Множества и речници

Множества, речници, мулти- и сложни речници

SoftUni Foundation

Проект "Отворено учебно съдържание по програмиране и ИТ", СофтУни Фондация
https://github.com/BG-IT-Edu

Курс "Структури от данни и алгоритми"

Софтуерни и хардуерни науки

Съдържание

1. **Речници**
2. **Мулти-речници**
   - Сложни речници
3. **Множества**
   - **`HashSet<T>`** и **`SortedSet<T>`**
   - **`List<T>`** срещу **`HashSet<T>`**

Dictionary<K, V>

Речници

Асоциативни масиви

- **Асоциативни масиви** == масиви, индексирани чрез **ключове**
  - Не по номера 0, 1, 2, … (както при масивите)
- Съдържат колекция от двойки {**ключ** → **стойност**}

Key	Value
John Smith	+1-555-8976
Lisa Smith	+1-555-1234
Sam Doe	+1-555-5030

Речници

- `Dictionary<K, V>` – представлява колекция от двойки
- Ключовете са **уникални**
- Поддържа ключовете в техния **ред на добавяне**

```
Dictionary<string, double> fruits = new Dictionary<string,
  double>();
fruits["banana"] = 2.20;
fruits["apple"] = 1.40;
fruits["kiwi"] = 3.20;
```

Сортирани речници

- `SortedDictionary<K, V>`
- Поддържа ключовете **сортирани**

```
SortedDictionary<string, double> fruits =
  new SortedDictionary <string, double>();
fruits["kiwi"] = 4.50;
fruits["orange"] = 2.50;
fruits["banana"] = 2.20;
```

Вградени методи (1)

- **Add(ключ, стойност)**

```
var airplanes = new Dictionary<string, int>();
airplanes.Add("Boeing 737", 130);
airplanes.Add("Airbus A320", 150);
```

- **Remove(ключ)**

```
var airplanes = new Dictionary<string, int>();
airplanes.Add("Boeing 737", 130);
airplanes.Remove("Boeing 737");
```

Вградени методи (2)

- **ContainsKey(ключ)** – много бърза операция

```
var dictionary = new Dictionary<string, int>();
dictionary.Add("Airbus A320", 150);
if (dictionary.ContainsKey("Airbus A320"))
   Console.WriteLine($"Airbus A320 key exists");
```

- **ContainsValue(стойност)** – много бавна операция

```
var dictionary = new Dictionary<string, int>();
dictionary.Add("Airbus A320", 150);
Console.WriteLine(dictionary.ContainsValue(150)); // true
Console.WriteLine(dictionary.ContainsValue(100)); // false
```

Обхождане на речник

- Можем да използваме **foreach-цикъл**
- Минаваме през обекти от тип **KeyValuePair<K, V>**
- Речникът **не може** да се модифицира (**read-only**)

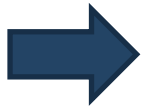
```
var fruits = new Dictionary<string, double>();
fruits.Add("banana", 2.20);
fruits.Add("kiwi", 4.50);
fruits.Add("orange", 3.20);
foreach (KeyValuePair<string, double> fruit in fruits)
  Console.WriteLine($"{fruit.Key} -> {fruit.Value}");
```

fruit.Key -> името на плода
fruit.Value -> цената на плода

Задача: Брой еднакви стойности в масив

- Прочетете **масив** от реални числа и отпечатайте **колко пъти се среща всяко от тях**

```
8 2.5 2.5 8 2.5
```



```
8 - 2 times
2.5 - 3 times
```

```
1.5 5 1.5 3
```

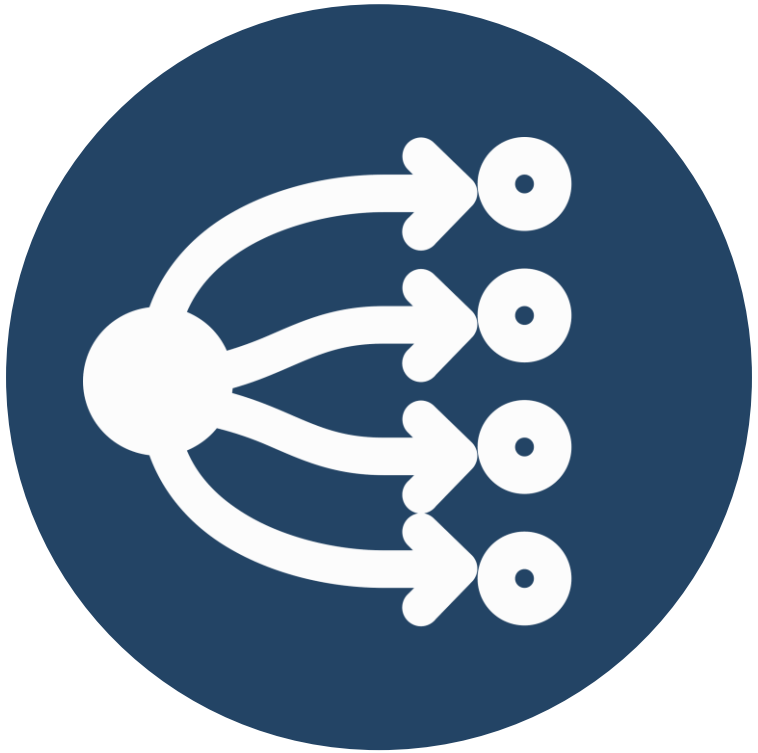
```
1.5 - 2 times
5 - 1 times
3 - 1 times
```

Решение: Брой еднакви стойности в масив

```
double[] nums = Console.ReadLine().Split(' ')
  .Select(double.Parse).ToArray();

Dictionary<double, int> counts = new Dictionary<double, int>();

foreach (int num in nums)
   if (counts.ContainsKey(num))
      counts[num]++;
   else
      counts[num] = 1;

foreach (KeyValuePair<double, int> num in counts)
    Console.WriteLine($"{num.Key} - {num.Value} times");
```

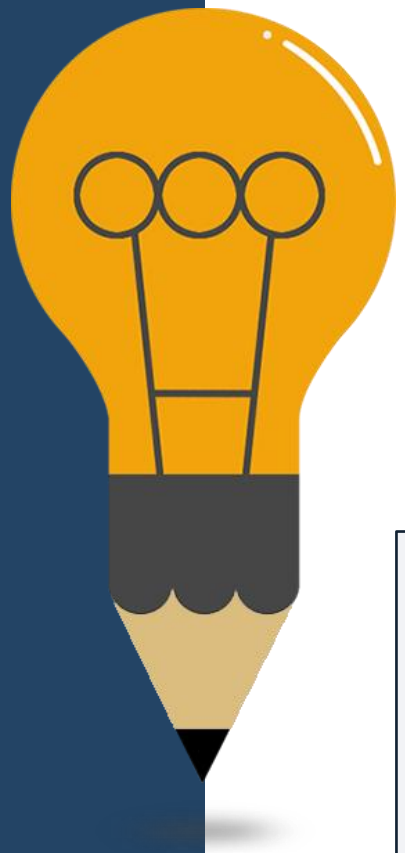
counts[num] винаги ще показва колко пъти се съдържа числото

Тествайте решението си в Judge: https://judge.softuni.org/Contests/Practice/Index/4160#0



Мулти-речници

Мулти-речници



- Един речник може да има **множество от стойности** за даден ключ
  - Пример: студентите могат да имат много оценки:
    - Петър → [5, 5, 6]
    - Кирил → [6, 6, 3, 4, 6]

```
var grades = new Dictionary<string, List<int>>();
grades["Peter"] = new List<int>();
grades["Peter"].Add(5);
grades["Kiril"] = new List<int>() { 6, 6, 3, 4, 6 };
Console.WriteLine(string.Join(" ", grades["Kiril"]);
```

Задача: Средноаритметичен успех

- Напишете програма, която прочита **имената** на учениците и **оценките**
- Отпечатайте **оцениките** и **средноаретметичния успех** за всеки ученик

```
6
Ivancho 5.20
Mariika 5.50
Mariika 2.50
Stamat 2.00
Mariika 3.46
Stamat 3.00
```

```
Ivancho -> 5.20 (avg: 5.20)
Mariika -> 5.50 2.50 3.46 (avg: 3.82)
Stamat -> 2.00 3.00 (avg: 2.50)
```

Решение: Средноаритметичен успех (1)

```
var grades = new Dictionary<string, List<double>>();

var n = int.Parse(Console.ReadLine());

for (int i = 0; i < n; i++) {
  var tokens = Console.ReadLine().Split();
  var name = tokens[0];
  var grade = double.Parse(tokens[1]);
  if (!grades.ContainsKey(name))
    grades[name] = new List<double>();
  grades[name].Add(grade);
}

// Продължаваме на следващия слайд ...
```

Уверете се, че списъците са създадени

Добавете оценките в списъка

Решение: Средноаритметичен успех (2)

```
foreach (var pair in grades)
{
  var name = pair.Key;

  var studentGrades = pair.Value;

  var average = studentGrades.Average();

  Console.Write($"{name} -> ");

  foreach (var grade in studentGrades)
    Console.Write($"{grade:f2} ");

  Console.WriteLine($"(avg: {average:f2})");
}
```

KeyValuePair<string, List<double>

Тествайте решението си в Judge: https://judge.softuni.org/Contests/Practice/Index/4160#1

Сложни речници

- Речници, съдържащи **речници** като **стойност**
- Пример: населението по държави и градове

BG →
Sofia → 1,211,000
Plovdiv → 338,657

UK →
London → 8,674,000
Manchester → 2,550,000

USA →
New York City, NY → 8,406,000
Washington, DC → 658,893

Задача: Хранителен магазин

- Напишете програма, която събира информация за **хранителни магазини**
- Ако получите магазин, който съществува, **добавете продукта**
- Сортирайте речника по **име на магазина**

{shop}, {product}, {price}

```
lidl, juice, 2.30
kaufland, banana, 1.10
lidl, grape, 2.20
Revision
```

Край на програмата

```
kaufland->
Product: banana, Price: 1.1
lidl->
Product: juice, Price: 2.3
Product: grape, Price: 2.2
```

Решение: Хранителен магазин (1)

SoftUni Foundation

```
var shops = new Dictionary<string, Dictionary<string, double>>();

string line;

while ((line = Console.ReadLine()) != "Revision")
{
  string[] productsInfo = line.Split(", ");
  string shop = productsInfo[0];
  string product = productsInfo[1];
  double price = double.Parse(productsInfo[2]);
// Продължаваме на следващия слайд...
```

Решение: Хранителен магазин (2)

```
  if (!shops.ContainsKey(shop))
  {
    shops.Add(shop, new Dictionary<string, double>());
  }
  shops[shop].Add(product, price);
}


var orderedShops = shops.OrderBy(s => s.Key)
  .ToDictionary(x => x.Key, x => x.Value);

// TODO: Отпечатайте сортирания речник
```

Уверете се, че речниците са създадени

Тествайте решението си в Judge: https://judge.softuni.org/Contests/Practice/Index/4160#2

Задача: Градове по континент и държава

- Напишете програма, която чете и съхранява информация за **континенти**, **държави** и **градове**
- Отпечатайте ги в следния формат:

```
6
Europe Bulgaria Sofia
Asia China Beijing
Asia Japan Tokyo
Europe Poland Warsaw
Europe Germany Berlin
Europe Poland Poznan
```

```
Europe:
  Bulgaria -> Sofia
  Poland -> Warsaw, Poznan
  Germany -> Berlin
Asia:
  China -> Beijing
  Japan -> Tokyo
```

Решение: Градове по континент и държава (1)

```
var continentsData =
    new Dictionary<string, Dictionary<string, List<string>>>();

var n = int.Parse(Console.ReadLine());

for (int i = 0; i < n; i++) {
  var tokens = Console.ReadLine().Split();
  var continent = tokens[0];
  var country = tokens[1];
  var city = tokens[2];
  // Продължаваме на следващия слайд...
```

Решение: Градове по континент и държава (2)

```
  if (!continentsData.ContainsKey(continent)) {
    continentsData[continent] =
        new Dictionary<string, List<string>>();
  }

  if (!continentsData[continent].ContainsKey(country)) {
    continentsData[continent][country] = new List<string>();
  }

  continentsData[continent][country].Add(city);
}
// Продължаваме на следващия слайд...
```

Инициализираме континентите

Инициализираме градовете

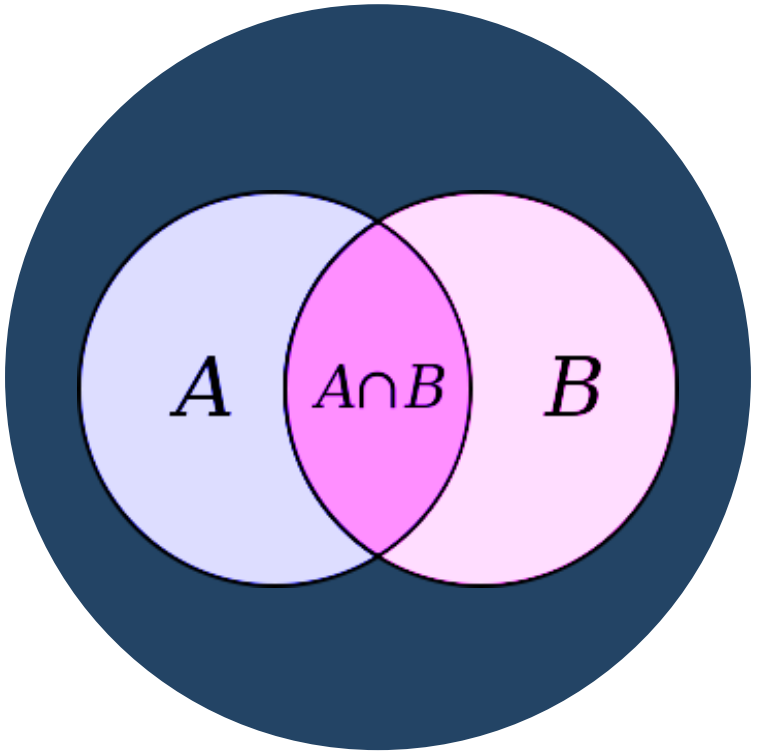
Добавяме града към държавата

Решение: Градове по континент и държава (3)

```
foreach (var continentCountries in continentsData) {
  var continentName = continentCountries.Key;
  Console.WriteLine($"{continentName}:");
  foreach (var countryCities in continentCountries.Value) {
    var countryName = countryCities.Key;
    var cities = countryCities.Value;
    // TODO: Отпечатайте държавата с нейните градове
  }
}
```

Градове в държавата

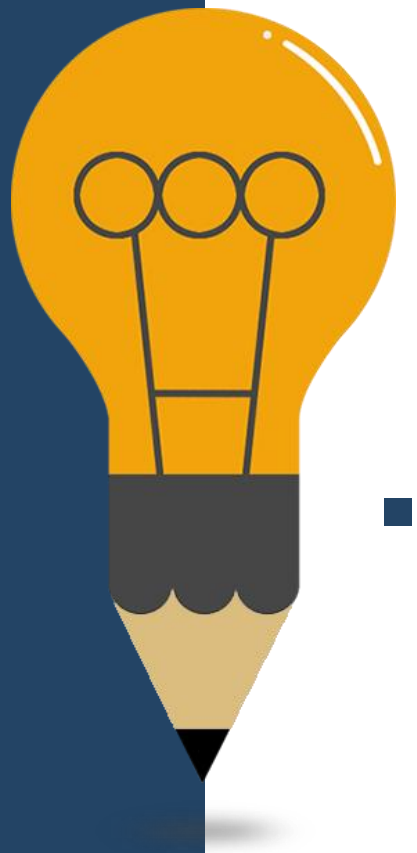
Тествайте решението си в Judge: https://judge.softuni.org/Contests/Practice/Index/4160#3



Множества

HashSet<T> и SortedSet<T>

Множество



- **Множество** (`Set`) == съвкупност от **уникални елементи**
  - С него можем да **добавяме**, **премахваме** и **търсим** елементи
  - Много бързо изпълнение
- `HashSet<T>`
  - Колекция от елементи в **hash-таблица**
  - Елементите **не са в определен ред**
  - Подобно на `List<T>`, с различна имплементация

HashSet<T> – Примери

SoftUni Foundation

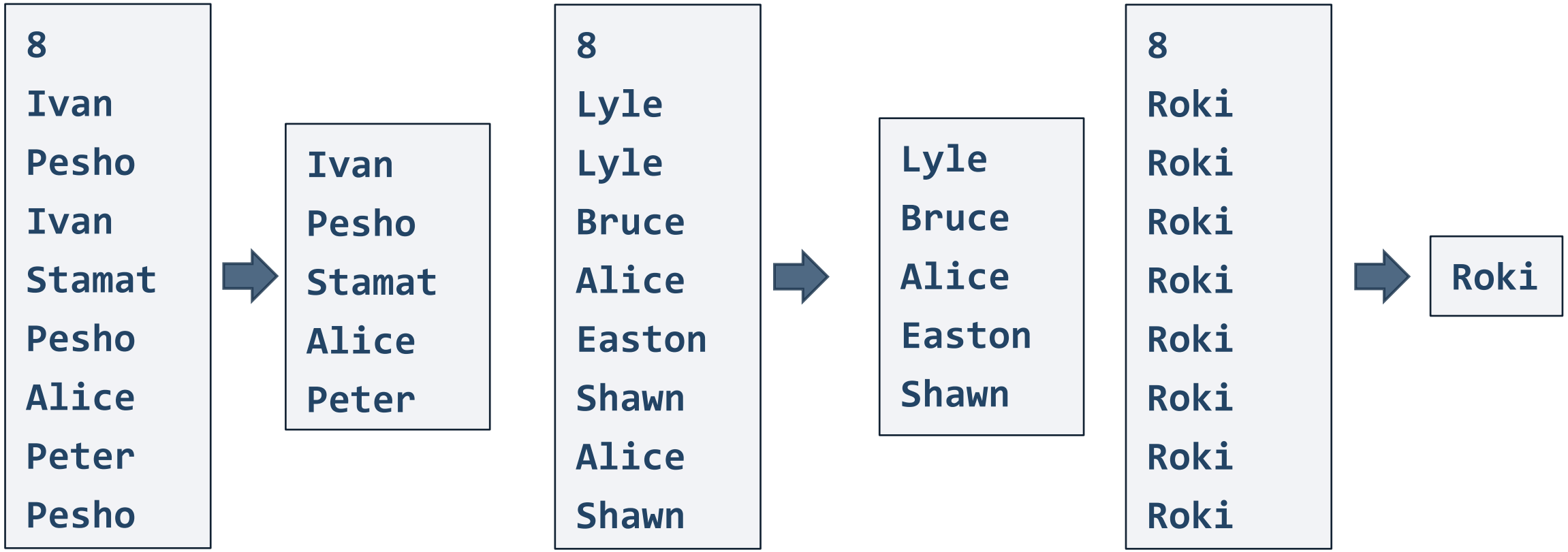
```
HashSet<string> set = new HashSet<string>();
set.Add("Pesho");
set.Add("Pesho"); // Не го добавя отново
set.Add("Gosho");
Console.WriteLine(string.Join(", ", set)); // Pesho, Gosho
Console.WriteLine(set.Contains("Georgi")); // false
Console.WriteLine(set.Contains("Pesho")); // true
set.Remove("Pesho");
Console.WriteLine(set.Count); // 1
```

List<T> срещу HashSet<T>

- `List<T>`
  - Бързо добавя,
    бавно търси и премахва
  - Може да има **повторения**
  - **Редът** на вмъкване е
    гарантиран
- `HashSet<T>`
  - Бързо добавя, търси и
    премахва (чрез **hash-таблица**)
  - **Не** позволява **повторения**
  - **Редът** на вмъкване не
    е **гарантиран**

Задача: Уникални имена

- Прочетете **поредица от имена** и отпечатайте всички **уникални имена**



Решение: Уникални имена

```
var names = new HashSet<string>();

var n = int.Parse(Console.ReadLine());

for (int i = 0; i < n; i++)
{
  var name = Console.ReadLine();
  names.Add(name);
}

foreach (var name in names)
  Console.WriteLine(name);
```

HashSet събира само уникалните имена

Добавяме името

Тествайте решението си в Judge: https://judge.softuni.org/Contests/Practice/Index/4160#4

SortedSet<T>

- Класът **SortedSet<T>** подрежда елементите в **сортиран ред** (азбучен или по големина)

```
var set = new SortedSet<string>();
set.Add("Pesho");
set.Add("Pesho");
set.Add("Gosho");
set.Add("Maria");
set.Add("Alice");
Console.WriteLine(string.Join(", ", set));
```

Alice, Gosho, Maria, Pesho

Какво научихме днес?

SoftUni Foundation

- **Речник** == колекция от двойки {ключ → стойност}
- **Мулти-речниците** съдържат **колекции** като **стойности**
- **Сложните речници** съдържат **речници** като **стойности**
- Множеството съдържа **уникални** стойности **без конкретно подреждане**
  - Без повторения
  - Бързо добавяне, търсене и премахване