Andre Marroquin, 22266

Se identifican correctamente las modificaciones que es necesario hacerle a cada una de las clases.

Los cambios que se van a llevar en las clases únicamente es en la clase torneo, donde se va a hacer solo 1 lista en vez de una para cada tipo de jugador

Se explican las modificaciones a realizar en los atributos y métodos.

Las modificaciones que tiene son en la clase principal y en la de torneo que se crea una única lista de jugadores. En donde los métodos se van a poner con casos para que tipo de jugador es y hacer el calculo de eficiencia.

Se considera el polimorfismo en la implementación.

El polimorfismo es utilizado cuando se eliminan las 3 listas anteriores y se crea solo una de ellas que es de tipo jugador.

Abajo análisis del eje 4

Requisitos Funcionales - Identifica correctamente todo lo que debe hacer el programa según la situación planteada.

El programa debe de mostrar los 3 mejores líberos del torneo, también debe de mostrar todos los jugadores del torneo y la cantidad de pasadores con más de un 80% de efectividad.

Identificación de Clases, atributos y métodos –

Se identifican correctamente las clases que se necesitan para resolver el problema. El número de clases identificadas es suficiente para darle solución a la situación planteada.

Se necesita la clase principal para la interacción con el usuario se va a necesitar la clase jugador, libero, pasador y auxiliares, y la clase torneo que será la lista de los competidores.

Se identifica correctamente la jerarquía de clases (herencia) que se presenta en el problema planteado.

La clase padre en este caso va a ser jugador de la cual sus hijos van a ser libero, pasador y auxiliar.

Se explica correctamente y de forma lógica el propósito de cada una de las clases.

La principal se va a utilizar para interactuar con el usuario. La clase torneo se va a utilizar para crear la lista de los competidores del torneo. La clase jugador es la clase padre con los atributos que todos los jugadores tienen, de ahí se hereda la clase libero que tiene atributos extra como recibos efectivos. De la clase jugador también se hereda la clase pasador la cual tiene los atributos extra de pases y jugadas

de engaño efectivas. La última clase que es hijo de jugador es la de auxiliares que tiene como atributos extra los ataques y bloqueos efectivos y bloqueos fallidos.

Atributos de las clases

Se identifican correctamente todos los atributos de cada una de las clases seleccionadas. Son los necesarios para resolver el problema planteado.

Clase principal -> main

Clase torneo -> array list tipo libero, array list tipo pasador, array list tipo auxiliar

Clase jugador -> nombre, país, errores, aces, total de servicios

Clase libero-> exntends clase jugador, recibos efectivos

Pasadores -> extends clase jugador, pases efectivos, fintas efectivas

Auxiliares -> extends clase jugador, ataques efectivos, bloqueos efectivos, bloqueos fallidos

Se identifican los atributos comunes que se ponen en la súper clase, no se repiten atributos en varias clases.

Atributos comunes-> nombre, país, errores, aces, total de servicios

Se identifican correctamente los atributos polimórficos.

No se hace dijo lynette

Se explica correctamente y de forma lógica el propósito de cada uno de los atributos.

Clase principal -> main (para interactuar con el usuario)

Clase torneo -> array list tipo pasador (para agregar pasadores al torneo), array list tipo auxiliar (para agregar pasadores al auxiliar). array list tipo libero (para agregar liberos al torneo)

Clase jugador -> nombre (registra el nombre de la persona) , país(registra el país de la persona), errores(registra los errores de la persona) , aces(registra los aces que hace la persona) , total de servicios(registra la cantidad de servicios que hace la persona)

Clase libero-> exntends clase jugador (agarra todos los atributos de la clase jugador) , recibos efectivos(registra los recibos efectivos que hace el libero)

Pasadores -> extends clase jugador (agarra todos los atributos de la clase jugador) , pases efectivos(registra los pases efectivos que hace el pasador) , fintas efectivas(registra las fintas efectivas que hace el pasador)

Andre Marroquin, 22266

Auxiliares -> extends clase jugador (agarra todos los atributos de la clase jugador) , ataques efectivos(registra los ataques efectivos del auxiliar) , bloqueos efectivos(registra los bloqueos efectivos del auxiliar) , bloqueos fallidos (registra los bloqueos fallidos del auxiliar)

Métodos de las clases

Se identifican correctamente los métodos necesarios para resolver la situación planteada.

Torneo-> crear listas , agregar jugador al torneo, mostrar jugadores del torneo, mostrar 3 mejores líberos, cantidad pasadores con más de 80% de efectividad.

Jugador-> constructor con parámetros, constructor sin parámetros.

Libero-> se llama constructor con parámetros de la clase jugador, se inicializan los atributos de la clase libero

Pasadores -> se llama constructor con parámetros de la clase jugador, se inicializan los atributos de la clase pasadores.

Auxiliares -> se llama constructor con parámetros de la clase jugador, se inicializan los atributos de la clase auxiliares

Queda claro cuáles son los métodos que se superponen (override) en la jerarquía de clases.

Los métodos que se superponen es el constructor con parámetros de la clase padre.

Se explica correctamente y de forma lógica el propósito de cada uno de los métodos de las clases.

Torneo-> crear lista (para que se puedan guardar los 3 tipos de jugadores), agregar jugador al torneo (para que la lista no sea vacía) , mostrar jugadores del torneo(para ensenar que jugadores están en el torneo) , mostrar 3 mejores líberos(calculo para demostrar los 3 mejores líberos) , cantidad pasadores con más de 80% de efectividad.(para demostrar los mejores pasadores arriba de un 80%)

Jugador-> constructor con parámetros (para que pueda llamarse la clase con sus atributos), constructor sin parámetros. (para darle valor inicial a los atributos)

Libero-> se llama constructor con parámetros de la clase jugador (para heredar los atributos de la clase padre), se inicializan los atributos de la clase libero. (para poder usar los atributos de la clase libero)

Pasadores -> se llama constructor con parámetros de la clase jugador (para heredar los atributos de la clase padre), se inicializan los atributos de la clase pasadores. (para poder usar los atributos de la lase pasadores)

Andre Marroquin, 22266

Auxiliares -> se llama constructor con parámetros de la clase jugador (para heredar los atributos de la clase padre), se inicializan los atributos de la clase auxiliares. (para poder usar los atributos de la clase auxiliares)