**Parking System**

**Available from**: Sunday, 23 October 2022, 10:00  
 **Requested files**: Program.cs, ParkingSpot.cs, BusParkingSpot.cs, CarParkingSpot.cs, SubscriptionParkingSpot.cs, ParkingInterval.cs, ParkingController.cs ( [Download](https://it-kariera.mon.bg/e-learning/mod/vpl/views/downloadrequiredfiles.php?id=241))  
**Type of work**:  Individual work  
 **Настройки на оценките**: Максимална оценка: 100  
 **Run**: Не.  **Evaluate**: Да. **Evaluate just on submission**: Да  
**Automatic grade**: Да.

**Parking System**

**Преглед на задачата**

Паркингите са едно изключително хаотично съоражение. Все някой влиза, все някой излиза, но някак си все няма място. След като на Гошо му писна на неговия платен паркинг да цари безкраен хаос, той реши, че трябва да ви постави специалната задача да напишете софтуер за паркинга му. Софтуерът трябва да поддържа основните функционалности на паркинга и да е способен да създава различни справки.

**Забележка: Преди да започнете със задачата, за да нямате проблеми със системата премахнете namespaces от всеки клас. Проверяващата система работи с версия .NET Core 3.1.**

**Problem 1.   Структура**

**ParkingSpot**

Мястото за паркиране (**ParkingSpot**) е базов клас за всички видове места за паркиране и не трябва да може да се инстанцира.

**Data**

Всяко място разполага с идентификатор, информация дали е заето или не, вид, както и цена за започнат час престой. За всяко място трябва да се съхранява и списък с отделните паркирания върху това място (обекти от клас **ParkingInterval**)

Id – цяло число

Occupied – булева стойност (True, ако мястото е заето и False, в противен случай)

Type – текстов низ

Price – реално число

* Цената на час не може да бъдат отрицателна или равна на 0.

Съобщение: Parking price cannot be less or equal to 0!

* При подаване на невалидна стойност хвърлете изключение – ArgumentException с конкретното съобщение.

**Конструктор**

Това са параметрите за конструктора:

**int id, bool occupied, string type, double price**

В рамките на конструктора не забравяйте да създадете инстанция на списъка за паркиранията.

**Поведение**

**override string ToString()**

**> Parking Spot #{Id}**

**> Occupied: {Occupied}**

**> Type: {Type}**

**> Price per hour: {Price} BGN**

Цената за час трябва да излиза с точно два знака след запетаята.

**Дъщерни класове**

**CarParkingSpot**

Този клас описва място за паркиране, предназначено за кола. Неговият вид автоматично се задава на “car”.

**Конструктор**

Това са параметрите за конструктора:

**int id, bool occupied, double price**

Конструкторът трябва да извиква конструктора на родителския клас с подходящи параметри.

**BusParkingSpot**

Този клас описва място за паркиране, предназначено за автобуси. Неговият вид автоматично се задава на “bus”.

**Конструктор**

Това са параметрите за конструктора:

**int id, bool occupied, double price**

Конструкторът трябва да извиква конструктора на родителския клас с подходящи параметри.

**SubscriptionParkingSpot**

Този клас описва място за паркиране, предназначено за абонати. Неговият вид автоматично се задава на “subscription”. Мястото за абонати трябва да допуска паркиране само и единствено на своя абонат. За тази цел създайте допълнително свойство:

RegistrationPlate - текстов низ

* Регистрационния номер не трябва да бъде null или празен низ.

Съобщение: Registration plate can’t be null or empty!

* При подаване на невалидна стойност хвърлете изключение – ArgumentException с конкретното съобщение.

**Конструктор**

Това са параметрите за конструктора:

**int id, bool occupied, double price, string registrationPlate**

Конструкторът трябва да извиква конструктора на родителския клас с подходящи параметри.

**Parking Interval**

Всяко паркиране на дадено място се описва чрез ParkingInterval обект. За всеки интервал на паркиране се запазва номера на мястото, регистрационния номер на паркираното превозно средство и часовете, за които средството е паркирано и печалбата от паркирането.

ParkingSpot – обект от класа ParkingSpot

RegistrationPlate – текстов низ

HoursParked – цяло число

Revenue – реално число

Стойността на свойството Revenue трябва да се пресмята като цената за паркомястото се умножи по броя часове. Ако паркомястото е абонаментно, стойността не Revenue трябва да се приема за нула.

* Регистрационния номер не трябва да бъде null или празен низ.

Съобщение: Registration plate can’t be null or empty!

* Часовете, в които автомобила е паркиран не трябва да бъдат 0 или отрицателно число.

Съобщение: Hours parked can’t be zero or negative!

* При подаване на невалидна стойност хвърлете изключение – ArgumentException с конкретното съобщение.

**Конструктор**

Това са параметрите за конструктора:

**ParkingSpot parkingSpot, string registrationPlate, int hoursParked**

**Методи за класа ParkingSpot**

**bool ParkVehicle(string registrationPlate, int hoursParked, string type)**

В метода трябва да проверите дали мястото за паркиране е свободно и е предназначено за съответния вид превозно средство. Ако това място е заето или не е подходящо за превозното средство, не може да се паркира на него и метода връща false. В противен случай го добавете в колекцията с реализирани паркирания.

**List<ParkingInterval> GetAllParkingIntervalsByRegistrationPlate(string registrationPlate)**

Методът трябва да върне списък от всички реализирани паркирания от дадено превозно средство по неговия регистрационен номер върху конкретното място. Ако не съществуват паркирания, да се върне празен списък.

**double CalculateTotal()**

Методът трябва да върне сумата от приходите от всички реализирани паркирания върху мястото.

Забележка: ParkVehicle и CalculateTotal трябва да се презаписват (override) от класа SubscriptionParkingSpot

**Методи за класа ParkingInterval**

**override string ToString()**

> Parking Spot #{ParklingSpot.Id}

> RegistrationPlate: {RegistrationPlate}

> HoursParked: {HoursParked}

> Revenue: {Revenue} BGN

Цената за интервала на паркиране трябва да излиза с точно два знака след запетаята.

**Забележка: Гошо не е платил достатъчно, поради тази причина в текущата версия на софтуера няма да включваме приходите от абонаменти в системата.**

**Problem 2.   Бизнес логика**

**The Controller Class**

Бизнес логиката на програмата трябва да бъде концентрирана около няколко команди. Внедрете клас, наречен ParkingController, който ще притежава главната функционалност, представена от тези публични методи:

|  |
| --- |
| ParkingController.cs |
| public string CreateParkingSpot(List<string> args)  {  //TODO: Add some logic here …  }    public string ParkVehicle(List<string> args)  {  //TODO: Add some logic here …  }    public string FreeParkingSpot(List<string> args)  {  //TODO: Add some logic here …  }    public string GetParkingSpotById(List<string> args)  {  //TODO: Add some logic here …  }    public string GetParkingIntervalsByParkingSpotIdAndRegistrationPlate(int parkingSpotId)  {  //TODO: Add some logic here …  }  public string CalculateTotal()  {  //TODO: Add some logic here …  } |

**ЗАБЕЛЕЖКА: Не трябва да променяте нищо по методите. Трябва да имплементирате логиката на самите методи. Не прихвайщайте никакви изключения!**

**Команди**

Има няколко команди, които контролират бизнес логиката на приложението и трябва да ги имплементирате.

Те са посочени по-долу.

**CreateParkingSpot Команда**

Създава паркомясто и го регистрира в системата.

Не всички данни ще бъдат валидни!!!

**Параметри**

* id – цяло число
* occupied – булева стойност
* type – текстов низ (валидни стойности: "car", "bus", "subscription")
* price – реално число

Ако за type е посочено “subscription”, ще се въведе допълнителен параметър:

* registrationPlate – текстов низ

Съобщения

* Ако командата премине успешно, трябва да върнете низ в следния формат:
  + Parking spot {spot.Id} was successfully registered in the system!
* Ако командата премине неуспешно, трябва да върнете низ в следния формат:
  + Unable to create parking spot!
* Ако друго паркомясто с посочения идентификатор вече съществува, изведете:
  + Parking spot {id} is already registered!

**ParkVehicle Команда**

Създава обект от ParkingInterval за дадено място, ако то съществува и е свободно, както и е от подходящ вид.

Не всички данни ще бъдат валидни!!!

**Параметри**

* parkingSpotId – цяло число
* registrationPlate – текстов низ
* hoursParked – цяло число
* type – текстов низ

Съобщения

* Ако командата премине успешно, трябва да върнете низ в следния формат:
  + Vehicle {registrationPlate} parked at {parkingSpotId} for {hoursParked} hours.
* Ако превозното средство не може да бъде паркирано на даденото паркомясто, защото то е заето или не е подходящ тип, изведете:
  + Vehicle {registrationPlate} can't park at {parkingSpotId}.
* Ако паркомясто с посочен идентификатор не съществува, изведете:
  + Parking spot {parkingSpotId} not found!

**FreeParkingSpot Команда**

В рамките на тази команда трябва да се освободи дадено паркомясто, ако то е било заето.

Не всички данни ще бъдат валидни!!!

Параметри

* parkingSpotId – цяло число

Съобщения

* Ако командата премине успешно, трябва да върнете низ в следния формат:
  + Parking spot {parkingSpotId} is now free!
* Ако паркомястото не е било заето, трябва да върнете низ в следния формат:
  + Parking spot {parkingSpotId} is not occupied.
* Ако паркомясто с посочен идентификатор не съществува, изведете:
  + Parking spot {parkingSpotId} not found!

**GetParkingSpotById Команда**

При тази команда се връща състоянието на паркомястото обърнато към низ чрез неговия ToString() метод.

Не всички данни ще бъдат валидни!!!

**Параметри**

* parkingSpotId – цяло число

**Съобщения**

* Ако паркомясто с посочен идентификатор не съществува, изведете:
  + Parking spot {parkingSpotId} not found!

**GetParkingIntervalsByParkingSpotIdAndRegistrationPlate Команда**

Тази команда връща информация за всички паркирания на превозно средство с даден регистрационен номер върху дадено паркомясто. За всеки от ParkingInterval обектите, използвайте ToString метода му.

**Параметри**

* parkingSpotId – цяло число
* registrationPlate – текстов низ

Съобщения

* Ако паркомясто с посочен идентификатор не съществува, изведете:
  + Parking spot {parkingSpotId} not found!

**CalculateTotal Команда**

Тази команда изчислява сумата от всички реализирани паркирания (без тези на абонаменти) и извежда съобщение във формат:

* Total revenue from the parking: {sum:F2} BGN

**Problem 3.   Вход / Изход**

**Вход**

* Четете редове с различни команди, докато не получите команда за приключване на програмата.

По-долу можете да видите формата, в който всяка команда ще бъде дадена във входа:

* CreateParkingSpot:{parkingSpotId}:{occupied}:{type}:{price}
* ParkVehicle:{parkingSpotId}:{registrationPlate}:{hoursParked}:{type}
* FreeParkingSpot:{parkingSpotId}
* GetParkingSpotById:{parkingSpotId}
* GetParkingIntervalsByParkingSpotIdAndRegistrationPlate:{parkingSpotId}:{registrationPlate}
* CalculateTotal

**Изход**

За всяка команда трябва да бъде изведен низа, който връща съответния ѝ метод от ParkingController.

Забележка: Всички парични стойности трябва да се закръгли до втория знак след десетичната запетая.

**Ограничения**

* Винаги ще получавате команда за приключване на програмата.
* Входните данни ще бъдат валидни от страна на типове данни.

**Примери**

|  |  |
| --- | --- |
| Input | Output |
| CreateParkingSpot:1:False:car:1.50  ParkVehicle:1:CB1111AA:5:car  FreeParkingSpot:1  ParkVehicle:1:A2222MA:3:car  CreateParkingSpot:2:False:bus:2.50  ParkVehicle:2:B8846KX:3:bus  FreeParkingSpot:1  ParkVehicle:1:CB1111AA:3:car  GetParkingSpotById:1  GetParkingIntervalsByParkingSpotIdAndRegistrationPlate:1:CB1111AA  CalculateTotal  End | Parking spot 1 was successfully registered in the system!  Vehicle CB1111AA parked at 1 for 5 hours.  Parking spot 1 is now free!  Vehicle A2222MA parked at 1 for 3 hours.  Parking spot 2 was successfully registered in the system!  Vehicle B8846KX parked at 2 for 3.  Parking spot 1 is now free!  Vehicle CB1111AA parked at 1 for 3 hours.  > Parking Spot #1  > Occupied: True  > Type: car  > Price per hour: 1.50 BGN  Parking intervals for parking spot 1:  > Parking Spot #1  > RegistrationPlate: CB1111AA  > HoursParked: 5  > Revenue: 7.50 BGN  > Parking Spot #1  > RegistrationPlate: CB1111AA  > HoursParked: 3  > Revenue: 4.50 BGN    Total revenue from the parking: 24.00 BGN |

**Точкуване**

**Всяка различна задача Ви дава точки:**

* **Структура - 30 точки**
  + **За присъждането на точките за структура е нужно да сте реализирали ToString методите и CreateParkingSpot командата в контролера.**
* **Бизнес логика - 50 точки**
* **Вход/Изход - 20 точки**