**Restorant**

**Avaliable from**: Sunday, 2 September 2018, 10:00  
**Due date**: Sunday, 2 September 2018, 16:00  
**Requested files**: Fridge.cs, Product.cs ([Download](https://it-kariera.mon.bg/e-learning/mod/vpl/views/downloadrequiredfiles.php?id=1470))  
**Type of work**: Individual work  
**Настройки на оценките**: Максимална оценка: 100  
**Run**: Да. **Evaluate**: Да  
**Automatic grade**: Да.

**Продукти**

**Общ преглед**

Във вашата фирма постъпва проект за създаване на приложение, обслужващо „ресторант“.

Вашият софтуер трябва да описва **хладилник** **(**Fridge**)** и **продукт (**Product**).**

Tрябва да реализирате функционалност, която да позволява добавяне и премахване на продукти, проверка за налиности и приготвяне на ястите с определени продукти – 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 (вж. структурата му по-долу), и ще трябва да реализирате само необходимите класове **Fridge.cs** и **Product.cs.**

**Основната идея се базира на това, че т.нар. хладилник е структура, която ще съдържа n на брой продукти. Структурата не трябва да пази продуктите в колекция! Всеки продукт пази референция към следващия в поредицата.**

|  |
| --- |
| Program.cs |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace Exam  {  class Program  {  static Fridge fridge = new Fridge();  static void Main(string[] args)  {  string line;  while ("END" != (line = Console.ReadLine()))  {  string[] cmdArgs = line.Split(' ');  switch (cmdArgs[0])  {  case "Add":  AddProduct(cmdArgs[1]);  break;  case "Check":  CheckProductIsInStock(cmdArgs[1]);  break;  case "Remove":  try  {  int index = int.Parse(cmdArgs[1]);  RemoveProductByIndex(index);  }  catch (FormatException e)  {  RemoveProductByName(cmdArgs[1]);  }  break;  case "Print":  ProvideInformationAboutAllProducts();  break;  case "Cook":  CookMeal(cmdArgs.Skip(1).ToArray());  break;  }  }  }  private static void CookMeal(string[] indexes)  {  string[] products = fridge.CookMeal(int.Parse(indexes[0]), int.Parse(indexes[1]));  Console.WriteLine("Meal cooked. Used Products: " + string.Join(", ", products));  }  private static void ProvideInformationAboutAllProducts()  {  string[] info = fridge.ProvideInformationAboutAllProducts();  foreach (var item in info)  {  Console.WriteLine(item);  }  }  private static void RemoveProductByName(string name)  {  string ProductName = fridge.RemoveProductByName(name);  if (ProductName != null)  {  Console.WriteLine("Removed: " + ProductName);  }  else  {  Console.WriteLine("Product not found!");  }  }  private static void RemoveProductByIndex(int index)  {  string ProductName = fridge.RemoveProductByIndex(index);  if (ProductName != null)  {  Console.WriteLine("Removed: " + ProductName);  }  else  {  Console.WriteLine("Product not found!");  }  }  private static void CheckProductIsInStock(string name)  {  bool isInStock = fridge.CheckProductIsInStock(name);  Console.WriteLine(isInStock ? $"Product {name} is in stock."  : "Not in stock");  }  private static void AddProduct(string name)  {  fridge.AddProduct(name);  }  }  } |

**Подзадача 1: 30 точки**

**Product**

Всички продукти имат име и референция към следващ продукт:

name – низ, съставен от малки и/или големи латински букви

next – референция към следващ продукт

|  |
| --- |
| Product.cs |
| private string name;  private Product next;    public Product(string name)  {      //TODO: Добавете вашия код тук …  }    public string Name  {      //TODO: Добавете вашия код тук …  }    public Product Next  {      //TODO: Добавете вашия код тук …  }    public override string ToString()  {      //TODO: Добавете вашия код тук …  } |

**Fridge**

Всички хладилници Product head, Product tail, Count:

head – Product, първи в поредицата

tail – Product, последен в поредицата

count – Брой продукти

|  |
| --- |
| Fridge.cs |
| private Product head;  private Product tail;  private int count;    public Fridge(){ }    public int Count  {      //TODO: Добавете вашия код тук …  }    public void AddProduct(string ProductName)  {      //TODO: Добавете вашия код тук …  }    public string[] CookMeal(int start, int end)  {      //TODO: Добавете вашия код тук …  }    public string RemoveProductByIndex(int index)  {      //TODO: Добавете вашия код тук …  }    public string RemoveProductByName(string name)  {      //TODO: Добавете вашия код тук …  }    public bool CheckProductIsInStock(string name)  {      //TODO: Добавете вашия код тук …  }    public string[] ProvideInformationAboutAllProducts()  {      //TODO: Добавете вашия код тук …  } |

**Командa за добавяне на продукт**

Вашето приложение трябва да обслужва следната команда за добавяне на продукти:

* **Add** <**име**> - тази команда има за цел да добави продукт с неговото име.

**Команда за извеждане на информация**

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

* **Print** - отпечатва информация за всички продукти в структурата във формат:  
  **Product {name}**
* За успешна реализация трябва да реализирате ваша версия на **ToString()**метода за класа **Product.**
* **RemoveProductByIndex** **<index**> - Трябва да бъде премахнат елемент, който се намира на посочения индекс. Тъй като вашата структура не използва индексиране, удобен похват би бил използването на брояч. При успешно намиране и премахване на Product трябва да върнете неговото име, което ще бъде изпечатано на конзолата от Main метод-а. При ненамиране на такъв Product, трябва да бъде върната null стойност.
* **RemoveProductByName** **<name**> - Трябва да бъде премахнат първият елемент, на който името отговаря на подаденото. При успешно намиране и премахване на Product трябва да върнете неговото име, което ще бъде изпечатано на конзолата от Main метод-а. При ненамиране на такъв Product, трябва да бъде върната null стойност.
* **CheckProductIsInStock<name**> - Трябва да бъде намерен елемент, на който името отговаря на подаденото. При успешно намиране Product трябва да върнете **true** в обратен случай **false**
* **CookMeal<int startIndex, int endIndex**> - Трябва да бъдат намерени всички продукти от **startIndex** до **endIndex** .  Имената на всички намерени продукти трябва да бъдат събрани в стрингов масив, който да бъде върнат от метода.

**Подзадача 2: 30 точки**

**Подзадача 3: 20 точки**

**Подзадача 4: 20 точки**

**Kоманди**

Вашето приложение трябва да реализира следните команди:

* **Add <name> -**Добавя продукт към структурата
* **Print** – изпечатва се информация за всички налични продукти
* **Remove** **<int index> -**Трябва да бъде премахнат елемент, който се намира на посочения индекс. Тъй като вашата структура не използва индексиране, удобен похват би бил използването на брояч. При успешно намиране и премахване на Product трябва да върнете неговото име, което ще бъде изпечатано на конзолата от Main метод-а. При ненамиране на такъв Product, трябва да бъде върната null стойност.
* **Remove** **<string name**> - Трябва да бъде премахнат първият елемент, на който името отговаря на подаденото. При успешно намиране и премахване на Product трябва да върнете неговото име, което ще бъде изпечатано на конзолата от Main метод-а. При ненамиране на такъв Product, трябва да бъде върната null стойност.
* **Check <name>**- При намерен продукт – **Product <name> is in stock** в обратен случай – **Not in stock**
* **Cook <int startIndex, int endIndex**> -  “Приготва се ястие“, в контекста на програмата, това означава да извадите имената на всички продукти, който се намират от **startIndex ДО endIndex.**

**В случай, че endIndex e след последния елемент, вземете колкото продукти имате от startIndex**

**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 други е необходимо форматиране на изход – напр. Product.ToString(), Product.Name()

**Ограничения**

* Всички цели числа ще бъдат в диапазона **–10000** до **+10000**
* Имената няма да съдържат интервал

**Примери**

|  |  |
| --- | --- |
| **Вход** | **Изход** |
| Add cherry  Add salami  Print  END | Product cherry  Product salami |
| Add cherry  Add salami  Add eggs  Remove 1  Remove eggs  Print  Check dadadada  Check cherry  Check eggs  Add eggs  Cook 0 2  Cook 0 25  Remove 0  Print  END | Removed: salami  Removed: eggs  Product cherry  Not in stock  Product cherry is in stock.  Not in stock  Meal cooked. Used Products: cherry, eggs  Meal cooked. Used Products: cherry, eggs  Removed: cherry  Product eggs |