Varför är förändringarna bra?

- LIFOStorage och RAStorage beror på Storable istället för direkt på Car; De ska bero på abstraktioner och inte konkreta implementationer enligt DIP
- Skapa ny abstrakt class StorableCar som subclass till Car och låt Volvo240 och Saab95 vara subclasser till denna nya class. Om vi i framtiden skulle vilja skapa en Car som inte är Storable, så slipper vi göra förändringar i den befintliga koden, vilket följer OCP.
- Skapa en ny abstrakt class Truck som blir superclass till CarTransport och VehiclesWithPlatform, och därmed Scania. Det känns ganska troligt att dessa klasser och eventuella framtida kommer dela en del funktionalitet som är gemensam för alla “Trucks”, genom att från början införa en sådan hierarki behöver vi inte förändra så mycket i framtiden, även detta enligt OCP.
- Logiken för riktning flyttas till ett enum Direction istället för att använda en lista av Strings och ett index för att komma åt rätt riktning. Som det är nu har Vehicle ansvar inte bara för funktionaliteten av ett Vehicle, utan också att definiera vilka riktningar som finns i programmet. Genom att flytta ut ansvaret för riktningarna till ett enum får Vehicle ett mer korrekt avgränsat ansvarsområde, vilket är i enlighet med SRP.
- Flyttar ControlPanel från CarView till CarController; CarView ska visa upp modellen, men ControlPanel visar inte upp modellen på något sätt utan bara kontroller för att styra den. När ControlPanel ligger i CarView finns två anledningar för hur klassen skulle ändras. Dels hur modellen ska visas upp, dels hur den ska styras, vilket inte är bra enligt SRP. Genom att flytta ControlPanel behöver inte CarView längre något CarController komponent, alltså kan vi ta bort "has-A"/komposition pilen åt ena hållet, och åstadkommer på så vis mer “low coupling”.
- Lägg till interfaces HasTurbo, HasAngledPlatform “mellan” CarController och de konkreta implementartionerna, för att följa DIP. Skapa också en VehicleFactory för att kunna “servera” dessa mer abstrakta typer till CarController.
- Skapa en Application class som ansvarar för timern istället för CarController och hanterar positioneringen på skärmen istället för CarView, för att dessa klasser ska få ett mer avgränsat ansvarsområde enligt SRP. Flytta även main metoden till Application klassen, för att den ska ha ansvar för att köra programmet, inte CarController.

Refaktoriseringsplan

- Byt bounds på LIFO och RAStorage.
- Skapa den nya klassen StorableCar som en subklass till Car och låt Saab95 och Volvo240 bli subklasser till den istället för Car. Ersätt Car med StorableCar på de platser där det behövs.
- Skapa ny klass Truck som är subklass till Vehicle och låt VehiclesWithPlatform och CarTransport vara subklasser till den istället för Vehicle.

- Skapa ett enum Direction och implementera funktionalitet för exempelvis getNext() och getPrevious(). Skriv om metoder som behandlar riktning i termer av det Direction enum:et.
- Flytta ControlPanel från CarView till CarController, och låt CarController direkt hantera när användaren använder den genom att “vidarebefordra” till modellen. Ta bort beroendet från CarView till CarController.
- Flytta nödvändiga delar från CarView respektive CarController till en ny Application klass
- Lägg till interfaces HasTurbo, HasAngledPlatform. Skapa också en VehicleFactory för att kunna skapa Vehicles med dessa mer abstrakta typer.