PROGRAMACIÓ 2: PRÀCTICA 2

Marina Noguera Mataró

Joan Florit Rodríguez

Programació 2

Pràctica 2

Data de lliurament: 16/04/2023

ÍNDEX

INTRODUCCIÓ:

L'objectiu de la pràctica és modelar i dissenyar un programa que serveixi per a la gestió d'un club social, els seus socis a través d'un menú. El programa ha de servir per gestionar els socis del club (amb les modificacions, consultes i verificacions corresponents), les activitats (amb les modificacions, consultes i verificacions corresponents), la factura mensual dels socis segons les seves característiques i l'emmagatzematge i recuperació de les dades en un fitxer per tal de no perdre-les d'una execució a una altra.

QÜESTIONS:

1. **Explicar les classes implementades.**

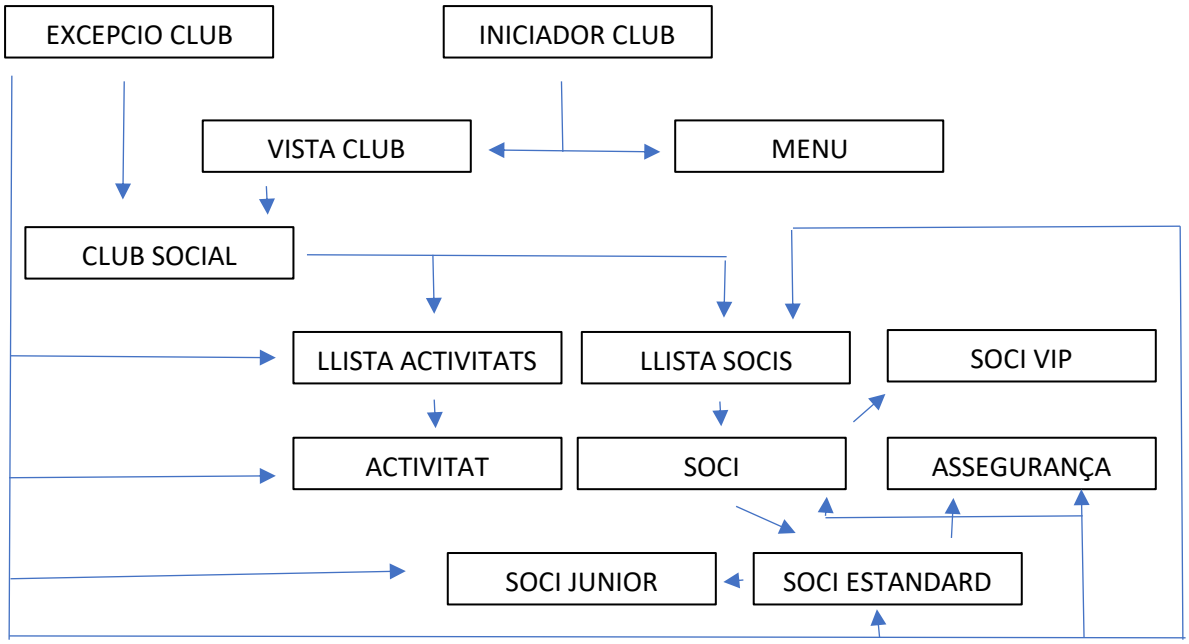
- Soci: la classe *soci* és la que guarda les dades que tenen en comú tots els socis (nom, dni, quota i llista d'activitats) i a més te els mètodes, implementats o no, que faran servir les seves tres subclasses (els getters i setters de tots els atributs, una funció per afegir una activitat a un soci determinat, una altra per calcular el preu d'una llista d'activitats, una altra per calcular la factura total d'un soci determinat, una per verificar les dades, una per imprimir-les, una que retorna quin tipus de soci és i, finalment, una funció per comprovar si dos socis són iguals a partir del dni).
- SociVIP: és una subclasse de S*oci* i té l'atribut de descompte (que és unic dels socis VIPs, la seva implementació de la funció per verificar les dades (comprova que el descompte no sigui superior al 50%), la seva implementació de la funció per calcular la factura (se li aplica el descompte) i la seva versió de la funció que retorna el tipus de soci (VIP).
- SociEstandard: és una subclasse de S*oci* i té l'atribut de l'assegurança (únic dels socis estàndards i dels juniors), el getter i el setter d'aquest atribut, la seva implemetació de la funció per verificar les dades (comprova que l'assegurança sigui bàsica o completa), la seva implementació de la funció per calcular la factura (a diferència dels socis VIPs aquest paga l'assegurança) i la seva versió de la funció que retorna el tipus de soci (estàndard).
- SociJunior: és una subclasse de *SociEstandard* i té l'atribut de l'edat (únic dels socis juniors), la seva implementació de la funció que verifica les dades (comprova que l'edat sigui inferior a 18 anys), la seva implementació de la funció que calcula la factura (els socis juniors no paguen les activitats) i la seva versió de la funció que retorna el tipus de soci (junior).
- LlistaSocis: aquesta classe té un array de tots els socis i un atribut de capacitat. Els seus mètodes són: getters i setters, una funció que retorna la mida de la llista, una funció per afegir un soci, una funció que comprova si un soci determinat és a la llista, una funció per eliminar un soci, una funció que retorna el soci que està en una posició donada de la llista, una funció que retorna un soci a partir del seu dni, una funció que dona una subllista de la llista de socis per tipus, una funció per esborrar la llista, una funció per veure si és plena i una per veure si és buida, una funció que verifica les dades dels socis de la llista, una funció que imprimeix les dades dels socis de la llista. una funció per canviar l'assegurança d'un soci determinat i una altra per crear un nou soci.

- Activitat: aquesta classe conté tota la informació d'una certa activitat (nom, dia, franja horaria i preu) i conté els getters i setters dels seus atributs i una funció per imprimir les dades de l'activitat.
- LlistaActivitats: aquesta classe té un array de totes les activitats. Els seus mètodes són: una funció que retorna la mida de la llista, una funció per afegir una activitat a la llista, una funció per esborrar una activitat, una funció que retorna l'activitat de la posició donada, una funció per esborrar la llista, una funció per veure si la llista és buida, una funció que calcula el preu de les activitats de la llista, una funció que imprimeix les dades de totes les activitats i una funció que omple la llista amb les activitats que hi ha en un fitxer.
- Assegurança: aquesta classe guarda les dades necessàries per la asseguraça de cada soci (tipus i preu) i conté els getters i setters dels seus atributs i una funció per imprimir les dades de l'assegurança.
- ClubSocial: aquesta és la classe principal del programa i conté tota la informació sobre el club, és a dir, els diferents preus de les quotes, els descomptes dels socis VIPs i les llistes de socis i activitats. A més a més conté les funcions que permeten realitzar les accions de cada opció del menú.
- ExcepcioClub: aquesta classe és la que es fa servir per comprovar les excepcions a les diferents funcions de tota la pràctica, quan es detecta una es crida el mètode d'aquesta classe i dona el missatge d'error que correspon.
- VistaClubSocial: aquesta és la classe que executa el menú i dona totes les opcions perquè l'usuari faci el que necessiti (donar d'alta un soci, mostrar els socis, mostrar els socis VIPs, mostrar els socis estàndards, mostrar els socis juniors, eliminar un soci, verificar les dades dels socis, mostrar totes les activitats, mostrar la llista d'activitats d'un soci determinat, afegir una activitat a un soci determinat, veure la factura d'un soci determinat, modificar el nom d'un soci, canviar l'assegurança d'un soci, guardar les dades en un fitxer, recuperar les dades d'un fitxar i sortir).

2. Explica com has declarat l'atribut de la classe ClubSocial i perquè.

A la classe ClubSocial hem declarat com a atributs les diferents quotes dels diferents tipus de socis (float), el descompte dels socis VIPs (float) i les llistes de socis i activitats (LlistaSocis i LlistaActivitats respectivament). La llista d'activitats la hem omplert amb la funció per carregar la llista a partir de les activitats d'un fitxer i la de socis l'hem creat per anar omplint-la a mesura que ho requereixi l'usuari en el main.

3. **Dibuixar el diagrama de relacions entre les classes que heu utilitzat a la vostra pràctica. No cal incloure la llista d'atributs i mètodes.**



4. **Explicar quins són els atributs de la classe Soci i perquè.**

Els atributs de la classe Soci són el nom del soci, el dni, la seva quota i la llista d'activitats que realitza, ja que són les dades que comparteixen tots els tipus de socis. Els atributs més específics com els descomptes, les assegurances i les edats estan declarats a cada una de les subclasses de Soci corresponents.

5. **Per què creieu que la classe Soci ha de ser abstracta?**

La classe Soci ha de ser abstracta perquè conté mètodes abstractes (mètodes que s'han definit però que tenen la seva implementació a cadascuna de les seves subclasses) com per exemple els mètodes verifica, calcular factura o tipus soci. Ja que cadascun d'aquests mètodes rebrà una implementació diferent segons la subclasse a la que estiguem ja que els diferents tipus de socis tenen diferents característiques.

6. **Creieu que podríem haver treballat només amb dues classes Soci i SociVip? Per què creus que és necessari crear les tres classes Soci, SociEstandard, SociVip i SociJunior?**

Podríem haver treballat només amb dues classe, però aleshores hauríem malgastat molta memòria ja que als socis VIPs, com a subclasse de *Soci*, se'ls haurien declarat una sèrie d'atributs que serien irrellevants per ells (assegurança i edat).

7. **Indiqueu si hi ha i on es troben els exemples de mètodes polimòrfics al vostre codi.**

A la nostra pràctica podem trobar exemples de mètodes polimòrfics en la classe *Soci* i les seves subclasses, ja que tant el mètode calcularFactura, com el mètode verifica, com el mètode toString, com el mètode tipusSoci estan definits de diferents maneres en cada una de les diferents classes nombrades degut a les diferències del que han de fer en cadascuna d'elles.

8. **Analitzeu perquè l'objecte de tipus Asseguranca es crea fora del constructor de la classe SociEstandard. Es podria instanciar dins? Com canviaria el sentit de la implementació?**

Es podria també simplement declarar els atributs de la classe *Asseguranca* dins la classe *SociEstandard* i afegir tots els seus mètodes, però crear la classe per separat permet millorar la possible reutilització del codi per algun futur projecte i millora la simplicitat i l'ordenació del codi.

9. **Expliqueu com heu utilitzat la classe *ExepcioClub* al vostre codi.**

A la classe *ExcepcioClub* hem definit un mètode que es fa servir quan es detecta un error i es llença per donar el missatge corresponent. Llavors a cada mètode de cada classe que es fa alguna comprovació que pot donar a algun error es fa un try/catch

que crida al mètode d'excepció quan es detecta una per informar de quin error s'està cometent.

10. Detallar les proves realitzades per comprovar el correcte funcionament de la pràctica, resultats obtinguts i accions derivades.

Per comprovar que el nostre programa funcionava l'hem executat i hem provat cadascuna de les opcions del main per veure si funcionaven bé i feien exactament el que havien de fer. En el cas de que no fos així ens hem fixat en si sortia algun error (i per tant hem intentat solucionar-lo on correspongués) o si el problema era que no donava el resultat correcte (i per tant hem buscat quin era el punt del codi que creava aquesta diferencia per modificar-lo).