Examenvragen / opdrachten

cvo leerstad
volwassenenonderwijs

Schooljaar:
Datum:
Afdeling:
Jaar/module:

Naam cursist:
Geb. datum en -plaats:

Evaluatie /

Voertuigenparking

Je bent verantwoordelijk voor een parking. Deze parking bevat verschillende soorten voertuigen, zoals fietsen, moto's en natuurlijk auto's.

Voertuigen delen een groot aantal karakteristieken. Ze worden allemaal gespecificeerd door een model en een type. Ze worden gekarakteriseerd door een vermogen en een kleur. En ze hebben een capaciteit qua passagiers. De kleuren van de voertuigen zijn gelimiteerd en als we een voertuig proberen te maken met een onbestaande kleur krijgen we een IllegalArgumentException exception. De kleuren zijn *comparable* en worden geïdentificeerd aan de hand van hun naam, maar het kan handig zijn om een mapping te hebben met de overeenstemmende Java *Color* objecten.

De volgende operaties zijn mogelijk op voertuigen:

- boolean takePassengers(Collection passengers)

  Deze methode analyseert de kandidaat-passagiers. Als de passagiers iets anders bevat dan een person, of als een kandidaat-passagier zich al in de auto bevindt, worden alle passagiers geweigerd en de methode heeft als return-waarde false. Anderzijds als de overblijvende capaciteit van voertuig groot genoeg is en als de kandidaat-passagiers worden toegevoegd aan het voertuig, is de return waarde true. Deze methode is niet van toepassing op fietsen omdat we veronderstellen dat fietsen geen passagiers kunnen meenemen.

- boolean isMotorized()

  Deze methode geeft true voor gemotoriseerde voertuigen.

- boolean equals(Object vehicle) - int hashCode()

  In het algemeen worden voertuig als equals beschouwd als ze worden bepaald door het zelfde model en hetzelfde type. Maar, om als equals beschouwd te worden moeten ze ook dezelfde kleur hebben.

  Zoals hun kleur zijn voertuigen ook comparable.

Nu je over een voertuig framework beschikt , kun je misschien eens nadenken over methoden die interessant

kunnen zijn voor het beheer van de parking.

- List selectVehicules(String model, String type)

  Deze methode geeft een list terug die alle voertuigen van de parking bevat met hun specifiek model en specifiek type.

- void sortVehicles(Comparator vehicleComparator)

  De methode sorts sorteert de lijst van de voertuigen van de parking. Voertuigen worden gesorteerd als volgend:

  ~ Eerst, de niet-gemotoriseerde voertuigen worden geplaatst voor de anderen

  ~ Ten tweede worden de voertuigen gesorteerd op capaciteit

  ~ Den derde worden de voertuigen gesorteerd op model en daarna op type.

Kunnen we deze comparator gebruiken met een gesorteerde collectie, zoals een TreeSet?Waarom? Plaats je antwoord in commentaar in je Parking klasse

Schrijf testen !

Session – Inscriptions

In deze oefening ga je experimenteren met de noties van comparable en comparator.

Je beschikt over de volgende klassen (zie uml diagram hieronder) :

- Session vertegenwoordigt een cursus
- Inscription vertegenwoordigt een inschrijving die geassocieerd wordt met een Student en een resultaat van een Examination.
- Student stelt een student voor met een naam en een matriculation
- Examination is het resultaat van een examen En je beschikt over de volgende test : o SessionTest die je toelaat om de methoden van de klasse Session te testen.

Je moet de vorige klassen aanpassen om een groen licht te krijgen in de testklasse. Je mag de testklasse NIET aanpassen!

Je taak bestaat er uit om de volgende methoden te implementeren :

- public boolean addInscription(Inscriptioninscription) add van een nieuwe inscription aan een session
  - session kan geen twee inschrijvingen bevatten voor dezelfde student
  - de methode geeft false terug als de inschrijving niet toegevoegd is
- public int getNbrStudents( )

- geeft het aantal inschrijvingen terug

- public double getStudentResult(Stringmatriculation)
  - geeft het resultaat van de student aan de hand van zijn matriculation en geeft -1.0 terug als er geen student is met deze matriculatie.

- public List getInscriptionsList( )
  - geeft een lijst van inschrijvingen terug gesorteerd eerst de naam van de student en als laatste op matriculation.
  - deze lijst moet unmodifiable zijn.

- public List getInscriptionsResultList( )
  - geeft een lijst van inschrijvingen terug gesorteerd eerst op resultaten dan op de naam van de studenten en als laatste op matriculation