|  |  |  |  |
| --- | --- | --- | --- |
| **Namn & egenskaper** | **Klassens/Metodens ansvarsområden** | **Klassens/Metodens förväntade ansvarsområden** | **Tillvägagång för att implementera ändringarna** |
| ***Biltransport***  ext Car  imp Transporter<Car> | Håller listan med lastade fordon, rampUp och max-last-kapaciteten. | Bör hantera allt Ferry och Scania har gemensamt. Bör endast innehålla Biltransport-specifika fält/metoder. | **Ta bort:**  **attachVehicle() & detachVehicle() incrementSpeed() & decrementSpeed()** |
| Konstruktor  (int,int,Color,String) | Kallar superkonstruktorn & instansierar listan med fordon. | Bör använda sig av Cars konstruktor och tilldela Biltransport-specifika egenskaper. | Ingen |
| *attachVehicle*(Car) : void  (från Transporter) | Abstrakt | Bör vara abstrakt då Scania och Ferry kan ha olika sätt att lasta på fordon.  Onödig då den ärvs av Transporter. | Ta bort |
| *detachVehicle*() : void  (från Transporter) | Abstrakt | Bör vara abstrakt då Scania och Ferry kan ha olika sätt att lasta av fordon.  Onödig då den ärvs av Transporter. | Ta bort |
| turnLeft() : void | Kallar på super.turnLeft() och foreach(Car::turnLeft) | Bör använda sig av den redan existerande metoden turnLeft i Car. | Ingen |
| turnRight() : void | Kallar på superns turnRight och foreach(Car::turnRight) | Bör använda sig av den redan existerande metoden turnRight i Car. | Ingen |
| move() : void | Kollar om isRampUp och isEngineOn innan den flyttar sig och alla lastade bilar. | Bör använda sig av den redan existerande metoden move i Car – förutsatt att rampen är uppfälld och motorn är igång. | Ingen |
| speedFactor() : double | Enginepower / 100 | Själv-definierad. | Ingen |
| incrementSpeed(double) : void | Math.min(...) | Själv-definierad – helst icke negativ.  Ska Scania och Ferry ha samma imp.? | Ta bort och flytta till Scania |
| decrementSpeed(double) : void | Math.max(... , 0) | Själv-definierad – helst icke negativ.  Ska Scania och Ferry ha samma imp.? | Ta bort och flytta till Scania |
| liftRamp() : void | rampUp = true | Fanns inga krav att fordonet måste vara stillastående för att lyfta rampen. | Ingen |
| lowerRamp() : void | Kollar om inte isEngineOn då sänks rampen. | Fanns krav att fordonet måste vara avstängd för att sänka rampen. | Ingen |
| canAttach(Car) : boolean | return (car instanceOf Transportable); | Inget efterfrågat men underlättar för Biltransporterna. | Ingen |

Labb-2-2: Klass-/Metod-överblick

|  |  |  |  |
| --- | --- | --- | --- |
| **Namn & egenskaper** | **Klassens/Metodens ansvarsområden** | **Klassens/Metodens förväntade ansvarsområden** | **Tillvägagång för att implementera ändringarna** |
| **Scania**  ext Biltransport  imp Transportable | Har inga fält utöver superklassen. Override:ar nödvändiga metoder. | Bör inte vara Transportable då Ferry inte är närvarande. Bör override:a superklassens metoder. | **Fixa SRP-problemet i attachVehicle()** |
| Konstruktor  () | Kallar superklassens konstruktor med redan satta värden. | Inget angivet – bör det vara förbestämda värden? | Lägga till överladdad konstruktor? |
| attachVehicle(Car) : void | Lastar på ett fordon OM: flaket är nere, fordonet är Transportable, fordonet är inte redan lastat och fordonet är nära och är riktat åt samma håll.  Flyttar fordonet till Scanias pos.  Har lite återkommande kod och kollar möjligtvis för många saker. | Inget krav att kolla riktning – endast position. | Bryter mot Single Responsibility Principle:  Bryt ut och skapa metoder som: isNearby() isParallel() OM \_\_\_\_\_\_\_\_\_ DÅ:  Flytta och Lasta fordonet. |
| detachVehicle(Car) : void | Om fordonet inte är null: anropa detachVehicle() nedan | Ska Scania ha en sådan metod/implementation? | Möjligtvis kolla om fordonet existerar på flaket och KAN lastas av. |
| detachVehicle() : void | Om rampen är nere, ta sista fordonet i listan, flytta fordonet och ta bort från listan. | Inget krav att lasta av fordon beroende på riktning. | Ingen |

|  |  |  |  |
| --- | --- | --- | --- |
| **Namn & egenskaper** | **Klassens/Metodens ansvarsområden** | **Klassens/Metodens förväntade ansvarsområden** | **Tillvägagång för att implementera ändringarna** |
| **Movable**  <<Interface>> | Interface för flyttbara ting. | Givet Interface | **Ingen** |
| move() : void | Definieras där interfacet implementeras | Ska override:as av implementerande klasser. | Ingen |
| turnLeft() : void | Definieras där interfacet implementeras | Ska override:as av implementerande klasser. | Ingen |
| turnRight() : void | Definieras där interfacet implementeras | Ska override:as av implementerande klasser. | Ingen |

|  |  |  |  |
| --- | --- | --- | --- |
| **Namn & egenskaper** | **Klassens/Metodens ansvarsområden** | **Klassens/Metodens förväntade ansvarsområden** | **Tillvägagång för att implementera ändringarna** |
| **Transporter<T>**  <<Interface>> | Interface för objekt som kan transportera andra objekt av typen T. | Typen T var inte efterfrågat men ju mer abstraktion - det så bättre. | **Ingen** |
| attachVehicle(T) : void | Definieras där interfacet implementeras | Givet (utan T). | Ingen |
| detachVehicle(T) : void | Definieras där interfacet implementeras | Givet (utan T). | Ingen |
| detachVehicle() : void | Definieras där interfacet implementeras | Frivillig? | Ingen |
| canAttach(T) : boolean | Definieras där interfacet implementeras | Givet (utan T). | Ingen |

|  |  |  |  |
| --- | --- | --- | --- |
| **Namn & egenskaper** | **Klassens/Metodens ansvarsområden** | **Klassens/Metodens förväntade ansvarsområden** | **Tillvägagång för att implementera ändringarna** |
| **Transportable**  <<Interface>> | Interface för objekt som kan bli transporterade. | Givet. | **Ingen** |

|  |  |  |  |
| --- | --- | --- | --- |
| **Namn & egenskaper** | **Klassens/Metodens ansvarsområden** | **Klassens/Metodens förväntade ansvarsområden** | **Tillvägagång för att implementera ändringarna** |
| **Car**  imp Movable | Givet | Givet | **Ingen** |
| Konstruktor  (int, double, double, Color, String) | Givet | Givet | Ingen |
| startEngine() : void | EngineOn = True;  CurrentSpeed = 0.1; | Startar motorn. | Ingen |
| stopEngine() : void | EngineOn = false;  CurrentSpeed = 0; | Stoppar motorn. | Ingen |
| move() : void | Flyttar bilen om engineOn och beroende på direction, position och currentSpeed. | Bör flytta bilen utifrån var den befinner sig och dess nuvarande hastighet. | Ingen |
| turnLeft() : void | Utifrån nuvarande riktning, svänga – 90\*. | Bör svänga bilen åt vänster beroende på vilket håll som bilen är riktad i nuvarande läge. | Ingen |
| turnRight : void | Utifrån nuvarande riktning, svänga + 90\*. | Bör svänga bilen åt höger beroende på vilket håll som bilen är riktad i nuvarande läge. | Ingen |
| *speedFactor() : double* | Abstrakt | Skall definieras i subklasserna. | Ingen |
| *incrementSpeed(double) : void* | Abstrakt | Skall definieras i subklasserna. | Ingen |
| *decrementSpeed(double) : void* | Abstrakt | Skall definieras i subklasserna. | Ingen |
| gas(double) : void | Kollar  0 <= gasAmount <= 1  innan incrementSpeed kallas. | Bör öka hastigheten så länge mängden gas är mellan 0 och 1. | Ingen |
| brake(double) : void | Kollar  0 >= brakeAmount <= 1  innan decrementSpeed kallas. | Bör minska hastigheten så länge mängden gas är mellan 0 och 1. | Ingen |

|  |  |  |  |
| --- | --- | --- | --- |
| **Namn & egenskaper** | **Klassens/Metodens ansvarsområden** | **Klassens/Metodens förväntade ansvarsområden** | **Tillvägagång för att implementera ändringarna** |
| **Saab95**  ext **Car**  imp **Transportable** | Givet. | Givet. | **Ingen** |

|  |  |  |  |
| --- | --- | --- | --- |
| **Namn & egenskaper** | **Klassens/Metodens ansvarsområden** | **Klassens/Metodens förväntade ansvarsområden** | **Tillvägagång för att implementera ändringarna** |
| **Volvo240**  ext **Car**  imp **Transportable** | Givet. | Givet. | **Ingen** |

|  |  |  |  |
| --- | --- | --- | --- |
| **Namn & egenskaper** | **Klassens/Metodens ansvarsområden** | **Klassens/Metodens förväntade ansvarsområden** | **Tillvägagång för att implementera ändringarna** |
| **Ferry**  ext **Biltransport** | Inte relevant. | Inte relevant. | **Ingen** |

|  |  |  |  |
| --- | --- | --- | --- |
| **Namn & egenskaper** | **Klassens/Metodens ansvarsområden** | **Klassens/Metodens förväntade ansvarsområden** | **Tillvägagång för att implementera ändringarna** |
| **CarController** | Har bland annat;  CarView och listan med bilar. | Följer möjligtvis inte MVC – men det gäller inte nu. | **Skicka andra parametrar till DrawPanel och tillämpa dekomposition på checkCollision.** |
| main(String[]) : void | Skapar en CarController, lägger till bilar, instansierar CarView och startar timern. | Följer möjligtvis inte MVC – men det gäller inte nu.  Skulle kunna anropa en Factory. | Ingen |
| checkCollision(Car) : void | Kontrollerar positionen om den yttre-väggen nuddas beroende på riktning, loopar för varje fordon, om fordonet kolliderar eller inte. OM kollision DÅ ändras riktning. Positionen beräknas med förbestämda pixlar. | Kan ses att metoden utför för många saker.  Bör bil-dimensionerna vara bekanta i CarController? | Bryter mot Single Responsibility Principle:  Bryt ut och skapa metoder som: collidingWithBorders() collidingWithVehicles() OM \_\_\_\_ ELLER \_\_\_ DÅ:  Rotera fordonet 180\*.  (Skapa en uTurn() i Car?)  Bryter mot Separation of Concern:  Skapa metod i DrawPanel:  collidingWithBorders(Point) : boolean |
| gas(int) : void | Kallar gas-metoden för varje bil. | Den givna metoden. | Ingen |
| brake(int) : void | Kallar broms-metoden för varje bil. | Den givna metoden. | Ingen |
| **TimerListener**  imp ActionListener |  |  |  |
| actionPerformed(ActionEvent) : void | För varje bil kollas kollision innan anropet av move(). Därefter anropas DrawPanels moveIt() med fordonet. | Möjligtvis inget fel här – men DrawPanel får ett fordon. Vad sägs om att ge fordonets klassnamn som en sträng och dess position? | Bryter mot Separation of Concern:  Anropar en UI-klass med våra fordon och skapar onödiga beroenden.  Anropa DrawPanel med klassnamns-strängen och positionen. |

|  |  |  |  |
| --- | --- | --- | --- |
| **Namn & egenskaper** | **Klassens/Metodens ansvarsområden** | **Klassens/Metodens förväntade ansvarsområden** | **Tillvägagång för att implementera ändringarna** |
| **CarView**  ext JFrame | Har: CarController & DrawPanel | Givet. Följer MVC? | **Hantera inga bilar, bilarna ligger i CarControllern. Den borde hantera det!** |
| Konstruktor  (String, CarController) | Givet | Givet. | Ingen |
| **initComponents (String) : void** | Givet | Givet. | Ingen |
| gasButton.addActionListener() | carC.gas(gasAmount); | Frivillig? | Ingen |
| brakeButton.addActionListener() | carC.brake(gasAmount); | Givet (utan T). | Ingen |
| stopButton.addActionListener() | Foreach carC.car -> stopEngine() | Onödiga användningar/beroenden av Car. | Skapa metod i CarController: stopEngines() – stops all engines. |
| startButton.addActionListener() | Foreach carC.car -> startEngine() | Onödiga användningar/beroenden av Car. | Skapa metod i CarController: startEngines() – starts all engines. |
| turboOnButton.addActionListener() | Foreach carC.car OM saab DÅ turboOn(). | Onödiga användningar/beroenden av Car och Saab. | Skapa metod i CarController: turbosOn () – turns on all turbos. |
| turboOffButton.addActionListener() | Foreach carC.car OM saab DÅ turboOff(). | Onödiga användningar/beroenden av Car och Saab. | Skapa metod i CarController: turbosOff () – turns off all turbos. |
| lowerBedButton.addActionListener() | Foreach carC.car OM scania DÅ lowerRamp(). | Onödiga användningar/beroenden av Car och Scania | Skapa metod i CarController: lowerBeds () – lowers all beds. |
| liftBedButton.addActionListener() | Foreach carC.car OM scania DÅ liftRamp(). | Onödiga användningar/beroenden av Car och Scania. | Skapa metod i CarController: liftBeds () – lifts all beds. |

|  |  |  |  |
| --- | --- | --- | --- |
| **Namn & egenskaper** | **Klassens/Metodens ansvarsområden** | **Klassens/Metodens förväntade ansvarsområden** | **Tillvägagång för att implementera ändringarna** |
| **DrawPanel**  ext JPanel | Har: 3 bufferedImages 3 Points | Måste generaliseras! | **Implementera en ny klass för att underlätta lagringen av points och bilder. Ändra konstruktorn, ta bort/ ersätt moveIt och ändra paint-metoden.** |
| Konstruktor  (int , int) | Testar att ladda bilderna och tilldelar våra 3 bufferedImages. | Måste generaliseras! Måste lägga till en ny ImageIO.read om vi vill läsa in ett nytt fordon! | Ta bort beroendet av Image och flytta det till en annan metod. |
| moveIt(Car) : void | Tar in ett fordon och ändrar klassens points till fordonets. | Måste generaliseras!  Ska inte hantera fordon överhuvudtaget. | Ta bort / Ersätt med annat namn och parametrar för att uppdatera fordonens positioner. |
| paintComponent(Graphics) : void | Anropar super.paintComponent och därefter ritar ut de 3 bilarnas bilder. | Måste generaliseras! | Bör endast hantera bilder och points abstrakt - utan att uppge fordonstyp. |

**Flytta domänlogik från vår kontroller "CarView" till vår modell "CarController":**

stop/start-Engines(), turbos-On/Off(), lower/lift-beds()

**Person 1:** Anropar CarController vid actionPerformed (metoderna ovan)

**Person 2:**

Implementera metoderna ovan i CarController

**Generalisera DrawPanels hantering och lagring av bilder och points med hjälp av en ny klass "VehicleImage":**  
**Person 1:**

Implementera klassen "VehicleImage" med fälten image : BufferedImage & point : Point

I CarController, anropa en updateImages() med listan av objektens (bilarnas) klassnamn som en sträng och dess position i DrawPanel.

**Person 2:**

I DrawPanel, skapa metoden updateImages() som får parametrar för vilka bildfiler som ska laddas in och var de ska positioneras. Som för varje sträng/point kallar på en metod i DrawPanel "addImage(string,point)".

I DrawPanel, skapa metoden addImage(string, point) som testar att ladda in en bildfil med objektets klassnamn och skapar sedan ett objekt av VehicleImage med parametrarna (Image, Point). (flyttad från konstruktorn)

I DrawPanel, skapa en lista med VehicleImages som sedan ska användas i paintComponent: foreach VehicleImage -> g.drawImage(Image, point.x, point.y, null) etc....