PRÀCTICA 3 MERCATUB

MercatUB

Oscar de Caralt Roy MC

Pau Segura Baños MC

ENGINYERIA INFORMÀTICA UB

Primer de carrera: Programació 2

26/04/2021

ÍNDEX:

Introducció:

Se'ns ha demanat desenvolupar un sistema de gestió de comandes per a una companyia que treballa en l'àmbit del comerç electrònic. Els principals elements que la companyia fa servir són els articles que ven, els clients i les pròpies comandes.

Per a realitzar això es requereixen uns coneixements bàsics i un cert domini de POO (programació orientada a objectes), herència de classes, de polimorfisme, de creació de classes i interficies i d'encapsulament.

A l'apartat d'implementació de classes hi ha una explicació de cada classe, però l'explicació de cada mètode està inclosa als respectius javadocs.

1. **Explica les classes implementades:**

➔ **Articles:** aquesta classe conté un constructor que inicialitza l'objecte article amb els atributs codi, el nom del producte,el preu, si admet un enviament urgent o no, i el temps. Cada atribut té el seu respectiu getter i setter, i la clase també conté un mètode toString().

➔ **Client:** és una classe abstracta que conté un constructor que inicialitza l'objecte client amb els atributs nom, correu electrònic i la direcció. Cada atribut té el seu respectiu getter i setter, i la clase també conté un mètode toString().

➔ **ClientEstandard:** aquesta classe hereda de la classe client per crear un client estandard. Té els mateixos atributs, però la característica més ressenyable que té és que no paga cap mensualitat ni té cap descompte.

➔ **ClientPremium:** aquesta classe hereda de la classe client per crear un client premium. Té els mateixos atributs, però la característica més ressenyable que té és que paga mensualitat de 4 euros i té un descompte de 20%.

- **Comanda:** aquesta classe abstracta crea una comanda amb diferents articles per cada client. El constructor inicialitza l'obkecte comanda amb els atributs client, articles, quants articles te demanats i la data de creació de la comanda. La data i la quantitat d'objectes tenen getter i setter. Declara 4 mètodes abstractes: tipusComanda, comandaEnviada, comandaRebuda i preuEnviament. Implementa dos mètodes: calcPreu, per calcular el preu de la comanda sense l'enviament i el toString.

- **ComandaNormal:** hereda la classe comanda i implementa els mètodes declarats a la classe comanda.

- **ComandaUrgent:** hereda la classe comanda i implementa els mètodes declarats a la classe comanda.

- **InDades:** declara els mètodes que s'han d'implementar a la classse Dades.

- **Dades:** Serà la classe principal del package model, on implementarem tots els mètodes que posteriorment seran cridats al menú pel controlador, és a dir, és la columna vertebral del nostre treball ja que és on podrem ajuntar la resta de classes implementades anteriorment i donar forma al nostre projecte.

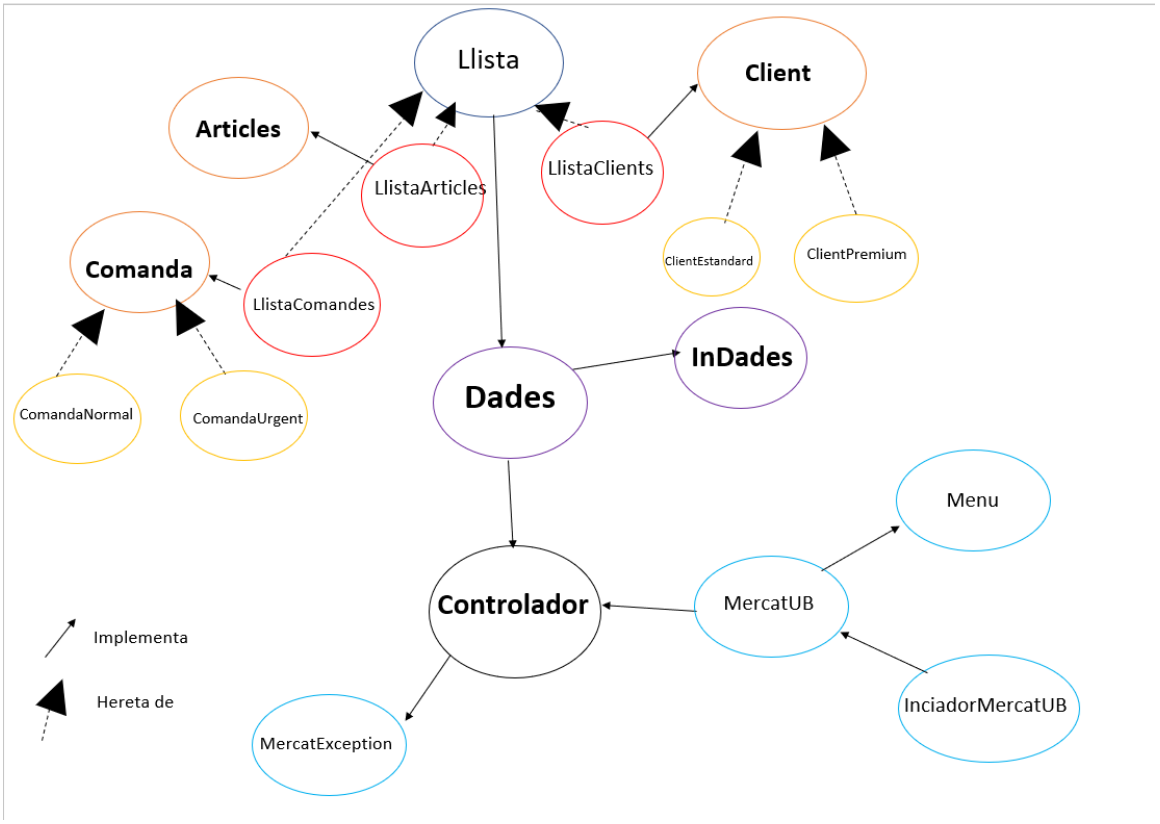
- **Llista:** aquesta és una classe genèrica que declara els mètodes per treballar amb arraylist. Es declaren d'una manera genèrica(amb T) per no haver-los de declarar tres vegades, per cada tipus d'objecte que hi ha. Es declaren els mètodes, getSize,afegir(T t), esborrar(T t), getAt(position), clear, isEmpty i getArrayList.

- **LlistaArticles:** Aquesta classe també hereda de la classe Llista. S'han d'especificar els mètodes que necessiten un objecte i implementar els mètodes amb objectes de la classe Articles en comptes d'objectes de la classe genèrica

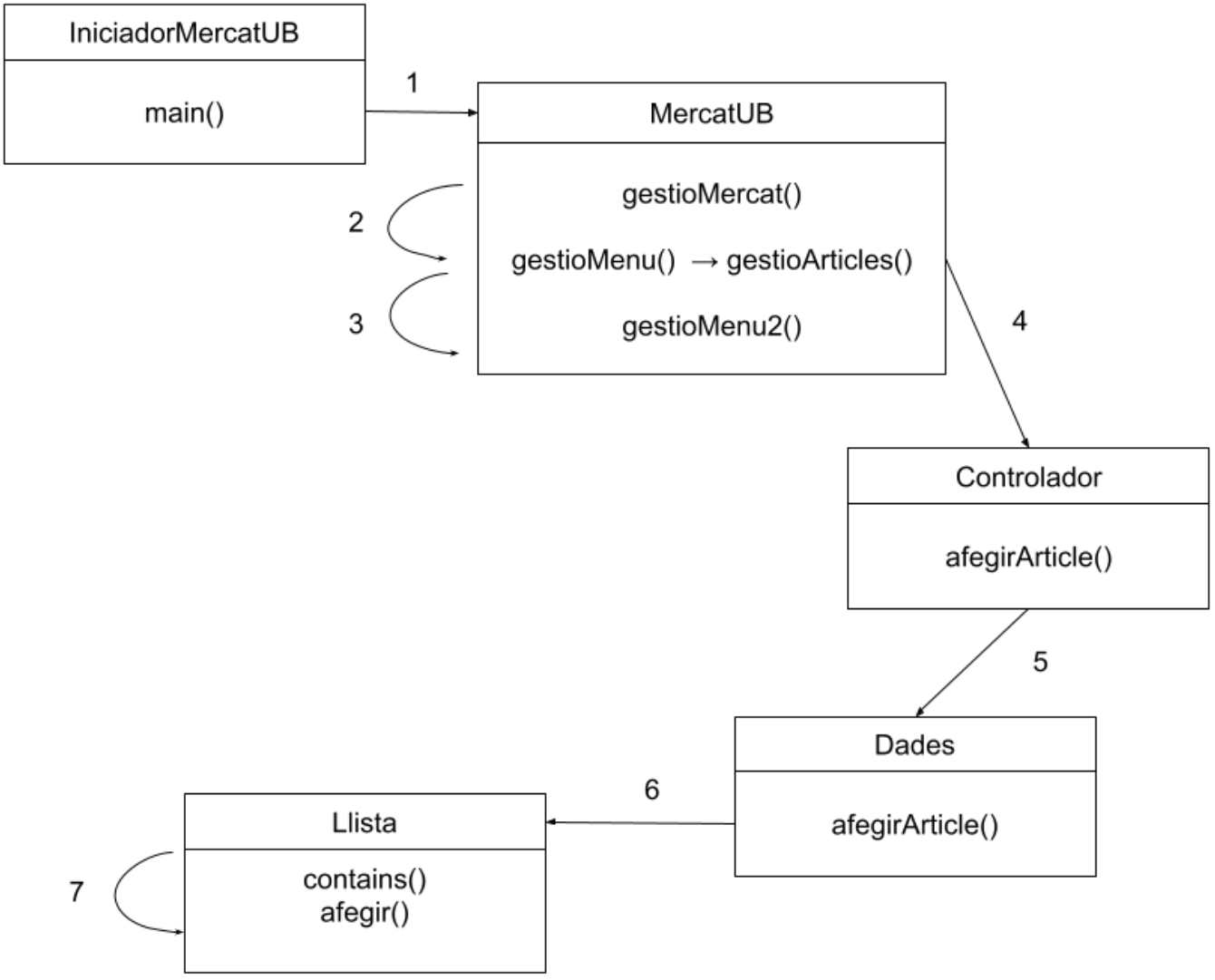
➔ **LlistaClients**: Aquesta classe també hereda de la classe Llista. S'han d'especificar els mètodes que necessiten un objecte i implementar els mètodes amb objectes de la classe Clients en comptes d'objectes de la classe genèrica

➔ **LlistaComandes:** Aquesta classe també hereda de la classe Llista. S'han d'especificar els mètodes que necessiten un objecte i implementar els mètodes amb objectes de la classe Comandes en comptes d'objectes de la classe genèrica.

2. **Dibuixar el diagrama de relacions entre les classes que has utilitzat en la teva pràctica. No cal incloure la llista d'atributs i mètodes.**



3. **Feu un diagrama de flux per mostrar el recorregut que fa el vostre programa quan s'executa l'opció d'afegir un article al sistema. Especificar els mètodes que es criden en cadascuna de les classes. Feu servir fletxes i números per indicar l'ordre de les crides.**



4. **Explicar quins són els atributs de la classe Dades i per què.**

Els atributs de la classe Dades són els següents:

```
private LlistaArticles llista = new LlistaArticles();
private LlistaClients llistaclients = new LlistaClients();
private LlistaComandes llistacomandes = new LlistaComandes();
```

Tots tres estan en private per a afavorir l'encapsulament de les dades, és a dir, per amagar els valors o l'estat d'una estructura de dades dins d'un objecte, prevenint d'aquesta manera accessos a la seva informació per part de tercers. Les classes solen tenir els atributs com a privats i els mètodes com a públics, perque tracten la seva informació interna, de manera que tercers (altres objectes) poden interactuar d'una forma segura amb la informació interna d'aquestes.

L'atribut llista ens permet crear una ArrayList on podrem cridar als mètodes de la classe LlistaArticles per a operar amb l'ArrayList formada per articles, mentres que l'atribut llistaclients ens permet crear una ArrayList on podrem cridar als mètodes de la classe LlistaClients per a operar amb l'ArrayList formada per clients i l'atribut llistacomandes ens permet crear una ArrayList on podrem cridar als mètodes de la classe LlistaComandes per a operar amb l'ArrayList formada per comandes.

5. I**magina que un company teu et passés la seva implementació de la classe Dades per tal de substituir la del teu codi, quines modificacions serien necessàries en la vista i el controlador? Com es relaciona la interfície InDades amb la pregunta anterior? Justifica la resposta.**

En la classe controlador ens donaria diversos errors ja que hi haurien diversos mètodes que no coincidirien en els noms (com per exemple aquells que impliquin al fitxer), i segurament ens faltarien getters de la nostra classe dades, la qual cosa causaria que retoquèssim aquells mètodes que interactuessin amb ArrayLists (tant d'Articles, com de clients, com de comandes) i com a conseqüència d'aquests errors a la classe controlador, en el paquet de vista, a la classe MercatUB, ens donarien els mateixos tipus d'errors.

6. **Si tinguessis llibertat per implementar LlistaArticles, LlistaClients i LlistaComandes com volguessis, però no poguessis fer servir classes genèriques (com Llista <T>), com portaries a terme la implementació? Analitza les avantatges i inconvenients de la solució adoptada.**

   La implementació acabaria resultant la mateixa en les tres classes. Contindrien les tres els mateixos mètodes. Només canviarien d'una classe a l'altra els mètodes que necessitin un objecte, que en cada classe contindran l'objecte corresponent.

   Els inconvenients serien que perdríem massa temps ja que hauríem d'escriure tres vegades literalment el mateix, amb una petita diferència del tipus d'objectes que entrèssin als mètodes i en el constructor.

   Avantatges: Evitaríem el fet d'utilitzar classes genèriques que, de moment, és una cosa més desconeguda i complicada per a nosaltres que encara no entenem gaire bé com funcionen.

7. **Detallar les proves realitzades per comprovar el correcte funcionament del codi, resultats obtinguts i accions derivades.**

   Primer creem un article. Per cada article ens demana el codi, el nom del producte, el preu, si admet o no enviament urgent i el temps d'arribada del producte. Si afegim dos productes amb el mateix codi el programa llança una excepció i no afegeix el segon article a la llista.

   Un cop hem creat l'article, és el torn de crear un client. Per crear un client ens demana el correu electrònic, el nom, la direcció i si és premium o no. Si hi ha dos clients amb el mateix correu electrònic el programa llança l'excepció i no afegeix el segon client a la llista de clients.

   Ara creem la comanda. Demana la posició de l'article en la llista d'articles i la posició del client a la llista de clients. Si alguna de les llistes està buida, no es pot crear una comanda. Quan creem la comanda també ens demana quants productes vol i si la comanda és urgent o no. Comprovem que s'hagi creat la comanda imprimint la llista de comandes. Si al crear l'article hem posat un temps de 2 mins haurem d'esperar 2 mins a partir de la creació d'aquest perquè quan imprimim la comanda, l'atribut enviat sigui true enlloc de false. Quan l'atribut enviat i rebut son true implica que la comanda s'ha enviat i rebut.

Ara guardem en un arxiu el que hem fet. Li donem a l'opció de guardar i tanquem l'execució del programa. El tornem a executar i fem l'opció de recuperar l'arxiu. Fem que el programa torni a imprimir la llista de comandes i tot està com abans de sortir.
Si tot això s'ha reproduït amb èxit, vol dir que el programa és correcte.

8. **Observacions generals.**

En totes les classes tenim *implements* Serializable perque l'*Stream* es pugui generar correctament.

D'aquest treball, el que més podríem destacar, són els mètodes on tractem amb dates, ja que són el tipus de mètodes als quals hem dedicat més temps però que finalment hem pogut implementar. També podem destacar el tractament d'excepcions, que ens ha comportat bastanta feina ja que han sorgit certes dificultats que al final hem pogut tractar i superar a l'hora de crear-les i programar-les, i hem pogut notar una gran millora en el tractament d'excepcions respecte la pràctica passada perquè aquest cop no aturavem el programa cada cop que es produïa una excepció, simplement imprimiem un missatge per pantalla i actuàvem en conseqüència a aquest error, per exemple: si no podiem afegir un article a l'ArrayList perquè en aquesta ja hi havia un article amb aquell codi, al finalitzar d'introduir les dades d'aquest article 'repetit' es llançava una excepció i no s'introduïa l'article però podies seguir treballant amb el programa des d'on ho havíem deixat .

A part d'això, aquesta entrega ens ha semblat bastant similar a nivell de dificultat respecte la segona, però amb més coses a tenir en compte, com per exemple, el controlador.

Conclusions:

Hem assolit els objectius personals que ens vam proposar al començament de la pràctica, que eren, anar treballant poc a poc cada dia, però amb una certa constància i tenir el projecte acabat amb 5 dies d'antelació al lliurament. Així que els resultats obtinguts en aquest projecte són altament satisfactoris.