Curso C# Completo

Capítulo: Generics, Set, Dictionary

http://educandoweb.com.br

Prof. Dr. Nelio Alves

Introdução aos Generics

http://educandoweb.com.br

Prof. Dr. Nelio Alves

Generics

- Generics permitem que **classes, interfaces** e **métodos** possam ser parametrizados por tipo. Seus benefícios são:
  - Reuso
  - Type safety
  - Performance

- Uso comum: coleções

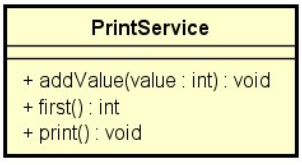
```
List<string> list = new List<string>();
list.Add("Maria");
string name = list[0];
```

Problema motivador 1 (reuso)

Deseja-se fazer um programa que leia um conjunto de N números inteiros (N de 1 a 10), e depois imprima esses números de forma organizada conforme exemplo. Em seguida, informar qual foi o primeiro valor informado.

```
How many values? 3
10
8
23
[10, 8, 23]
First: 10
```

Criar um serviço de impressão:

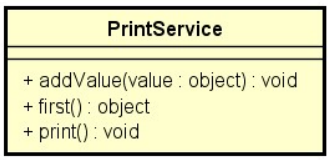


Problema motivador 2 (type safety & performance)

Deseja-se fazer um programa que leia um conjunto de N números inteiros (N de 1 a 10), e depois imprima esses números de forma organizada conforme exemplo. Em seguida, informar qual foi o primeiro valor informado.

```
How many values? 3
10
8
23
[10, 8, 23]
First: 10
```

Criar um serviço de impressão:

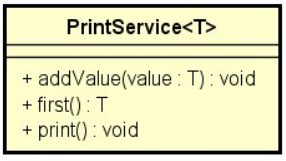


Solução com generics

Deseja-se fazer um programa que leia um conjunto de N números inteiros (N de 1 a 10), e depois imprima esses números de forma organizada conforme exemplo. Em seguida, informar qual foi o primeiro valor informado.

```
How many values? 3
10
8
23
[10, 8, 23]
First: 10
```

Criar um serviço de impressão:



https://github.com/acenelio/generics1-csharp

Restrições para generics

http://educandoweb.com.br

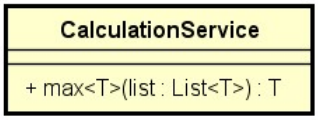
Prof. Dr. Nelio Alves

Problema

Uma empresa de consultoria deseja avaliar a performance de produtos, funcionários, dentre outras coisas. Um dos cálculos que ela precisa é encontrar o maior dentre um conjunto de elementos. Fazer um programa que leia um conjunto de N produtos, conforme exemplo, e depois mostre o mais caro deles.

```
Enter N: 3
Computer,890.50
IPhone X,910.00
Tablet,550.00
Max:
IPhone, 910.00
```

Criar um serviço de cálculo:



https://github.com/acenelio/generics2-csharp

Restrições possíveis

- https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/generics/constraints-on-type-parameters

- where T: struct
- where T : class
- where T : unmanaged
- where T : new()
- where T : <base type name>
- where T : U

GetHashCode e Equals

http://educandoweb.com.br

Prof. Dr. Nelio Alves

GetHashCode e Equals

- São operações da classe Object utilizadas para comparar se um objeto é igual a outro

- Equals: lento, resposta 100%
- GetHashCode: rápido, porém resposta positiva não é 100%

- Os tipos pré-definidos já possuem implementação para essas operações. Classes e structs personalizados precisam sobrepô-las.

Equals

Método que compara se o objeto é igual a outro, retornando true ou false.

```
string a = "Maria";
string b = "Alex";

Console.WriteLine(a.Equals(b));
```

GetHashCode

Método que retorna um número inteiro representando um código gerado a partir das informações do objeto

```
string a = "Maria";
string b = "Alex";

Console.WriteLine(a.GetHashCode());
Console.WriteLine(b.GetHashCode());
```

Regra de ouro do GetHashCode

- Se o código de dois objetos for diferente, então os dois objetos são diferentes

- Se o código de dois objetos for igual, **muito provavelmente** os objetos são iguais (pode haver colisão)

GetHashCode e Equals personalizados

```
class Client {

    public string Name { get; set; }
    public string Email { get; set; }
}
```

HashSet<T> e SortedSet<T>

http://educandoweb.com.br

Prof. Dr. Nelio Alves

HashSet<T> e SortedSet<T>

- Representa um conjunto de elementos (similar ao da Álgebra)
  - Não admite repetições
  - Elementos não possuem posição
  - Acesso, inserção e remoção de elementos são rápidos
  - Oferece operações eficientes de conjunto: interseção, união, diferença.
- HashSet
  - https://msdn.microsoft.com/en-us/library/bb359438(v=vs.110).aspx
- SortedSet
  - https://msdn.microsoft.com/en-us/library/dd412070(v=vs.110).aspx

Diferenças

- HashSet Mais rápido
  - Armazenamento em tabela hash Cada elemento do conjunto tem um código específico, gerado internamente, associado a ele.
  - Extremamente rápido: inserção, remoção e busca O(1) Apenas um passo, executação do mesmo processamento sempre
  - A ordem dos elementos não é garantida
- SortedSet
  - Armazenamento em árvore
  - Rápido: inserção, remoção e busca O(log(n))
  - Os elementos são armazenados ordenadamente conforme implementação IComparer<T>

Alguns métodos importantes

- Add
- Clear
- Contains
- UnionWith(other) - operação união: adiciona no conjunto os elementos do outro conjunto, sem repetição
- IntersectWith(other) - operação interseção: remove do conjunto os elementos não contidos em other
- ExceptWith(other) - operação diferença: remove do conjunto os elementos contidos em other
- Remove(T)
- RemoveWhere(predicate)

Demo 1

```
using System;
using System.Collections.Generic;

namespace Course {
    class Program {

        static void Main(string[] args) {
            HashSet<string> set = new HashSet<string>();

            set.Add("TV");
            set.Add("Notebook");
            set.Add("Tablet");

            Console.WriteLine(set.Contains("Notebook"));

            foreach (String p in set) {
                Console.WriteLine(p);
            }
        }
    }
}
```

Demo 2

```
using System;
using System.Collections.Generic;

namespace Course {
    class Program {

        static void Main(string[] args) {
            SortedSet<int> a = new SortedSet<int>() { 0, 2, 4, 5, 6, 8, 10 };
            SortedSet<int> b = new SortedSet<int>() { 5, 6, 7, 8, 9, 10 };

            //union
            SortedSet<int> c = new SortedSet<int>(a);
            c.UnionWith(b);
            printCollection(c);

            //intersection
            SortedSet<int> d = new SortedSet<int>(a);
            d.IntersectWith(b);
            printCollection(d);

            //difference
            SortedSet<int> e = new SortedSet<int>(a);
            e.ExceptWith(b);
            printCollection(e);
        }

        static void printCollection<T>(IEnumerable<T> collection) {
            foreach(T obj in collection) {
                Console.Write(obj + " ");
            }
            Console.WriteLine();
        }
    }
}
```

Como as coleções Hash testam igualdade?

http://educandoweb.com.br

Prof. Dr. Nelio Alves

Como as coleções Hash testam igualdade?

- Se GetHashCode e Equals estiverem implementados:
  - Primeiro GetHashCode. Se der igual, usa Equals para confirmar.

- Se GetHashCode e Equals **NÃO** estiverem implementados:
  - Tipos referência: compara as referências dos objetos
  - Tipos valor: comparar os valores dos atributos

```
namespace Course.Entities {
    struct Point {
        public int X { get; set; }
        public int Y { get; set; }

        public Point(int x, int y) : this() {
            X = x;
            Y = y;
        }
    }
}
```

```
namespace Course.Entities {
    class Product {

        public string Name { get; set; }
        public double Price { get; set; }

        public Product(string name, double price) {
            Name = name;
            Price = price;
        }
    }
}
```

```
using System;
using System.Collections.Generic;
using Course.Entities;

namespace Course {
    class Program {
        static void Main(string[] args) {

            HashSet<Product> a = new HashSet<Product>();
            a.Add(new Product("TV", 900.0));
            a.Add(new Product("Notebook", 1200.0));

            HashSet<Point> b = new HashSet<Point>();
            b.Add(new Point(3, 4));
            b.Add(new Point(5, 10));

            Product prod = new Product("Notebook", 1200.0);
            Console.WriteLine(a.Contains(prod));

            Point point = new Point(5, 10);
            Console.WriteLine(b.Contains(point));
        }
    }
}
```

Exercício resolvido (Set)

http://educandoweb.com.br

Prof. Dr. Nelio Alves

Problema exemplo

Um site de internet registra um log de acessos dos usuários. Um registro de log consiste no nome de usuário e o instante em que o usuário acessou o site no padrão ISO 8601, separados por espaço, conforme exemplo. Fazer um programa que leia o log de acessos a partir de um arquivo, e daí informe quantos usuários distintos acessaram o site.

Example

input file:

```
amanda 2020-08-26T20:45:08
alex86 2020-08-26T21:49:37
bobbrown 2020-08-27T03:19:13
amanda 2020-08-27T08:11:00
jeniffer3 2020-08-27T09:19:24
alex86 2020-08-27T22:39:52
amanda 2020-08-28T07:42:19
```

Execution:

```
Enter file full path: c:\temp\in.txt
Total users: 4
```

https://github.com/acenelio/set1-csharp

```
using System;
using System.IO;

namespace Course {
    class Program {
        static void Main(string[] args) {

            Console.Write("Enter file full path: ");
            string path = Console.ReadLine();

            try {
                using (StreamReader sr = File.OpenText(path)) {
                    while (!sr.EndOfStream) {
                        string line = sr.ReadLine();
                        Console.WriteLine(line);
                    }
                }
            }
            catch (IOException e) {
                Console.WriteLine(e.Message);
            }
        }
    }
}
```

Exercício proposto (Set)

http://educandoweb.com.br

Prof. Dr. Nelio Alves

Em um portal de cursos online, cada usuário possui um código único, representado por um número inteiro.

Cada instrutor do portal pode ter vários cursos, sendo que um mesmo aluno pode se matricular em quantos cursos quiser. Assim, o número total de alunos de um instrutor não é simplesmente a soma dos alunos de todos os cursos que ele possui, pois pode haver alunos repetidos em mais de um curso.

O instrutor Alex possui três cursos A, B e C, e deseja saber seu número total de alunos.

Seu programa deve ler os alunos dos cursos A, B e C do instrutor Alex, depois mostrar a quantidade total e alunos dele, conforme exemplo.

https://github.com/acenelio/set2-csharp

Example:

```
How many students for course A? 3
21
35
22
How many students for course B? 2
21
50
How many students for course C? 3
42
35
13
Total students: 6
```

Dictionary e SortedDictionary

http://educandoweb.com.br

Prof. Dr. Nelio Alves

Dictionary<TKey, TValue>

- É uma coleção de pares chave / valor
  - Não admite repetições do objeto chave
  - Os elementos são indexados pelo objeto chave (não possuem posição)
  - Acesso, inserção e remoção de elementos são rápidos
- Uso comum: cookies, local storage, qualquer modelo chave-valor
- Dictionary
  - https://msdn.microsoft.com/en-us/library/xfhwa508(v=vs.110).aspx
- SortedDictionary
  - https://msdn.microsoft.com/en-us/library/f7fta44c(v=vs.110).aspx

Diferenças

- Dictionary
  - Armazenamento em tabela hash
  - Extremamente rápido: inserção, remoção e busca O(1)
  - A ordem dos elementos não é garantida
- SortedDictionary
  - Armazenamento em árvore
  - Rápido: inserção, remoção e busca O(log(n))
  - Os elementos são armazenados ordenadamente conforme implementação IComparer<T>

Alguns métodos importantes

- dictionary[key] - acessa o elemento pela chave informada
- Add(key, value) - adiciona elemento (exceção em caso de repetição)
- Clear() - esvazia a coleção
- Count - quantidade de elementos
- ContainsKey(key) - verifica se a dada chave existe
- ContainsValue(value) - verifica se o dado valor existe
- Remove(key) - remove um elemento pela chave

Demo

```
using System;
using System.Collections.Generic;

namespace Course {
    class Program {
        static void Main(string[] args) {

            Dictionary<string, string> cookies = new Dictionary<string, string>();

            cookies["user"] = "maria";
            cookies["email"] = "maria@gmail.com";
            cookies["phone"] = "99771122";
            cookies["phone"] = "99771133";

            Console.WriteLine(cookies["email"]);

            cookies.Remove("email");

            Console.WriteLine("Phone number: " + cookies["phone"]);

            if (cookies.ContainsKey("email")) {
                Console.WriteLine("Email: " + cookies["email"]);
            }
            else {
                Console.WriteLine("There is not 'email' key");
            }

            Console.WriteLine("Size: " + cookies.Count);

            Console.WriteLine("ALL COOKIES:");
            foreach (KeyValuePair<string, string> item in cookies) {
                Console.WriteLine(item.Key + ": " + item.Value);
            }
        }
    }
}
```

Exercício proposto (Dictionary)

http://educandoweb.com.br

Prof. Dr. Nelio Alves

Na contagem de votos de uma eleição, são gerados vários registros de votação contendo o nome do candidato e a quantidade de votos (formato .csv) que ele obteve em uma urna de votação. Você deve fazer um programa para ler os registros de votação a partir de um arquivo, e daí gerar um relatório consolidado com os totais de cada candidato.

Input file example:

```
Alex Blue,15
Maria Green,22
Bob Brown,21
Alex Blue,30
Bob Brown,15
Maria Green,27
Maria Green,22
Bob Brown,25
Alex Blue,31
```

Execution:

```
Enter file full path: c:\temp\in.txt
Alex Blue: 76
Maria Green: 71
Bob Brown: 61
```

Solução do exercício

https://github.com/acenelio/dictionary1-csharp