OP - Opdracht 4

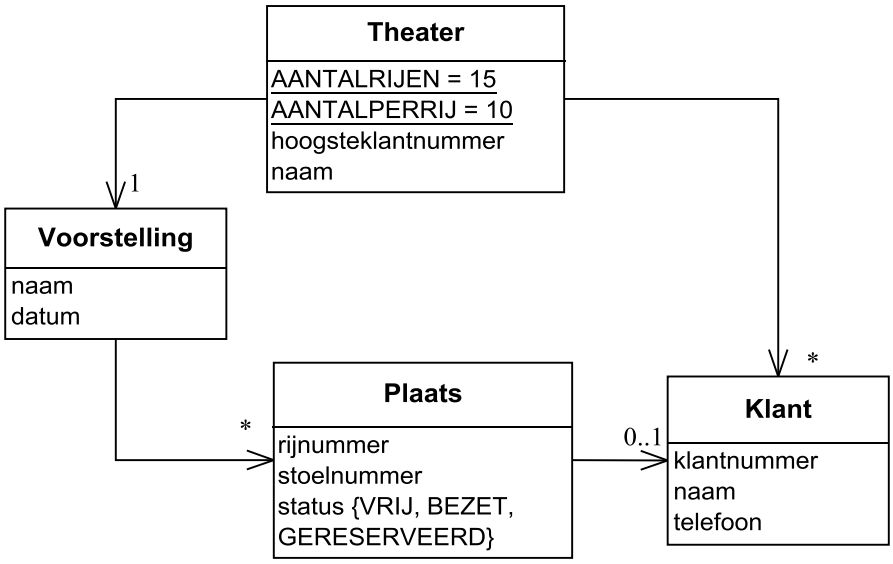
1 Inleiding

In deze opgave gaan we een applicatie maken ter ondersteuning van een klein theater. Dit programma zullen we in de loop van de cursus OpiJ2 verder uitbouwen. Het eindproduct is een programma waarmee kaarten gereserveerd kunnen worden voor verschillende voorstellingen, met een achterliggende database waarin de reserveringen worden opgeslagen en met een grafische weergave van de plaatsen in het theater. Maar zo ver zijn we nog niet. Voorlopig werken we zonder grafische interface. We gebruiken een tekstgeorienteerde interface voor het testen.

Er is in deze versie slechts één voorstelling met een bijbehorende naam en datum. Het plaatsen bestellen voor een voorstelling gaat in een aantal stappen.
− Eerst worden, plaats voor plaats, een aantal plaatsen gereserveerd. Deze plaatsen zijn nu nog niet definitief bezet.
− Vervolgens wordt een bestaande klant geselecteerd of, als het om een nieuwe klant gaat, wordt deze toegevoegd.
− Tenslotte worden alle gereserveerde plaatsen aan de geselecteerde klant gekoppeld en krijgen de plaatsen status bezet.

De transactie kan op ieder moment gestopt worden. Eventueel gereserveerde (maar nog niet definitief bezette) plaatsen worden dan weer vrij gegeven. Om het reserveren te ondersteunen moet het mogelijk zijn van iedere plaats na te gaan of deze vrij, bezet of gereserveerd is.

Het voorlopige klassendiagram dat bij deze omschrijving hoort wordt hier getoond.



Aanvullende informatie bij dit diagram:
- De nummering van de rijen en stoelen begint bij 1.
- Attribuut hoogsteklantnummer bevat het hoogst uitgegeven klantnummer.

2 Opdracht

U dient de applicatie voor het theater te implementeren. Om dit proces in goed banen te leiden is deze opdracht onderverdeeld in een aantal subopdrachten. Vergeet niet om iedere klasse te voorzien van Javadoc en te voorzien van een testklasse die alle public methoden test.
Aan het eind van dit document staat ook een voorbeeldprogrammaatje dat gebruik maakt van de klassen. Dat geeft een voorbeeld hoe het reserveringsproces gaat.

a Implementeer de klasse Klant. Geef deze klasse een constructor waarmee alle attributen via parameters een waarde krijgen. Geef de klasse verder alleen get-methoden voor de attributen, en een methode toString die de volledige klantinformatie als één string teruggeeft.

b Implementeer de klasse Theater. Geef deze klasse een constructor met één parameter, namelijk de naam van het theater.
Eén van de taken van deze klasse is het beheer van alle klanten: het maken van een nieuwe klant, het uitgeven van een nieuw klantnummer, en het opzoeken van een klant op basis van naam en telefoonnummer. Ontwerp en implementeer methoden voor dit beheer van klanten. We laten het verwijderen van klanten buiten beschouwing.

c Ontwerp en implementeer de klasse Plaats. Voor de status van een Plaats gebruiken we een enumeratietype Status met mogelijke waarden VRIJ, GERESERVEERD en BEZET (een plaats is gereserveerd als een klant bezig is met het bestellen van kaarten maar de bestelling nog niet is afgerond).
Aanwijzingen
– Geef de klasse een constructor met als parameters rijnummer en stoelnummer; de beginstatus wordt VRIJ.
– Geef de klasse een methode toString die een stringrepresentatie van de plaats teruggeeft.

d Ontwerp en implementeer de klasse Voorstelling.
Aanwijzingen
– Gebruik een tweedimensionaal array van type Plaats om de zaalbezetting te representeren
– Geef de klasse een methode die de reserveringsstatus van een bepaalde plaats (rijnummer, stoelnummer) kan wijzigen van VRIJ naar GERESERVEERD of omgekeerd.
– Geef de klasse een methode die een gegeven klant plaatst op alle plaatsen met status GERESERVEERD. Die plaatsen krijgen dan status BEZET en een link met de klant.
– Geef de klasse een methode die alle plaatsen met status GERESERVEERD terugzet naar vrij (nodig voor het cancelen van een transactie).
– Breid de klasse Plaats uit met methoden om bovenstaande methoden van Voorstelling te ondersteunen.

e Breid de klasse Theater uit. Geef het theater een voorstelling. De user interface zal in de uiteindelijke versie van dit programma een associatie met Theater bevatten. Geef de klasse Theater daarom een aantal methoden zodat alle beschreven handelingen (reserveren, plaatsen, resetten) door middel van aanroepen op Theater kunnen worden uitgevoerd.

3 Voorbeeld

Volgend programma (te downloaden van yOUlearn) toont het gebruik van het te implementeren theater. Om wat gegevens af te drukken is klasse Theater uitgebreid met een methode getAantalPlaatsen(Status) die het aantal plaatsen met gegeven status bepaalt.

```
package theater;

import theater.Plaats.Status;

/**
 * Voorbeeldprogramma voor de theaterapplicatie.
 */
public class TheaterApplicatie {

  /**
   * Main programma.
   * @param args niet gebruikt
   */
  public static void main(String[] args) {

    Theater theater = new Theater("Chassé Theater");
    theater.nieuweVoorstelling("War Horse", "26-12-2014");
    printInfo(theater);
    // reserveer een aantal plaatsen
    theater.reserveer(3, 3);
    theater.reserveer(3, 4);
    theater.reserveer(3, 5);
    theater.reserveer(3, 6);
    theater.reserveer(-1, 1); //  buiten grenzen
    theater.reserveer(100, 200); // buiten grenzen
    printInfo(theater);
    // plaats klant op gereserveerde plaatsen
    theater.plaatsKlant("Kok", "0678912345");
    printInfo(theater);
    // reserveer een aantal plaatsen
    theater.reserveer(3, 3); // is al bezet
    theater.reserveer(4, 3);
    theater.reserveer(4, 4);
    printInfo(theater);
    // plaats klant op gereserveerde plaatsen
    theater.plaatsKlant("Pootjes", "0654321987");
    printInfo(theater);
    // reserveer een aantal plaatsen
    theater.reserveer(5, 1);
    theater.reserveer(5, 2);
    theater.reserveer(5, 3);
    printInfo(theater);
    // cancel reserveringen
    theater.resetReservering();
    printInfo(theater);
  }
```

```
  private static void printInfo(Theater theater) {
    System.out.println(
      "VRIJ: " + theater.getAantalPlaatsen(Status.VRIJ) +
      " / GERESERVEERD: " +
      theater.getAantalPlaatsen(Status.GERESERVEERD) +
      " / BEZET: " +
      theater.getAantalPlaatsen(Status.BEZET));
  }
}
```

De uitvoer van dit programma is:

```
VRIJ: 150 / GERESERVEERD: 0 / BEZET: 0
VRIJ: 146 / GERESERVEERD: 4 / BEZET: 0
VRIJ: 146 / GERESERVEERD: 0 / BEZET: 4
VRIJ: 144 / GERESERVEERD: 2 / BEZET: 4
VRIJ: 144 / GERESERVEERD: 0 / BEZET: 6
VRIJ: 141 / GERESERVEERD: 3 / BEZET: 6
VRIJ: 144 / GERESERVEERD: 0 / BEZET: 6
```