# Бон Апетит

### Общ преглед

Вие сте собственик на ресторант “**STAKE** Overflow”. Освен това сте и добър програмист. С цел да се преборите с конкуренцията и капризните си клиенти вие трябва да създадете важен софтуер за вашето заведение!

Вашият софтуер трябва да описва **продукти** (Product) и **ястия (Meal).**

Tрябва да реализирате функционалност, която да позволява добавяне на продукти и ястия, а в последствие и тяхното поръчване – 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 методи в Meal.cs и Product.cs

## Подзадача 1: Структура на Продукт

### Product

Всички продукти имат име, цена и грамаж:

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

Price – число с плаваща запетая

Weight – цяло число в интервала [1, 10000]

Реализирайте и конструктор:

* **Product(string name, double price, int weight)** – този конструктор трябва да приема името на продукта, цената му и грамажа. За справка вижте следната схема за Product.cs:

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

### Команди за добавяне на продукти

Трябва да реализирате следните две команди за добавяне на продукти:

* **AddProduct** <**име**> <**цена**> <**грамаж**> - тази команда има за цел да добави продукт с неговото име, цена и грамаж.
* **AddMultiProducts <брой продукти> -** тази команда има за цел да добави няколко продукта наведнъж. Тя получава брой продукти, след което на съответния брой редове се подава информация за всеки продукт. Информацията се подава по същия формат като **AddProduct**

### Команда за извеждане на информация

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

* **PrintProduct** <**име**> - отпечатва информация за продукт във формат:  
  <**име**> - <**грамаж**>

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

## Подзадача 2: Структура на Ястие

### Meal

Всяко ястие има: име, вид, списък с участващи продукти, брой пъти, в които е поръчвано.

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

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

Products – списък от **Product**

Ordered – цяло число

Всяко ястие си има и цена, която се изчислява по следния алгоритъм:

1. Сумират се всички цени на продуктите, участващи в състава на ястието
2. Крайната цена е сумата от цените на продуктите и 30% надценка.

Вие трябва да реализирате и два конструктора:

* **Meal(string name, string type)** – този конструктор трябва да приема името на ястието и типа му.
* **Meal(string name, string type, List<Product> products)** – този конструктор трябва да приема името на ястието, типа му и списък с участващите в него продукти.

За справка вижте следната схема за Meal.cs:

|  |
| --- |
| Meal.cs |
| public Meal(string name, string type)  {  //TODO: Добавете вашия код тук …  }  public Meal(string name, string type, List<Product> products)  {  //TODO: Добавете вашия код тук …  } |

### Команди за добавяне на ястие

* **AddМeal** <**име**> <**тип**> - тази команда има за цел да добави ястие с неговото име и тип.
* **AddMealProducts** <**име**> <**тип**> <**брой** **продукти**>  
  <**продукт 1**> <**продукт 2**>... **-** Добавя информация за ястие - име, вид, брой продукти, след което имената продуктите са изредени на следващия ред, разделени с интервали

### Команда за извеждане на информация

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

* **PrintMeal <име> -** отпечатва информация за продукт във формат:  
  <**име**> - <**вид**>

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

## Подзадача 3: Поръчване

### Kоманди

Вашето приложение трябва да реализира следните команди:

* **AddProductToMeal** <**име на продукт**> <**име на ястие**> - Тази команда добавя даден продукт към списъка с ястия
* **ContainsProduct** <**име на продукт**> <**име на ястие**> - Проверява дали в състава на дадено ястие се съдържа даден продукт. Тази команда трябва да извежда съобщение “**<продукт> is contained in <ястие>.**”, ако продуктът се съдържа в списъка със съставки на ястието и “**<продукт> is NOT contained in <ястие>.**”, в противен случай.
* **GetMealPrice** <**ястие**> - Дава информация каква е цената на ястие. Цената се изчислява като сбор от цените на всички продукти + 30% надценка. Тази команда трябва да извежда съобщение: “**The price of <продукт> is: <цена>.**”, където цената е форматирана до втория знак след запетаята
* **PrintMealRecipe** <**ястие**> - Изпечатва продуктите участващи в състава на ястието по реда, **в който са били добавени към ястието**. Изпечатването се случва по следния формат:
  + Трябва да изведете един ред с 25 символа “**-**”.
  + На следващия ред трябва да изведете “**<ястие> RECIPE**”.
  + След това да изведете още един ред с 25 символа “**-**”.
  + За всеки продукт изведете по един ред във формата:  
    **<име> - <грамаж>**
  + Накрая изведете още един ред с 25 символа “**-**”.
* **OrderMeal** <**ястие**> - Поръчване на ястие. Броячът за това колко пъти е било поръчано едно ястие се увеличава с 1.
* **GetSpecialty –** тази команда не приема други параметри. Нейното предназначение е да отпечатва информация за най-поръчваното ястие. Командата трябва да отпечатва следния текст: **“The current specialty is: <ястие>”**

## Подзадача 4: Още няколко команди ☺

Трябва да реализирате още няколко команди:

* **RecommendByPrice** <**цена**> - Дава препоръка за ястие, което се доближава максимално до цената. За да получите това - изберете ястията с цена по-малка или равна на параметъра, след което ги сортирайте в намалящ ред по цените им и вземете първото. Съобщението, което трябва да изведе тази команда е: **“The recommended meal for <желана цена> is <ястие>. It costs <цена на ястието>”**
* **RecommendByPriceAndType <цена> <вид> -** Аналогична на предната команда, само че предложенията се ограничават и по тип. Съобщението за нея е „The recommended meal for <желана цена> of type <вид> is <ястие>. It costs <цена на ястието>”
* **Cheapest –** тази команда няма параметър. Тя трябва да изведе информация за най-евтиния продукт, като съобщението трябва да е следното: "The cheapest product is <име>."

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

|  |
| --- |
| Meal.cs |
| public Meal(string name, string type)  {  //TODO: Добавете вашия код тук …  }  public Meal(string name, string type, List<Product> products)  {  //TODO: Добавете вашия код тук …  }  public void AddProduct(Product p) {  //TODO: Добавете вашия код тук …  }  public bool ContainsProduct(string name) {  //TODO: Добавете вашия код тук …  }  public void PrintRecipe() {  //TODO: Добавете вашия код тук …  }  public void Order() {  //TODO: Добавете вашия код тук …  }  public static Meal GetSpecialty(Dictionary<string, Meal> meals) {  //TODO: Добавете вашия код тук …  }  public static Meal RecommendByPrice(double price, Product<string, Meal> meals) {  //TODO: Добавете вашия код тук …  }  public static Meal RecommendByPriceAndType(double price, string type, Product<string, Meal> meals) {  //TODO: Добавете вашия код тук …  } |

|  |
| --- |
| Product.cs |
| public Product (string name, double price, int weight)  {  //TODO: Добавете вашия код тук …  }    public static Product GetCheapestProduct(Dictionary <string, Product> products) {  //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

Освен всичко останало вие трябва да направите и валидация!

Не допускайте създаването на:

* Продукт с цена под 0.01
* Продукт с име по-късо от 3 символа
* Продукт с нулев или отрицателен грамаж
* Ястие с име по-късо от 3 символа
* Ястие с тип празен низ

### Допълнително информация свързана с валидацията

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

На **AddProductToMeal, ContainsProduct, GetMealPrice, PrintProduct, PrintMeal, PrintMealRecipe, OrderMeal ВИНАГИ** ще подава продукт/ястие, което съществува.

**GetSpecialty** винаги ще бъде извиквана, когато има продукти в списъка и някой от тях е бил поръчан до момента на нейното извикване, като в тестовите примери **НЯМА** да има случаи на два продукта, които да са специалитети.

Командите за препоръка **RecommendByPrice** и **RecommendByPriceAndType** винаги ще бъдат извиквани така че да има продукт, който да отговаря на съответните параметри.

**Ако получите команда, която се опитва да извърши операция, която нарушава валидацията, хвърлете изключение с текст “Invalid Command!”.**

## 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 останалите формата е:

#### ContainsProduct

Ако продуктът се съдържа в списъка с ястия:

{продукт} is contained in {ястие}.

Ако продуктът **НЕ** се съдържа в списъка:

{продукт} is NOT contained in {ястие}.

#### GetMealPrice

The price of {продукт} is: {цена}

#### PrintMealRecipe

* Трябва да изведете един ред с 25 символа “-”.
* На следващия ред трябва да изведете “<ястие> RECIPE”.
* След това да изведете още един ред с 25 символа “-”.
* За всеки продукт изведете по един ред във формата:  
  <име> - <грамаж>
* Накрая изведете още редин ред с 25 символа “-”.

#### GetSpecialty

The current specialty is: {ястие}

#### RecommendByPrice

The recommended meal for {желана цена} is {ястие}. It costs {цена на ястието}

#### RecommendByPriceAndType

The recommended meal for {желана цена} of type {вид} is {ястие}. It costs {цена на ястието}

#### Cheapest

The cheapest product is {име}.

### Ограничения

* Всички цели числа ще са такива, че да се побират в int.
* Всички числа с плаваща запетая ще бъдат въвеждани с до два знака след запетаята.
* Всички имена няма да съдържат интервал

### Примери

|  |  |
| --- | --- |
| **Вход** | **Изход** |
| AddProduct testo 1.50 100  PrintProduct testo  AddProduct meso 2.00 200  PrintProduct meso  AddMeal pizza pizza  PrintMeal pizza  End | testo - 100  meso - 200  pizza - pizza |
| AddProduct testo 1.50 100  PrintProduct testo  AddProduct meso 2.00 200  PrintProduct meso  AddMeal pizza pizza  PrintMeal pizza  AddMultiProducts 3  domatenSos 0.50 1  domatki 0.60 5  kashkaval 0.80 5  PrintProduct domatki  AddProductToMeal testo pizza  AddProductToMeal domatenSos pizza  AddProductToMeal domatki pizza  AddProductToMeal meso pizza  AddProductToMeal kashkaval pizza  ContainsProduct pesto pizza  GetMealPrice pizza  PrintMealRecipe pizza  Cheapest  End | testo – 100  meso - 200  pizza - pizza  domatki - 5  pesto is NOT contained in pizza.  The price of pizza is: 7.02  -------------------------  pizza RECIPE  -------------------------  testo - 100  domatenSos - 1  domatki - 5  meso - 200  kashkaval - 5  -------------------------  The cheapest product is domatenSos. |
| AddProduct testo 1.50 100  PrintProduct testo  AddProduct meso 2.00 200  PrintProduct meso  AddMeal pizza pizza  PrintMeal pizza  AddMultiProducts 6  domatenSos 0.50 1  domatki 0.60 5  kashkaval 0.80 5  domati 0.60 5  krastavici 0.30 3  zele 0.70 1  PrintProduct domatki  AddProductToMeal testo pizza  AddProductToMeal domatenSos pizza  AddProductToMeal domatki pizza  AddProductToMeal meso pizza  AddProductToMeal kashkaval pizza  ContainsProduct pesto pizza  GetMealPrice pizza  PrintMealRecipe pizza  Cheapest  OrderMeal pizza  AddMeal pArl3nka pizza  AddProductToMeal testo pArl3nka  AddProductToMeal kashkaval pArl3nka  PrintMeal pArl3nka  AddMealProducts salata salati 3  domati krastavici zele  PrintMeal salata  ContainsProduct testo salata  ContainsProduct testo pizza  GetMealPrice salata  OrderMeal pizza  OrderMeal salata  GetSpecialty  RecommendByPrice 5  RecommendByPriceAndType 8 pizza  End | testo – 100  meso - 200  pizza - pizza  domatki - 5  pesto is NOT contained in pizza.  The price of pizza is: 7.02  -------------------------  pizza RECIPE  -------------------------  testo - 100  domatenSos - 1  domatki - 5  meso - 200  kashkaval - 5  -------------------------  The cheapest product is krastavici.  pArl3nka - pizza  salata - salati  testo is NOT contained in salata.  testo is contained in pizza.  The price of salata is: 2.08  The current specialty is: pizza  The recommended meal for 5 is pArl3nka. It costs 2.99  The recommended meal for 8 of type pizza is pizza. It costs 7.02 |

## 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**