UNIVERSIDAD NACIONAL DE GENERAL

SARMIENTO

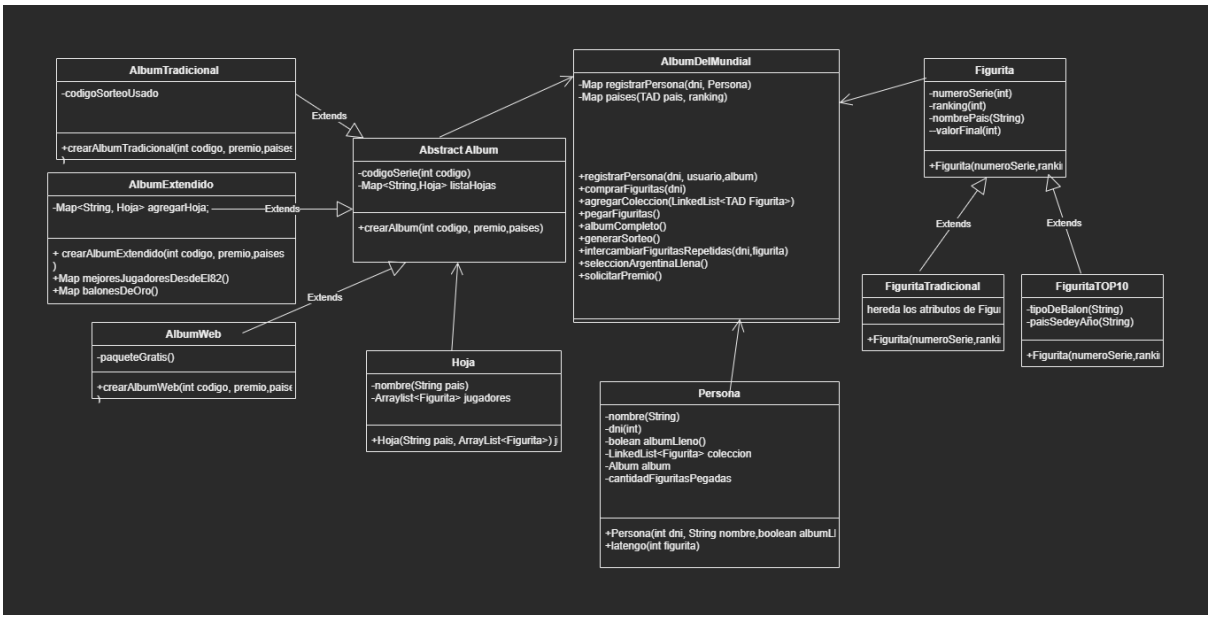
Programación II – 2do semestre de 2021

Trabajo Práctico:

Fecha de entrega límite: 05/11/2021
Comisión: COM-03 (Turno Mañana)
Docentes:Martin Pustilnik, Ximena Ebertz
Estudiante: Sosa Garcia Matias
RE ENTREGA

Introducción

Nos pidieron realizar en primera instancia un análisis del problema que se nos planteaba que era que nos pedían un Sistema que pueda administrar personas que poseen álbumes que pueden pegar figuritas, intercambiarlas y hacer sorteos. Para esto primero hicimos un diagrama de clases sobre los TADS que íbamos a usar y cómo se iban a relacionar entre sí, y qué métodos íbamos a usar para llevar esto a cabo. Luego en segunda instancia debemos implementar los métodos que nos habían dado como base en el tp, pudiendo crear métodos nuevos para combinarlos con estos y usar los atributos y métodos que nos proporciona la clase Fábrica. Debíamos pasar los Test correctamente, el código cliente y AlbumMundial



Desarrollo

TAD AlbumDelMundial

IREP:

Las personas no pueden repetirse y Los TAD persona tienen que estar en usuariosRegistrados.

Atributos:

private Map<Integer, Persona> usuariosRegistrados → clave: Integer que representa el dni de la persona, valor: TAD Persona, tiene los datos de la persona. Almacena los datos de las personas registradas en el sistema AlbumDelMundial

Operaciones:

AlbumDelMundial() → Constructor. Inicializa los atributos.

public int registrarParticipante(int dni, String nombre, String tipoAlbum) →
Crea una persona y llama al metodo tipoAlbum que crea un álbum y la añade a registrarParticipante siempre y cuando el dni no esté repetido en registrarParticipante.
El metodo tipoAlbum aplica justamente la herencia y el polimorfismo permiten agregar Álbumes sin necesitar un método para cada tipo

public void comprarFiguritas(int dni)→ Se generan 4 figuritas al azar y se asocia al participante correspondiente identificado por dni si el participante no está registrado, se debe lanzar una excepción.

public void comprarFiguritasTop10(int dni)→ Se generan 4 figuritas top 10 al azar y se asocia al participante correspondiente identificado por dni, si el participante no está registrado, se debe lanzar una excepción, si el participante no tiene album top10, se debe lanzar una excepción.

public void comprarFiguritasConCodigoPromocional(int dni)→ Compra por única vez un grupo de 4 figuritas con el codigo promocional asociado a su album web,si el participante no está registrado, se debe lanzar una excepción,si el participante no tiene codigo de sorteo o el mismo ya fué usado, se debe lanzar una excepción.

List<String> pegarFiguritas(int dni) → Busca entre las figuritas del participante cuales aún no están en el album y las asocia. Devuelve una lista con las figuritas asociadas. De cada figurita se devuelve un string "$pais-$numeroJugador", si el participante no está registrado, se debe lanzar una excepción.

boolean llenoAlbum(int dni) → Verifica si el participante identificado por dni ya completó el album. Devuelve true si está completo, sino false, este metodo debe resolverse en O(1), si el participante no está registrado, se debe lanzar una excepción.

String aplicarSorteoInstantaneo(int dni) → Realiza el sorteo instantáneo con el codigo asociado al album tradicional del participante y devuelve algún premio al azar. Si el participante no está registrado, se debe lanzar una excepción. Si no tiene codigo para el sorteo o ya fue sorteado, se debe lanzar una excepción.

int buscarFiguritaRepetida(int dni) Busca si el participante tiene alguna figurita repetida y devuelve el codigo de la primera que encuentre. Si no encuentra ninguna, devuelve el codigo -1. Si el participante no está registrado, se debe lanzar una excepción.

boolean intercambiar(int dni, int codFigurita) Dado el dni de un participante A y el codigo de una figurita, se busca entre los participantes con mismo tipo de album si

alguno tiene repetida alguna figurita que le falte al participante A con un valor menor o igual al de la figurita a cambiar. En caso de encontrar alguno, realiza el intercambio y devuelve true. Si no se encuentra ninguna para cambiar, devuelve false. Si el participante no está registrado o no es dueño de la figurita a cambiar, se debe lanzar una excepción.

boolean intercambiarUnaFiguritaRepetida(int dni) Dado el dni de un participante, busca una figurita repetida e intenta intercambierla. Devuelve true si pudo intercambiarla. Sino, devuelve false. Si el participante no está registrado, se debe lanzar una excepción.

String darNombre(int dni) Dado el dni de un participante, se devuelve el nombre del mismo. Si el participante no está registrado, se debe lanzar una excepción.

String darPremio(int dni) Dado el dni de un participante, devuelve el premio correspondiente por llenar el album. Si el participante no está registrado, se debe lanzar una excepcion. Si no tiene el album completo, se debe lanzar una excepcion.

String listadoDeGanadores() Devuelve un string con la lista de todos los participantes que tienen su album completo y el premio que les corresponde. El listado debe respetar el siguiente formato para cada ganador: " - ($dni) $nombre: $premio"

List<String> participantesQueCompletaronElPais(String nombrePais) Devuelve una lista con todos los participantes que llenaron el pais pasado por parametro. De cada participante se devuelve el siguiente String: "($dni) $nombre: $tipoAlbum"

TAD Album

Atributos:

protected Integer codigoSerie → Código único del album.

protected HashMap<String,Hoja>listaHojas → Almacena los paises que participan en el mundial, clave: pais, valor: Hoja

Operaciones:
Album(int codigoSerie,HashMap<String, Hoja> a) //Constructor. Inicializa los atributos.

public int get_id()//Devuelve la variable codigoSerie
public HashMap<String, Hoja> getlistaHojas()//Devuelve

IREP: Cada String tiene que tener una hoja, el codigo de serie debe ser unico.

TAD AlbumTradicional

Atributos:

private boolean codigoSorteoUsado //Booleano que almacena el valor del codigoDelSorteo

TAD AlbumExtendido

Atributos:

AlbumExtendido(int codigoSerie, HashMap<String, Hoja> a) //Constructor. Inicializa los atributos.

Operaciones

listaHojas.put("TOP10", new Hoja("TOP10",new ArrayList<Figurita>())) //Agrega una hoja mas para poder almacenar las figuritas TOP10

TAD AlbumWeb

Atributos:
private boolean paqueteGratis; ////Booleano que almacena el valor del paqueteGratis

Operaciones:

public AlbumWeb(boolean paqueteGratis,int codigoSerie, HashMap<String, Hoja> a) //Constructor. Inicializa los atributos.

TAD Fabrica

Operaciones:

public String[] getPremiosInstantaneos()// Devuelve un array de los premios que se pueden ganar los participantes

public Album tipoAlbum(String album)//Crea el tipo de album que se le pase por parámetro llamando al constructor correspondiente

public HashMap<String, Hoja>GenerarAlbum()// Genera las Hojas con sus correspondientes Array de jugadores

public FiguritaTradicional generarSobreTradicional()//Genera 1 Figurita Tradicional

public FiguritaTop10 generarSobreTopDiez()//Genera 1 Figurita TOP100

private String obtenerPais() //Devuelve un pais aleatorio de los 32 paises que van al mundial

TAD Figurita

IREP:

IREP: el número de serie no puede repetirse, no pueden repetirse los jugadores

Atributos:

private int numeroSerie// Código único de la figurita
private int ranking //Código único del ranking
private String pais // Nombre del país
private int valorFinal //Código único del valor final de la figurita

Operaciones:

Figurita(int numeroSerie,int ranking, String pais,int valorFinal)//Constructor. Inicializa los atributos.

TAD FiguritaTradicional

Operaciones

FiguritaTradicional(int numeroSerie, int ranking, String pais, int valorFinal)//Constructor. Inicializa los atributos.

TAD FiguritaTOP10

Atributos:

protected String paisSedeyAño // Nombre del pais y año

protected String balon; //// Nombre del tipo de balon que gano

Operaciones:

FiguritaTop10(int numeroSerie, int ranking, String pais, int valorFinal,String paisSedeyAño,String balon)//Constructor. Inicializa los atributos.

TAD Hoja

Atributos:

private String pais //// Nombre del pais
private ArrayList<Figurita> jugadores;//Array de TAD Figurita. Aca se puede observar el polimorfismo y herencia ya que podemos guardar cualquiera tipo Figurita dentro del mismo Array

Operaciones:

public Hoja(String string,ArrayList<Figurita> jugadores) //Constructor. Inicializa los atributos.

TAD Persona

IREP:

El nombre no puede ser vacio, el dni no nulo, el albumLleno debe tener true or false, debe tener una lista creada para las figuritas

Atributos:

private int dni;
private String nombre//String que representa el nombre
private boolean albumLleno //Boolean que representa si el album esta lleno
private LinkedList<Figurita> coleccion // LinkedList que almacena las Figurita que tiene la persona en posesión.
private Album album;// Almacena el tipo de album que tiene
private int cantFiguritasPegadas// Integer que representa la cantidad de figuritas pegadas en su album

Operaciones:

Persona(int dni, String nombre,boolean albumLleno,LinkedList<Figurita> coleccion,Album album,int cantFiguritasPegadas) //Constructor. Inicializa los atributos.

public boolean latengo (int figurita) //Se le pasa el Integer codigoUnico de una figurita a la persona y se fija si tiene esa figurita en su posesión.

public int getCantFiguritasPegadas() // devuelve la variable CantFiguritasPegadas

public void setCantFiguritasPegadas// Setea un nuevo valor de la variable CantFiguritasPegadas

public String getNombre()// Devuelve la variable nombre

public int getDni() //Devuelve la variable dni

public LinkedList<Figurita> getColeccion()// Devuelve la linkedList de coleccion

public Album getAlbum() //Devuelve el TAD Album

public boolean isAlbumLleno()//Devuelve el valor de AlbumLleno

Problemas encontrados y como fueron resueltos

-Como crear una clase Hoja que pueda relacionarse tanto con album como con las figuritas, para esto fue clave haber hecho correctamente un diagrama de clases ya que nos dejava bien claro que se necesitaba de un TAD intermediario para poder almacenar los paises y sus jugadores
-Como generar un paquete de figuritas, al principio ibamos a usar el metodo de fabrica que se le pasaba por parametro un numero pero no encontrábamos como crear un paquete pasandole por parametro algo, asi que creamos un método propio que crea una figurita y luego para crear el paquete, en la clase AlbumMundial hacer un for que itere 4 veces sobre una figurita, y asi crear 4 figuritas simulando ser un paquete
- Luego tambien tuvimos problemas porque no sabiamos como acceder a los metodos de las clases que eran hijas de la clase padre, hasta que hicimos lo de castear una clase en otra.

Conclusión

El trabajo practico sirvio para poder aplicar todo lo visto en clase, tanto de acumuladores booleanos, como el repaso de objetos sobre todo cuando tenes que usar metodos de instancia o de clase, lo que mas se aplico sobre el trabajo fue el tema de polimorfismo y herencia, el tema de castear una clase para poder usar sus metodos tambien fue necesario porque muchas veces habia que usar metodos propio de cada clase.

Metodos y Test no resueltos

Nos da error el test 10 y 18

Metodos no resueltos

boolean intercambiar(int dni, int codFigurita);

List<String> pegarFiguritas(int dni) // este metodo solo me deja ir pegando de a 4 figuritas en el album, no me deja seguir pegando las otras 8 que faltan, es como que va pisando los valores

Soluciones:
El test 18 le estaba pasando de primeras una excepción y esta tenia que ir al final

```
public String darPremio(int dni) {
    if(estaRegistrado(dni)) {
        if(usuariosRegistrados.get(dni).getAlbum() instanceof AlbumTradicional){
            if(usuariosRegistrados.get(dni).getAlbum().estaCompleto()==true) {
                return "Te ganaste una pelota";
            }

        }
        else if(usuariosRegistrados.get(dni).getAlbum() instanceof AlbumWeb) {
            if(usuariosRegistrados.get(dni).getAlbum().estaCompleto()==true) {
                return "Te ganaste una camiseta oficial de la selección";
            }

        }
        else if(usuariosRegistrados.get(dni).getAlbum() instanceof AlbumExtendido) {
            if(usuariosRegistrados.get(dni).getAlbum().estaCompleto()==true) {
                return "Te ganaste una pelota y un viaje";
            }

        }
    }

    throw new RuntimeException("El usuario no esta registrado");
}
```

```
public String darPremio(int dni) {
    if(!estaRegistrado(dni)) {
        throw new RuntimeException("El usuario no esta registrado");
    }
    if(usuariosRegistrados.get(dni).getAlbum() instanceof AlbumTradicional){
        if(usuariosRegistrados.get(dni).getAlbum().estaCompleto()==true) {
            return "Te ganaste una pelota";
        }

    }
    else if(usuariosRegistrados.get(dni).getAlbum() instanceof AlbumWeb) {
        if(usuariosRegistrados.get(dni).getAlbum().estaCompleto()==true) {
            return "Te ganaste una camiseta oficial de la selección";
        }

    }
    else if(usuariosRegistrados.get(dni).getAlbum() instanceof AlbumExtendido) {
        if(usuariosRegistrados.get(dni).getAlbum().estaCompleto()==true) {
            return "Te ganaste una pelota y un viaje";
        }

    }

    return "El album no esta lleno";
}
```

Para pegar figuritas lo que hice fue dividir responsabilidades para hacer funciones en cascadas, primero en la clase principal, llamar a un metodo pegar figurita que este en la clase album, luego llamar a un metodo que este en la clase hoja. Para poder ir modificando las instancias de cada uno