Programació Orientada a Objectes. Examen Parcial

Quadrimestre 2021-2022-Q2. 13 de Maig 2022

Es vol realitzar la web ViatgeFelis de crítica i reserva de viatges, que permeti a viatgers de tot el món consultar els comentaris d'altres viatgers, respecte als hotels, restaurants i atraccions que ofereixen les ciutats que visitarà.

Aquesta web també permetrà als seus usuaris realitzar les reserves que necessitin mentre planifiquen el viatge, així com realitzar els seus propis comentaris.

La web estarà formada pels llocs on han estat els viatgers: restaurants, hotels i atraccions, així com els comentaris i reserves que han realitzat els usuaris respecte a aquests llocs.

A la web podran accedir tot tipus de viatgers, que seran els usuaris. Els usuaris podran realitzar i consultar comentaris, i realitzar reserves.

La classe Controlador gestiona els usuaris (cada usuari té un email únic), els llocs (cada lloc té un nom únic) i les reserves (cada reserva té un identificador únic).

Heu de tenir en compte els següents fets:

- Els comentaris es guarden en l'ordre en que han estat realitzats i poden estar repetits.
- Cada cop que creem una reserva s'actualitza el número de reserves i, aquest valor s'utilitza com a id de la reserva.
- La data d'una reserva és una informació que ens han de donar per crearla.
- Per cada usuari, i per cada lloc, les reserves corresponents s'han de poder ordenar.

El diagrama UML de l'aplicació es mostra a l'última pàgina d'aquest enunciat.

NOTA 1: Per resoldre les preguntes de l'examen podeu suposar que els mètodes **getXXX() i setXXX(),** que accedeixen als atributs de las classes, ja estan implementats, encara que no apareguin a l'UML. **No cal que els implementeu**.

NOTA 2: Heu de considerar que aquest programa gestiona les dates mitjançant Strings, el format dels quals és **YYYYMMDD** (Y indica dígit d'any, M indica dígit de mes i D indica dígit de dia). Així, per exemple, la data 7 de Juny del 2018 s'indicaria per el string "21080607). Heu de considerar IMPLEMENTADA (NO HEU DE IMPLEMENTAR-LA) la classe Data, que implementa el SEGÜENT MÈTODE ESTÀTIC:

```
public static int comparaDates(String data1, String data2) ;
```

Aquest mètode retorna 0 si les dates representades pels dos arguments són la mateixa data; un enter negatiu si la data representada per l'argument data1 és una data ANTERIOR a la representada per l'argument data2; i un enter positiu si la data representada per l'argument data1 és una data POSTERIOR a la representada per l'argument data2.

PREGUNTES:

1. (1,5 punts) A partir del diagrama de classes UML adjunt, i per a totes les classes, escriviu la capçalera de la classe i la declaració d'atributs, incloent-hi aquells atributs que apareixen a l'implementar les relacions entre classes. De moment NO heu d'implementar cap mètode

En aquest apartat no és necessari que defineixis els constructors ni els mètodes.

```
public class InterficieUsuari {
    private Controlador controlador;
...}

public class Controlador {
    public static final int OK = 0;
    public static final int LLOC_DESCONEGUT = 1;
    public static final int USUARI_DESCONEGUT = 2;
    private String web;
```

```
    private String mail;
    private Map<String, Usuari> usuaris; //clau: email de l'usuari
    private Map<Integer, Reserva> reserves; //clau: id de la reserva
    private Map<String, Lloc> llocs; //clau: nom del lloc
...}

public class Usuari {
    private String nom;
    private String email;
    private String contrasenya;
    private String dadesBancaries;
    private List<Reserva> reserves; // Per cada usuari, i per cada lloc, les
      //reserves corresponents s'han de poder ordenar
    private List<Comentari> comentaris;
...}

public class Reserva {
    private static int numReserves = 0;
    private int id;
    private String data;
    private double preu;
    private Usuari usuari;
    private Lloc lloc;
...}

public class Comentari {
    private String titol;
    private String text;
    private double puntuacio;
    private Lloc lloc;
    private Usuari usuari;
...}

public class Lloc {
    protected String nom;
    protected String adress;
    protected String pais;
    protected List<Comentari> comentaris;
    protected List<Reserva> reservas; // Per cada usuari, i per cada lloc, les
      //reserves corresponents s'han de poder ordenar
...}

public class Atraccio extends Lloc{
    private String activitats;
    private int duracioVisita;
...}

public class Hotel  extends Lloc{
    private String estrelles;
...}

public class Restaurant  extends Lloc{
    private String millorPlat;
```

```
...}
```

2. (2 punts) Implementa el mètodes constructors de les classes `Controlador`, `Lloc`, `Restaurant`, `Usuari` i `Reserva`. Has de decidir tu mateix/a els paràmetres que es passen a cada constructor.

```
public Controlador(InterficieUsuari iu, String web, String mail) {
	this.iu = iu;
	this.web = web;
	this.mail = mail;
	this.usuaris = new HashMap<>();
	this.reserves = new HashMap<>();
	this.llocs = new HashMap<>();
}

public Lloc(String nom, String adress, String pais) {
	this.nom = nom;
	this.adress = adress;
	this.pais = pais;
	this.comentaris = new ArrayList<>();
	this.reservas = new ArrayList<>();
}

public Restaurant(String millorPlat, String nom, String adress, String pais) {
	super(nom, adress, pais);
	this.millorPlat = millorPlat;
}

public Usuari(String nom, String email, String contrasenya, String
dadesBancaries) {
	this.nom = nom;
	this.email = email;
	this.contrasenya = contrasenya;
	this.dadesBancaries = dadesBancaries;
	this.reserves = new ArrayList<>();
	this.comentaris = new ArrayList<>();
}

public Reserva(double preu, Usuari usuari, Lloc lloc, String data) {
	this.preu = preu;
	this.usuari = usuari;
	this.lloc = lloc;
	this.id = ++Reserva.numReserves;
	this.data = data;
}
```

3. (2 punts) Implementa el mètode de la classe `Lloc`

```
public double calculaMitjanaPuntuacio();
```

Aquest mètode retorna el promig de les puntuacions d'un lloc en funció de tots els comentaris rebuts. Si no hi ha comentaris, retorna -1.

```
public double calculaMitjanaPuntuacio(){
```

```
        double puntuacion = 0.0;
        int numComentarios = this.comentaris.size();
        if(numComentarios==0){
            return -1;
        }
        for(Comentari comentari: this.comentaris){
            puntuacion += comentari.getPuntuacio();
        }
        return puntuacion/numComentarios;
}
```

4. (2 punts) Implementa el mètode de la classe `Controlador`

```
public int creaReserva(String nomLloc, String emailUsuari, String data, double preu);
```

Aquest mètode intenta fer una reserva. Rep com a paràmetres el nom del lloc que l'usuari vol reservar, el email de l'usuari que vol fer la reserva i el preu de la reserva.

Si aconsegueix fer la reserva, retorna el valor `Controlador.OK`. Si no hi ha cap lloc amb el nom del lloc passat com argument, retorna el valor `Controlador.LLOC_DESCONEGUT`. Si no hi ha cap usuari amb l'email passat com argument, retornarà el valor `Controlador.USUARI_DESCONEGUT`.

```
public int creaReserva(String nomLloc, String emailUsuari, String data, double
preu) {
        Lloc lloc = this.llocs.get(nomLloc);// en aquest cas no s'ha d'iterar
           // sobre els valors del mapa
        if (lloc == null) {
            return Controlador.LLOC_DESCONEGUT;
        }
        Usuari usuari = this.usuaris.get(emailUsuari); );// en aquest cas no s'ha
           // no s'ha d'iterar sobre els valors del mapa
        if (usuari == null) {
            return Controlador.USUARI_DESCONEGUT;
        }
        Reserva reserva = new Reserva(preu, usuari, lloc, data);
        lloc.getReservas().add(reserva);
        this.reserves.put(reserva.getId(), reserva);
        usuari.getReserves().add(reserva);
        return Controlador.OK;
}
```

5. **(2,5 punts)** Implementa el mètode de la classe `Controlador`

```
public HashMap<String,ArrayList<Reserva>> reservesPerAny();
```

Aquest mètode crea i retorna un mapa on les claus son Strings representant un any ("2012", per exemple) i els valors son llistes que contenen les reserves en l'any indicat per la clau.

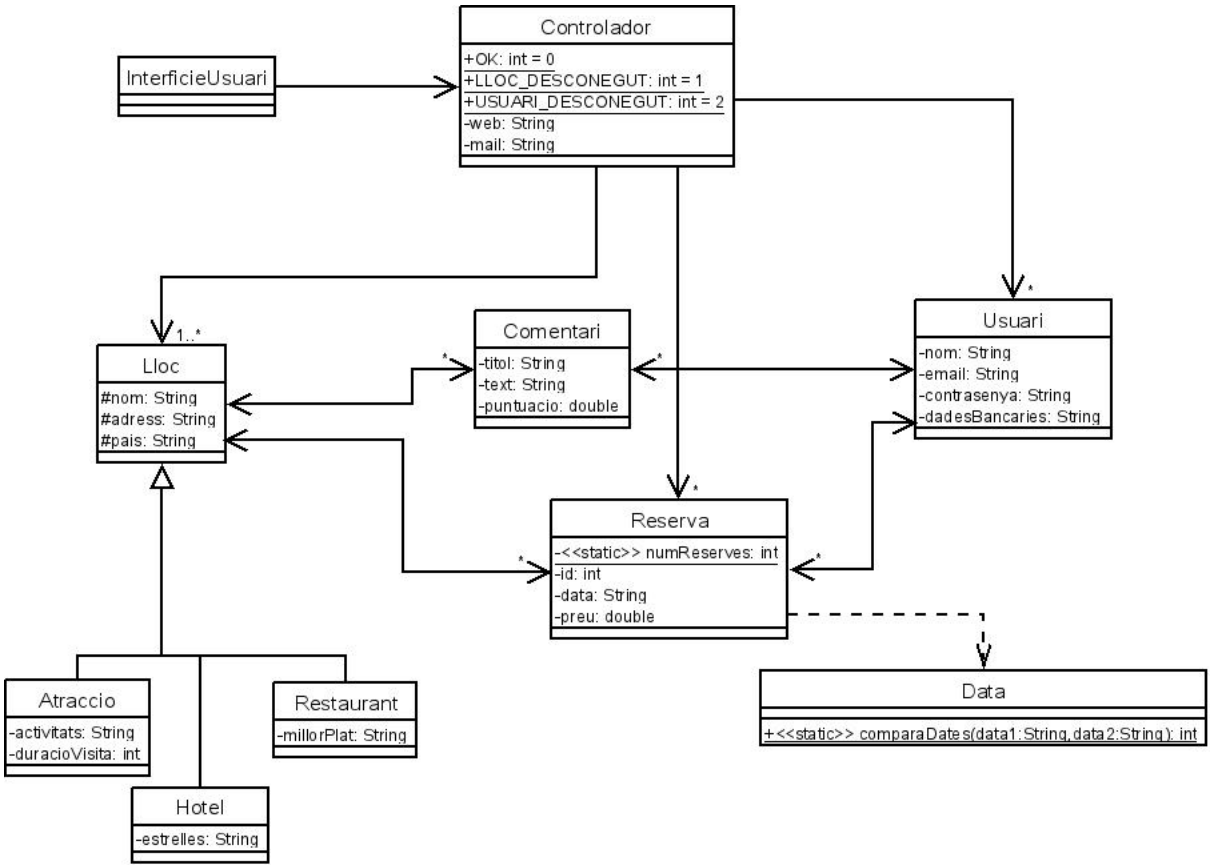
NOTA: Recordeu que la classe String conté el mètode següent:

```
public String substring(int beginIndex, int endIndex);
```

L'especificació d'aquest mètode es "retorna un string que és un substring d'aquest string. El substring comença a l'index `beginIndex` especificat i s'estén fins al caràcter a l'índex `endIndex-1`. Per tant, la longitud del substring és `endIndex-beginIndex`".

```
public HashMap<String, ArrayList<Reserva>> reservesPerAny() {
        HashMap<String, ArrayList<Reserva>> reservesPerAny = new HashMap<>();
```

```
        for(Reserva reserva: this.reserves.values()){
            String any = reserva.getData().substring(0, 4);
            ArrayList<Reserva> reserves = reservesPerAny.get(any);
            if(reserves==null){
                reserves = new ArrayList<>();
                reservesPerAny.put(any, reserves);
            }
            reserves.add(reserva);
        }
        return reservesPerAny;
}
```


Controlador
+OK: int = 0
+LLOC_DESCONEGUT: int = 1
+USUARI_DESCONEGUT: int = 2
-web: String
-mail: String
InterficieUsuari
Lloc
#nom: String
#adress: String
#pais: String
Comentari
-titol: String
-text: String
-puntuacio: double
Usuari
-nom: String
-email: String
-contrasenya: String
-dadesBancaries: String
Reserva
-<<static>> numReserves: int
-id: int
-data: String
-preu: double
Data
+<<static>> comparaDates(data1:String,data2:String): int
Atraccio
-activitats: String
-duracioVisita: int
Restaurant
-millorPlat: String
Hotel
-estrelles: String
1..*
*