compara lanzamientos determinando la consistencia de este.

**Validar:** se va a asegurar de que no tenga ningún lanzamiento inválido.

**Clase: Distancia**

Implementa la interfaz CriterioDeEvaluacion realizando su propia implementación para los métodos **calcular** y **compare.** En este caso calcula y compara lanzamientos determinando la suma total de las distancias de estos.

**Validar:** se va a asegurar de que no tenga todos los lanzamientos inválidos.