**Автомобилен парк**

**Общ преглед**

Във вашата фирма постъпва проект за създаване на приложение, обслужващо автомобилен парк.

Вашият софтуер трябва да описва **части** (Part) и **коли (**Car**).**

Tрябва да реализирате функционалност, която да позволява добавяне на части и коли, а в последствие каране на автомобили – всичко това ще работи чрез **команди**, които вие ще получавате. Поредицата от команди приключва с „**End**”. За ваше удобство ще получите готов Program.cs файл, а вие ще трябва да реализирате само необходимите класове в Part.cs и Car.cs

**Подзадача 1: Структура на Част – 15 точки**

**Part**

Всички части имат име и цена:

name – низ, съставен от малки и големи латински букви, както и цифри, но без други специални знаци

               Уникална стойност, която различава всички части

price – число с плаваща запетая

Реализирайте конструктори:

* **Part(string name, double price)** – този конструктор трябва да приема името и цената на частта.
* **Part(string name)** – този конструктор трябва да приема името на частта и да постави базисна цена на стойност 25лв.

За справка вижте следната схема за  Part.cs:

|  |
| --- |
| Part.cs |
| public Part(string name, double price)  {      //TODO: Добавете вашия код тук …  }    public Part(string name)  {     //TODO: Добавете вашия код тук …  } |

**Команди за добавяне на продукти**

Трябва да реализирате следните две команди за добавяне на продукти:

* **CreatePart** <**име**> <**цена**> - тази команда има за цел да добави част с нейното име и цена.
* **CreatePart**<**име**>-тази команда има за цел да добави част с нейното име и базисна цена, на стойност 25лв.

**Команда за извеждане на информация**

Вашето приложение във всеки един момент може да получи заявка да отпечата информация за част. Командата за това е следната:

* **PrintPartByName**<**име**> - отпечатва информация за част във формат:  
  **->**<**име**> - <**цена**>   Цената, следва да бъде форматирана до втория знак след десетичния разделител

Тази команда ще получава винаги валидни и съществуващи имена на части. За успешна реализация трябва да реализирате ваша версия на **ToString()**метода за класа **Part.**

**Подзадача 2: Структура на Кола – 15 точки**

**Car**

Всяка кола има: производител, модел, товароносимост списък от части, количество гориво.

manufacturer– низ, съставен от малки и големи латински букви, както и цифри, но без други специални знаци

model– низ, съставен от малки и големи латински букви, както и цифри, но без други специални знаци

               Уникална стойност, която различава всички коли

loadCapacity– число с плаваща запетая

parts – списък от **Part**

fuel – всяка кола стартира с базисна стойност – 100 литра гориво.

Всяка кола си има и цена, която се равнява на сумата на цените на всички части в нея.

**Част от логиката на приложението е да се пази в променлива общия брой на всички поръчани коли. За целта трябва да бъде добавен брояч, който да бъде увеличаван с 1 за всяка поръчана кола.**

Трябва да реализирате конструктор:

* **Car(string manufacturer, string model, double loadCapacity)** – този конструктор трябва да приема производител, модел и товароносимост.

За справка вижте следната схема за  Car.cs:

|  |
| --- |
| Car.cs |
| public Car(string manufacturer, string model, double loadCapacity)    {        //TODO: Добавете вашия код тук …    } |

**Команда за поръчване на кола**

* **OrderCar**<**производител**> <**модел**><**товароносимост**> - тази команда има за цел да добави кола с производител, модел и товароностимост. Уверете се, че всички полета има валидни състояния!

**Команда за извеждане на информация**

Вашето приложение във всеки един момент може да получи заявка да отпечата информация за кола. Командата за това е следната:

* **PrintCarByModel<модел> -**отпечатва информация за кола във формат:  
  <МОДЕЛ> made by <Производител>

Available parts:

<part1>

<part2>…

With total price of: <цена> lv.

Тази команда ще получава винаги валидни и съществуващи имена на коли. За успешна реализация трябва да реализирате ваша версия на **ToString()**метода за класа **Car.**Oчаква се модела да бъде изписан с главни букви, цената форматирана до два знака след десетичния разделител. Останалата информация не следва да бъде форматирана.

**Подзадача 3: Логика – 40 точки**

**Kоманди**

Вашето приложение трябва да реализира следните команди:

* **AddPartToCar**<**име на кола**><**име на част>** -Тази команда добавя дадена част към определената кола
* **AddMultiplePartsToCar**<**име на кола**><**име на част>**<**име на част>...**Тази команда добавя всички посочени части към определената кола.
* **RemovePartFromCar**<**име на кола**><**име на част>**- Тази команда премахва посочената част от списъка с части от колата.
* **PrintCarPrice**<**име на кола**> - Изпечатване на цената на посочената кола във формат:

<модел на колата> with price = <цена на колата>

* **PrintPartPrice**<**име на част**> - Изпечатване на цената на посочената част във формат:

<име на частта> with price = <цена на частта>

* **ContainsPart**<**име на кола**> <**име на част**> - Проверява дали в списъка от части на дадената кола се съдържа част с такова име. Тази команда трябва да извежда **true**или **false** в зависимост от резултата.

**Подзадача 4: Бонус логика – 20 точки**

Трябва да реализирате още няколко команди:

* **DriveCar**<**име на кола**> <**double дистанция**>**-**Следва да „придвижите“ колата. Придвижването става по следния начин – текущото гориво трябва да бъде намалено с **loadCapacity \* 0.2 \* distance**
* **PrintCarMostExpencivePart**<**име на кола**> **–**Изпечатва най-скъпата част от посочената кола
* **PrintCarPartsWithPriceAbove**<**име на кола**><**цена**> **-**Изпечатва всички части, налични в посочена кола с цена над дадената във формат:

<модел на кола> with parts more expensive than <дадената цена>: "

Part1 Part2 Part3…

**За безпроблемната работа на всички изброени команди от 3 и 4 подзадача трябва да реализирате Car.cs и Part.cs, по аналогичен начин на показаното по-долу:**

|  |
| --- |
| Car.cs |
| public Car(string manufacturer, string model, double loadCapacity)  {       //TODO: Добавете вашия код тук …  }    public double GetCarPrice()  {       //TODO: Добавете вашия код тук …  }    public ReadOnlyCollection<Part> Parts  {       //TODO: Добавете вашия код тук …  }    public void AddPart(Part part)  {       //TODO: Добавете вашия код тук …  }    public void AddMultipleParts(List<Part> passedParts)  {       //TODO: Добавете вашия код тук …  }    public void RemovePartByName(string name)  {       //TODO: Добавете вашия код тук …  }    public List<Part> GetPartsWithPriceAbove(double price)  {      //TODO: Добавете вашия код тук …  }    public Part GetMostExpensivePart()  {       //TODO: Добавете вашия код тук …  }    public static int OrdersCount  {       //TODO: Добавете вашия код тук …  }    public void Drive(double distance)  {       //TODO: Добавете вашия код тук …  }    public bool ContainsPart(string partName)  {       //TODO: Добавете вашия код тук …  }    public override string ToString()  {       //TODO: Добавете вашия код тук … } |

|  |
| --- |
| Part.cs |
| public Part(string name, double price)  {      //TODO: Добавете вашия код тук …  }    public Part(string manufacturer)  {     //TODO: Добавете вашия код тук …  }    public override string ToString()  {     //TODO: Добавете вашия код тук …  } |

**Забележка:**Освен горепосочените методи трябва да реализирате и необходимите свойства за всеки от класовете. Възможно е да е удачно да реализирате допълнителни полета, свойства и методи, по ваша преценка.

**Подзадача 5: Валидация – 10 точки**

Освен всичко останало вие трябва да направите и валидация!

Не допускайте създаването на:

* Част с цена под 0.00, message => Price should be positive!
* Част с име по-късо от 5 символа, message => Invalid part name!
* Кола с име на производител, по-късо  от 3 символа , message => Invalid manufacturer name!
* Кола с име на модел, по-късо  от 3 символа , message => Invalid model name!
* Кола с товароносимост под 0.00 , message => Invalid load capacity!

При невалидни данни хвърлете грешка от тип **ArgumentException(“message”)**

**Допълнително информация свързана с валидацията**

Командите **OrderCar, CreatePart**могат да съдържат информация, която да наруши правилата от по-горе. Въпреки това имената на колите и частите в тези команди **винаги** ще бъдат низове, съдържащи малки и/или големи латински букви и цифри, без каквито и да е други символи.

На**PrintPartByName, PrintCarByName, AddPartToCar,  AddMultiplePartsToCar, RemovePartFromCar, PrintCarPrice, PrintPartPrice, CarContainsPart, PrintCarMostExpencivePart, PrintCarPartsWithPriceAbove, DriveCar ВИНАГИ**ще се подават кола/част, която съществува.

**Ако получите команда, която се опитва да извърши операция, която нарушава валидацията, хвърлете изключение с текст, посочен за съответната валидация!**

**Вход / Изход**

**Вход**

* Програмата ще получава множество редове с информация. Всеки ред представлява команда. Самият вход се обработва изцяло от примерния Program.cs.
* Всички команди приключват с въвеждането на End

**Изход**

За някои от командите не е нужно да извеждате нищо. За всички останали изпечатването ще ви бъде дадено в Program.cs, освен ToString() методите, които са оставени на вас.

**Ограничения**

* Всички цели числа ще бъдат в диапазона **–10000** до **+10000**
* Всички числа с плаваща запетая ще бъдат въвеждани с до два знака след запетаята.
* Всички имена няма да съдържат интервал

**Примери**

|  |  |
| --- | --- |
| **Вход** | **Изход** |
| CreatePart carSeat 100  CreatePart carBackSeat 70  CreatePart carFrontSeat  PrintPartByName carSeat  PrintPartByName carFrontSeat  End | -> carSeat - 100.00  -> carFrontSeat - 25.00 |
| CreatePart carSeat 100  OrderCar Mercedes clk 1500  AddPartToCar clk carSeat  PrintCarByModel clk  End | CLK made by Mercedes  Available parts:  -> carSeat - 100.00  With total price of: 100.00 lv. |
| CreatePart partName 200  CreatePart partName -20  OrderCar BMWWW superCar 50  PrintCarByModel superCar  PrintPartByName partName  AddPartToCar superCar partName  CreatePart partName2  CreatePart partName3 56.8  AddMultiplePartsToCar superCar partName2 partName3  PrintCarPrice superCar  RemovePartFromCar superCar partName3  PrintCarPrice superCar  PrintPartPrice partName2  CarContainsPart superCar partName3  CarContainsPart superCar partName2  PrintCarMostExpencivePart superCar  AddPartToCar superCar partName3  PrintCarPartsWithPriceAbove superCar 50  DriveCar superCar 1  DriveCar superCar 222020  PrintTotalCarOrders  PrintCarByModel superCar  End | Price should be positive!  SUPERCAR made by BMWWW  Available parts:  With total price of: 0.00 lv.  -> partName - 200.00  superCar with price = 281.80  superCar with price = 225.00  partName2 with price = 25.00  False  True  -> partName - 200.00  superCar with parts more expensive than 50:  -> partName - 200.00-> partName3 - 56.80  Drive not possible!  1  SUPERCAR made by BMWWW  Available parts:  -> partName - 200.00  -> partName2 - 25.00  -> partName3 - 56.80  With total price of: 281.80 lv. |

**Точки**

Разбивката по подзадачи е следната:

1. **15** точки, като трябва да имате и задължително реализиран ToString() метод
2. **15** точки, като трябва да имате и задължително реализиран ToString() метод
3. **40** точки
4. **20** точки
5. **10** точки

Общ брой точки: **100**