

Pràctica 3

Assignatura: Programació

NOM DE L'EQUIP: Grup21

EQUIP:

Bernat Sort Rufat

Sergi Sabalete Muns

Alejandro Teodoro Sinisterra

Oriol Villanova Llorens

DIRECTOR: Raúl Almenara Peñaranda

DATA: 16 / Desembre / 2018

Índex

1.	<i>Continguts</i>	2
1.1	Introducció	2
1.2	Anàlisi i disseny del programa	2
1.2.1	Productes de viatge	3
1.2.2	Clients	5
1.2.3	Reserves.....	6
1.2.4	Main.....	7
1.3	Fitxers de text	8
1.4	Avaluació	9

1. Continguts

1.1 Introducció

Al començar aquesta pràctica, vam observar 4 parts ben diferenciades.

Primer de tot, la part corresponent a tots els productes amb els que treballaríem dins del programa. Tot seguit, tot el relacionat amb els clients de l'aplicació, així com les reserves que es podien efectuar dins de l'aplicació per poder controlar els diferents clients.

Per últim, la part del main o programa principal, on podem diferenciar dues parts: la que gestionarà els clients de l'empresa de viatges i la que podran interactuar els clients.

El treball l'hem repartit de manera que tots els membres de l'equip tinguin una quantitat de feina similar.

En el nostre cas vam estar d'acord des del començament de quina seria la part que faria cadascú:

Alejandro Teodoro: S'ha encarregat del Programa Principal i del menús necessaris per a fer funcionar l'aplicació.

Sergi Sabalete: S'ha encarregat de la classe reserves i tot el seu contingut i variants que estiguin directament relacionades amb aquest apartat.

Bernat Sort: S'ha encarregat de la classe clients i totes les seves variants, la llista de clients i de totes les funcions que els clients podien desenvolupar en aquestes classes.

Oriol Villanova: S'ha encarregat de la classe productes i tot el seu contingut i variants que aquest puguin tindre, com poden ser els circuits i els transports.

En l'apartat de desenvolupar la interfície gràfica pel programa de clients s'ha treballat d'una manera més conjunta. Tot i així, mentre que el Bernat ha documentat als seus companys sobre el funcionament de la interfície i ha iniciat els primers passos per a desenvolupar-la, el Sergi i l'Alejandro han acabat de desenvolupar-la amb profunditat.

1.2 Anàlisi i disseny del programa

Per a poder començar a programar de manera eficient, primer de tot vam realitzar un estudi previ de la pràctica. Aquest estudi inicial ens va servir per identificar les diferents parts del treball, adonant-nos que algunes s'identificaven sense problemes mentre d'altres presentaven algunes complicacions.

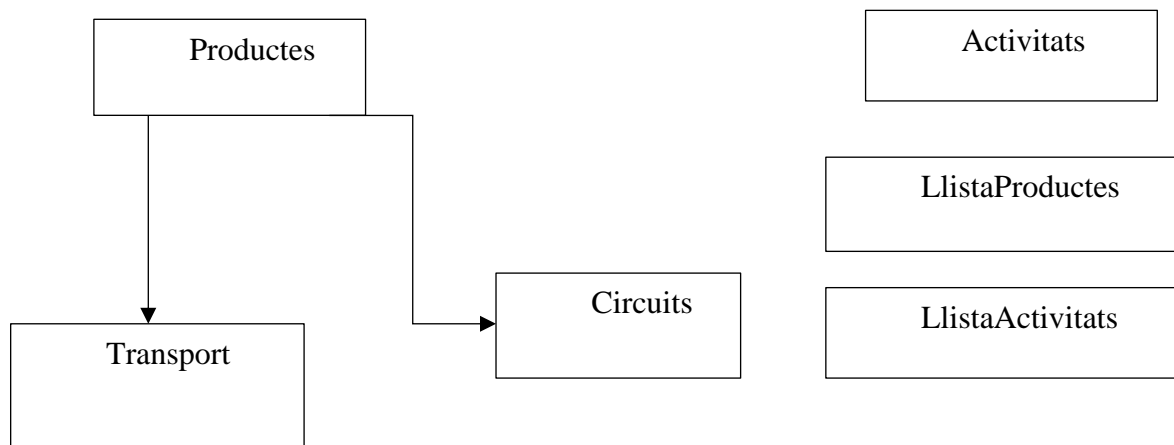
Al plantejar el desenvolupament de la pràctica, vam observar que aquesta constava principalment de 4 parts: la part relacionada amb tots els productes de viatge, la part relacionada amb els clients, la relacionada amb les reserves i per últim el programa principal o main que s'encarrega de controlar les tres anteriors.

Per tal de tenir la informació ben endreçada, el nostre programa ha estat dividit en 4 paquets: els fitxer per l'aplicació de la consola, l'aplicació amb la interfície gràfica, les dades i les excepcions definides.

1.2.1 Productes de viatge

L'enunciat de la pràctica demanava controlar productes de viatge. En aquest cas només poden haver-hi de dos tipus: transport, que són productes més habituals, com pot ser un viatge en tren o en autobús i circuits. Aquest darrer consistia en una ruta tancada de viatge on es podien fer varies activitats disponibles i amb una durada d'uns dies.

Per a programar aquest apartat hem seguit el següent esquema de classes:



Per a programar el control de productes hem hagut d'implementar un total de 6 classes.

La primera de totes és la classe Productes. Aquesta classe és l'encarregada de guardar els atributs comuns en tots els productes que hi ha o hi haurà en un futur, ja que com està feta la implementació ens dona pas a poder afegir més productes en un futur si fos necessari.

Aquets atributs són: la ID o identificador del producte (un número que se li assigna a cada producte i que és diferent en cadascun d'ells), el nom del producte (on s'especifica el seu nom), el preu d'aquest, el total de places que hi ha en els productes (per exemple 52 en un autobús), les places disponibles que queden lliures i el tipus de transport que s'utilitzarà.

Aquesta classe, al tractar-se de la classe pare, hem d'afegir mètodes comuns. En aquest cas hem definit els constructors, getters i setters i el toString de la classe.

La classe Transport rep tota la informació de la classe productes. Apart dels atributs que rep per la herència, s'han definit els següents atributs propis: la ciutat d'origen del transport, la ciutat de destí, la data de sortida i la data d'arribada.

Cal destacar a l'hora de controlar les dates al llarg de tota la pràctica hem creat una classe data amb la informació necessària per tractar-ho tot. En aquesta classe hem definit els següents mètodes: els constructors de la classe, getters i setters, el toString i un mètode per poder fer una còpia del transport en qüestió.

La classe Circuit rep per herència tota la informació de la classe Productes. A més a més, aquesta implementa els seus propis atributs: el país on es fa el circuit, els dies que dura, la data que s'ofereix i la data que s'acaba d'oferir.

També s'ha afegit una llista d'enters per a controlar les activitats que es poden fer en aquell circuit (poden ser més d'una). En aquesta llista es guarden els números de la posició que es troben les activitats dins de la classe llista Activitats. En aquesta classe a part de tots els mètodes comuns que ja hem comentat en les anteriors classes, hem hagut de definir un mètode nou de control de llistes: en aquest cas hem afegit el mètode d'afegir una activitat que rep un enter per paràmetre i l'afegeix a la llista en el seu espai corresponent.

La classe LlistaProductes és la classe que més funcions ha de controlar, ja que s'encarrega d'emmagatzemar una llista amb tots els productes que estiguin disponibles. En aquesta llista podem guardar tant circuits com transports, ja ambdós són productes. En aquesta classe hem de controlar una gran quantitat d'aspectes de la llista i de les seves funcions. Per aquest motiu, hem hagut de fer els següents mètodes: un mètode per afegir circuit i un altre per transport (encarregats de posar el respectiu circuit o transport dins de la llista), dos mètodes per actualitzar la informació d'un circuit o un transport. A aquests dos mètodes se li passa la informació d'un circuit o transport del qual es vol actualitzar la informació. Si el troba dins la llista s'actualitza, sinó, afegeix el circuit directament a la llista. Un mètode per esborrar un circuit que sigui com el passat per paràmetre, un altre per esborrar un transport passat per

paràmetre, un mètode que retorna una llista de tots els productes que tenen places disponibles i finalment, un mètode que retorna una llista amb tots els productes que no hagin caducat.

La classe Activitat es una classe independent on hem de controlar tot el que es pot fer en una activitat. En aquest cas, ha de ser el nom de l'activitat, el lloc on es realitza i la descripció d'aquesta. En aquesta classe hem implementat els constructors de la classe, getters i setters i el toString, ja que només ens serveix per emmagatzemar la informació.

Per últim, la classe LlistaActivitats. En aquesta classe es controlen totes les activitats que es poden realitzar dins dels circuits. Aquesta classe fa les funcions bàsiques d'una llista: afegir, esborrar i buscar. És una classe destinada a guardar informació i a la gestió d'aquesta. El que ens interessa d'aquesta classe és l'índex de la llista on està guardada cada activitat per poder-les assignar als circuits.

1.2.2 Clients

En l'enunciat ens demanava un control dels clients que anessin a fer us de l'aplicació per a contractar serveis de viatges. En aquest cas, al fer l'estudi del que havíem de fer vam arribar a la conclusió que només necessitàvem dos classes, una classe de clients i una classe LlistaClients, com es mostra en el següent esquema:



També hem definit una excepció d'errors per a poder controlar la introducció del DNI.

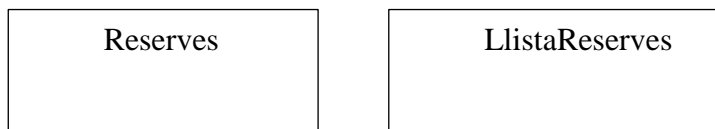
En la classe clients, hem definit els següents atributs: el dni, el nom i l'adreça correu electrònic. El dni ha de ser únic per a cada client i no poden haver-hi dos de repetits. Aquesta classe s'ocupa d'emmagatzemar dades. Consta de mètodes bàsics, els constructors de la classe, els getters i setters, el mètode toString, un mètode per a poder fer una còpia del client i finalment un mètode que ens diu si un dni passat per paràmetre es igual al que tenim.

En la classe LlistaClients implementem una llista per a fer el control dels clients que tenim i de totes les seves funcionalitats.

Aquesta classe presenta mètodes de control de llista, com el mètode per afegir un client (dins d'aquest mètode té el control d'excepcions per a comprovar si un dni està repetit). Si el dni està repetit, genera un missatge d'error indicant-nos que no pot afegir el client (ja està donat d'alta). Dins de la classe també tenim el mètode d'esborrar la dada d'un client, getters i setters i el mètode toString.

1.2.3 Reserves

En la classe Reserves s'han de controlar totes les reserves que els clients poden fer. Els clients podran fer reserves de productes que no estiguin caducats. Observem el següent esquema:



Hem de definir aquestes dos classes: la classe Reserves i la classe LlistaReserves.

La primera conté tota la informació necessària per a fer una reserva, mentre que la segona conté la llista de reserves que té la companyia de transports.

En la classe Reserves tenim el producte que s'ha reservat, el client que l'ha reservat, la data de quan s'ha reservat i un mètode estàtic que ens diu en tot moment quin és el producte que té més reserves.

En aquesta classe hem implementat els mètodes bàsics, els constructors de la classe, getters i setters, un mètode per fer una copia de la reserva i un mètode per actualitzar el atribut estàtic.

En la classe LlistaReserves fem les funcions bàsiques d'una llista, on hem d'emmagatzemar tota la informació de les reserves realitzades. En aquest cas hem implementat mètodes per buscar reserves, així com per esborrar reserves, getters i setters, i afegir reserves que aquesta implementa una excepció d'errors per a veure si la data està caducada o no.

1.2.4 Main o Programa Principal

El programa principal s'encarrega d'ajuntar totes les parts en el menú. Hem dissenyat un programa principal que et demana, primer de tot, el dia actual per poder saber quan o no un producte ha caducat. A continuació, et demana si vols utilitzar el programa com a treballador de l'agència de viatges, com a client o si desitges sortir. En aquest cas si dones la opció de treballador, es carrega el següent menú:

```
1
Opcions del menú:

1. Afegir un transport
2. Afegir un circuit
3. Afegir una activitat
4. Donar d'alta un client
5. Donar de baixa un client
6. Mostrar els productes de viatge amb alguna reserva
7. Mostrar els productes de viatge amb places disponibles
8. Mostrar els productes de viatge no caducats
9. Mostrar el producte de viatge amb més reserves
10. Mostrar la llista de productes
11. Mostrar la llista d'activitats
12. Mostrar la llista de clients
```

Amb totes aquestes opcions es poden fer tots els apartats que demanava l'enunciat de la pràctica.

En el cas del client, s'obrirà una interfície gràfica, encarregada de realitzar diverses funcions: Primer de tot, per a utilitzar l'aplicació, el client s'haurà de registrar amb el seu DNI per trobar la seva informació. El resultat serà una finestra com la següent:

Aquesta és una captura d' pantalla d'una finestra d'interfície gràfica d'usuari. La finestra té un títol "Iniciar Sessió" amb un icona de clau. Dins de la finestra, hi ha el text "Introdueix el teu DNI:" seguit d'un camp de text buit amb un rectangle blau al voltant. A la part inferior de la finestra, hi ha dos botons: "Iniciar Sessió" i "Registrarse".

Un cop hem iniciat la sessió, tindrem una finestra amb varies opcions:

-Buscar els transports la data dels quals no ha caducat i tenen places disponibles. Un cop dins, podem buscar els transports per l'origen, el destí i la data d'arribada. Mostrarà una llista, en la qual el client podrà fer reserves indicant quantes places vol.

-Buscar circuits que introduint el destí i la data d'arribada, mostrarà una llista amb tots els circuits que no estiguin caducats i tinguin places disponibles. En la llista, el client podrà fer reserves indicant quantes places vol.

-Un Botó on es poden consultar les reserves que ha fet el client, les reserves que encara no han passat es poden eliminar.

-Un botó per sortir, el qual et torna a la pagina d'inici on et pots identificar com a client.

Aquestes funcionalitats estan totes implementades en el següent menú:



Per a poder tancar el programa definitivament, un cop hem sortit del menú i per tant hem tancat la sessió de l'usuari hem de tancar el programa per la creueta vermella.

1.3 Fitxers de text

En la pràctica per a guardar tota la informació de manera permanent (per a que no es perdi un cop s'apagui el programa) ho guardem tot a fitxers. Aquets fitxers estan separats per temes. En el nostre cas tenim un fitxer pels circuits, un pels transports, un pels clients, un per les reserves i un per les activitats. Aquets han estat desenvolupats de la mateixa manera: s'han ficat els atributs en l'ordre en que estan instanciats dins de les classes per a una lectura més ràpida i separats entre ells per un ; per saber quin es el límit. Un exemple podria ser el següent:

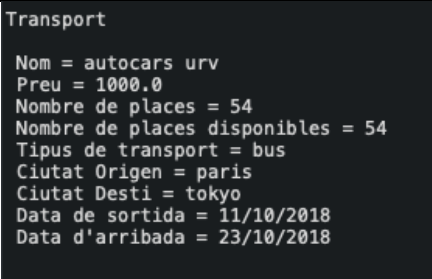
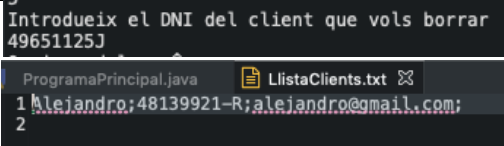

Visita a Paris;Ruta turística;Paris;

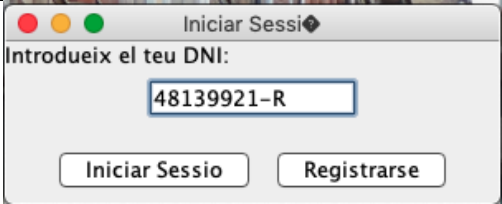
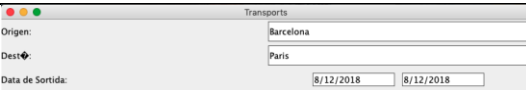
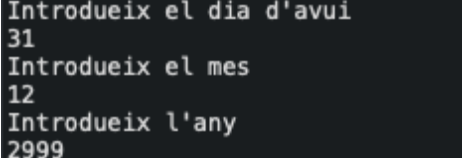
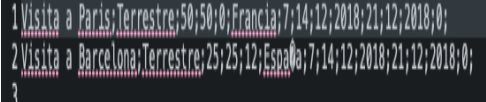
Visita al Louvre;Visita guiada al museu;Paris;

Visita a la Sagrada Família;Visita guiada a la Sagrada Família;Barcelona;

1.4 Avaluació

Per a provar el programa, hem creat un joc de proves amb el qual podrem veure els casos especials en que poden haver problemes a l'hora d'executar codis. Observem doncs la següent taula:

Paràmetres d'entrada	Sortida Teòrica	Sortida Real
Entrar com a treballador de l'agència, introduir un producte nou i consultar-lo fent que la taula estigui plena i l'hagi de fer més gran	1.- Entrar com a treballador i que es mostri el menú correcte 2.- Afegir el producte i que es col·loqui i mostri correctament.	 <pre>Transport Nom = autocars urv Preu = 1000.0 Nombre de places = 54 Nombre de places disponibles = 54 Tipus de transport = bus Ciutat Origen = paris Ciutat Desti = tokyo Data de sortida = 11/10/2018 Data d'arribada = 23/10/2018</pre> Resultat : OK
Des del menú de administrador donar d'alta un client i donar-lo de baixa.	Es crea el client correctament, s'afegeix al seu fitxer corresponent i es eliminat posteriorment.	 <pre>Introdueix el DNI del client que vols borrar 49651125J ProgramaPrincipal.java LlistaClients.txt 1 Alejandro;48139921-R;alejandro@gmail.com; 2</pre> S'afegeix i es borra correctament. Resultat : OK
Des del menú d'administrador, afegir una activitat, afegir-la en un circuit i comprovar que la llista d'activitats correspon a la introduïda al circuit	El índex que està guardat en la llista de circuits correspon a la dada corresponent a les Activitats.	 <pre>Circuit Nom = Tour italia Preu = 200 Nombre de places totals = 23 Nombre de places disponibles = 23 Tipus de transport = tancat Dest0 = italia Dies = 30 Data Oferiment = 15/11/2018 Data Acabament = 30/11/2018</pre> Comprovació activitat Debugger. Resultat : OK

Comprovar que s'executa la part del client.	S'obre la pestanya per poder identificar-se com a client. Si es fica un DNI no identificat no deixa entrar.	 <p>Login correcte amb client identificat i incorrecte amb no identificat</p> <p>Resultat : OK</p>
Buscar un transport per a poder reservar-lo com a client.	Es busca el transport, surt correctament per pantalla i es pot reservar.	 <p>Es pot buscar correctament.</p> <p>Resultat : OK</p>
Comprovar que a la llista no mostri els productes que estan caducats i que diferenciï bé entre circuits i transports.	Entrar amb un any molt elevat i comprovar que no ens retorni productes que estan caducats.	 <p>Quan es demana la opció de mostrar els no caducats no contesta res.</p> <p>Resultat : OK</p>
Tancar l'aplicació, tornar-la a obrir i tots els canvis s'hagin guardat correctament.	Al tancar l'aplicació i al obrir els fitxers .txt on està guardada la informació trobar-la emmagatzemada.	 <p>Es guarden totes les dades corresponents i es queden per al següent ús del programa.</p> <p>Resultat : OK</p>