Maak een Nieuw project aan voor deze oefening

In dit project maak je voor nu 4 Packages aan:

1) Garage Package: waar je alle Garage en Auto klassen in aanmaakt

2) Interface packages waar je alle interfaces gaat in onderbergen

3) enum Packages

4) app Packages waar je alle verschillende mains zal in onderbergen

**Opgave1 GarageExceptions**

Begin met het uitwerken van de uml en voorzie alle klassen van de nodige constructors en methodes Maak Hierna een garage object aan met een maximum capaciteit van 80 auto's.

**Opdracht 1:**

maak zelf 5 Auto Objecten aan.(3 familyAuto's, een sportsauto en een cabrio -kies zelf de kleuren) aan en plaats ze in de garage met behulp van een addVehicle methode.

Bedenk hoe je dit zal aannemen. De garage is namelijk een draaiend bedrijf en sommige plekken op het begin van hun parkeerplaats komen dus terug vrij. Zorg ervoor dat steeds de eerste mogelijke vrije plaats ingevuld worden zodat de werknemers gespaard worden.

**Opdracht 2:**

Maak 40 Autos aan met behulp van een random genator. Zowel kleur als random acceleratie snelheid. (ze komen van de boot gerold) er mag max 90 gereden worden.

tips:Random, Math.Rand, modulo, switch of if else

**Opdracht 3:**

maak 100 auto's aan en probeer deze aan de garage toe te voegen

**Opdracht 4:**

Vang het probleem uit opdracht 3 op met exceptions en een vriendelijke boodschap terug naar de user

maak een 5de packages aan met Alle eigen Exceptions

Links:

<https://www.geeksforgeeks.org/exceptions-in-java/>  
<https://www.geeksforgeeks.org/types-of-exception-in-java-with-examples/>

HeatException Example op github

**Opgave 2 JAVADOCS**

Becommentarieer je eigen project en maak met behulp van java docs een website voor aan. Voeg aan de main een auteur, versie en beschrijving van je programma toe.

Doe dit ook voor de klassen en voorzie ook de constructors en de methoden van de nodige Commentaar en annotaties.Genereer de javadoc aan de hand van je IDE.

Vind deze bestanden en ga eens zien naar de structuur van gegenereerde bestanden. Bekijk deze en navigeer door je documentatie en zie waar er nog eventuele fouten zitten.

**Opgave 3 finalize**

Bekijk de cabriolet oefening.

1) Implementeer een count-- in de finalize methodes en bekijk wat deze doet met de counters

2) Implementeer een betrouwbare manier om het aantal autos te gaan bijouden die op de dokken staan.

tip: loops en !null

<https://www.tutorialspoint.com/java/lang/object_finalize.htm>

<https://www.geeksforgeeks.org/g-fact-24-finalfinally-and-finalize-in-java/>

**Opgave 4 Equals en HashCode Implementeren**

We gaan een equals methode implementeren voor de auto Objecten. Bij het implementeren van de equals methode is het ook altijd aangeraden de hashCode methode te gaan overschrijven. Hier zullen we dat doen aan de hand van een berekening van de optelling van RGB waarden x de snelheid x een constante.

<https://www.geeksforgeeks.org/equals-hashcode-methods-java/>

<https://www.quora.com/Is-this-the-correct-way-to-implement-an-equals-method-for-a-Person-class>

**Heeeeel duidelijk filmpje:**

[https://www.youtube.com/watch?v=7V3589CReug](https://www.youtube.com/watch?v=7V3589CReug&feature=youtu.be&fbclid=IwAR04MvyDyFR_7nzHwdptGybVT1CfSnHDDXV8-rmgjrc2YNcasqmVS6Fy4M8)

**Opgave 5 LokalClassCarsApp**

Maak een nieuwe mainclasse aan in je app package: LokalClassCarsApp

We maken ook een nieuwe garage aan: ProtoGarage. Deze zal een inner classer ProtoCar bevatten die zelf subklasse is van de Car.

De auto's van deze klasse zullen altijd wit zijn en zullen ook een extra boolean wrapped gaan bevatten, die zal altijd true geïnitialiseerd worden. We krijgen in de garage een lading van prototype wagens binnen. Maak hiervoor een methode aan binnen de garage: protoCars(int amount).

In deze methode ga je een loop aanmaken die het aantal gevraagde auto's aanmaakt en aan de garage toevoegt.

**Opgave 6**

We gaan ons project gaan uitbreiden met een haven. Een haven heeft een naam en een land haven bezitten ook meerdere dokken. In ons geval maken we 2 concrete docken Klassen aan. Een ContainerDock en CarDock.

Het Car dock bevat steeds een max aantal auto's.Deze autos worden tijdens het lossen verdeeld in verschillende carzonesen een garage - 100 plaatsen - voor het verwerken van de wagens.

Deze CarZones krijgen elk een ID(A,B,C,....) mee en kunnen max 200 auto's bevatten elk. Zorg er dus voor dat tijdens het aanmaken van het CarDock de juiste aantal Zones worden aangemaakt.

zie UML Online

**Opgave 7**

We gaan nu het Factory design pattern gaan toepassen op onze Haven oefening. Je kan de oplossing van opgave 4 gebruiken maar we gaan eerst in ons eigen package van Facotory gaan werken.

We gaan eerst om het ons makkelijk te maken en een Enum aanmaken met de Cartypes die we gaan aanmaken. FAMILY,SPORT,CABRIOLET,SUV. Schrijf dit type ook weg in je Car.

Maak een CarFactory aan en ga hier aan de hand van een buildCarMethode Cars gaan terug geven.

Tijdens het aanmaken van deze auto's zullen we het kleur random gaan kiezen, als ook het gewicht. De snelheid mag hier standaard 0 zijn.

<https://howtodoinjava.com/design-patterns/creational/implementing-factory-design-pattern-in-java/>

**Opgave 8**

Net zoals het maken van de Ufo's zullen we dit hier ook het facotory gaan abstraheren, zodat we nog specifiekere auto's kunnen gaan maken.

We gaan dus een Abstract carFactory maken en hier enkele ConcreteCarFactories van maken voor de Verschillende merken(Audi,Porshe,Tesla,Renault). Tijdens het aanMaken van deze auto's zullen we zowel een Random type gaan kiezen( kan terwijl ook een modelnaam instellen) als ook een random BestemmingsLand(BEL,UK,VS,NED,FR)

Tesla cabrio model S Audi Cabrio a3 Renault cabrio Mega Porshe cabrio 911

Tesla family model 3 Audi family a6 Renault family PicassoPorshe family Panamera

Tesla sport model x Audi Sport Rs Renault sport Sprint Porshe sport 911 rs turbo

Tesla suv model y Audi Suv q7 Renault Suv qajar Porshe Suv Cayenne

<https://howtodoinjava.com/design-patterns/creational/abstract-factory-pattern-in-java/>

**Opgave 9**

Container

We gaan ook een Container boot gaan maken die vier verschillende Containers aan boort zal hebben.

- eentje die een object kan vervoeren van gelijk welk type

- de tweede kan enkel Animals vervoeren (haal het animals pakket uit een ander project)

- de derde kan 3 willekeurige objecten gaan vervoeren

- vierde kan 3 auto's tegelijk verschepen