#### Делегаты Action<T>, Func<T>, Predicate<T>

using System;

using System.Windows.Forms;

delegate void DisplayMessage(string message);

public class TestCustomDelegate

{

public static void Main()

{

DisplayMessage messageTarget;

if (Environment.GetCommandLineArgs().Length > 1)

messageTarget = ShowWindowsMessage;

else

messageTarget = Console.WriteLine;

messageTarget("Hello, World!");

}

private static void ShowWindowsMessage(string message)

{

MessageBox.Show(message);

}

}

Перепишем код с использованием стандартных делегатов

using System;

using System.Windows.Forms;

public class TestAction1

{

public static void Main()

{

Action<string> messageTarget;

if (Environment.GetCommandLineArgs().Length > 1)

#### messageTarget = ShowWindowsMessage;// delegate(string s) { ShowWindowsMessage(s); };// s => ShowWindowsMessage(s);

else

#### messageTarget = Console.WriteLine;// delegate(string s) { Console.WriteLine(s); };// s => Console.WriteLine(s);

messageTarget("Hello, World!");

}

private static void ShowWindowsMessage(string message)

{

MessageBox.Show(message);

}

}

#### Делегаты **Func<T>**

static void Main(string[] args)

{

Func<int, int> retFunc = Factorial;

int n1 = GetInt(6, retFunc);

Console.WriteLine(n1); // 720

int n2 = GetInt(6, x=> x \*x);

Console.WriteLine(n2); // 36

Console.Read();

}

static int GetInt(int x1, Func<int, int> retF)

{

int result = 0;

if (x1 > 0)

result = retF(x1);

return result;

}

static int Factorial(int x)

{

int result = 1;

for (int i = 1; i <= x; i++)

{

result \*= i;

}

return result;

}

#### Predicate<T>

Predicate<int> isPositive = delegate (int x) { return x > 0; };

Predicate<int> isPlus = x => x > 0;

Console.WriteLine(isPositive(20));

Console.WriteLine(isPositive(-20));

#### Размер коллекции в памяти

using System;

using System.Collections.Generic;

public class Part {

public string PartName { get; set; }

public int PartId { get; set; }

public override string ToString()

{

return "ID: " + PartId + " Name: " + PartName;

}

}

public class Example

{

public static void Main()

{

List<Part> parts = new List<Part>(20);

Console.WriteLine("\nCapacity: {0}", parts.Capacity);

parts.Add(new Part() { PartName = "1", PartId = 1234 });

parts.Add(new Part() { PartName = "2", PartId = 1334 });

parts.Add(new Part() { PartName = "3", PartId = 1434 });

parts.Add(new Part() { PartName = "4", PartId = 1534 });

parts.Add(new Part() { PartName = "5", PartId = 1634 });

//parts.Add(new Part() { PartName = "6", PartId = 1534 });

//parts.Add(new Part() { PartName = "7", PartId = 1634 }); ;

Console.WriteLine();

foreach (Part aPart in parts)

{

Console.WriteLine(aPart);

}

Console.WriteLine("\nCapacity: {0}", parts.Capacity);

Console.WriteLine("Count: {0}", parts.Count);

parts.TrimExcess();

Console.WriteLine("\nTrimExcess()");

Console.WriteLine("Capacity: {0}", parts.Capacity);

Console.WriteLine("Count: {0}", parts.Count);

parts.Add(new Part() { PartName = "6", PartId = 1234 });

parts.Add(new Part() { PartName = "7", PartId = 1334 });

parts.Add(new Part() { PartName = "8", PartId = 1434 });

parts.Add(new Part() { PartName = "9", PartId = 1534 });

//parts.Add(new Part() { PartName = "10", PartId = 1634 });

//parts.Add(new Part() { PartName = "11", PartId = 1634 });

Console.WriteLine("Capacity: {0}", parts.Capacity);

Console.WriteLine("Count: {0}", parts.Count);

parts.Clear();

Console.WriteLine("\nClear()");

Console.WriteLine("Capacity: {0}", parts.Capacity);

Console.WriteLine("Count: {0}", parts.Count);

}

}

#### IList

public class Person {

        public string FirstName;

        public string LastName;

        public int Age;

    }

static void UseGenericList()

{

List<Person> people = new List<Person>()

{

new Person {FirstName= "Homer", LastName="Simpson", Age=47},

new Person {FirstName= "Marge", LastName="Simpson", Age=45},

new Person {FirstName= "Lisa", LastName="Simpson", Age=9},

new Person {FirstName= "Bart", LastName="Simpson", Age=8}

};

Console.WriteLine("Items in list: {0}", people.Count);

foreach (Person p in people)

Console.WriteLine(p);

Console.WriteLine("\n->Inserting new person.");

people.Insert(2, new Person { FirstName = "Maggie", LastName = "Simpson", Age = 2 });

Console.WriteLine("Items in list: {0}", people.Count);

Person[] arrayOfPeople = people.ToArray();

for (int i = 0; i < arrayOfPeople.Length; i++)

{

Console.WriteLine("First Names: {0}", arrayOfPeople[i].FirstName);

}

}

#### Queue<T>

using System;

using System.Collections.Generic;

namespace Collections

{

class Program

{

static void Main(string[] args)

{

Queue<int> numbers = new Queue<int>();

numbers.Enqueue(3); // очередь 3

numbers.Enqueue(5); // очередь 3, 5

numbers.Enqueue(8); // очередь 3, 5, 8

// получаем первый элемент очереди

int queueElement = numbers.Dequeue(); //теперь очередь 5, 8

Console.WriteLine(queueElement);

Queue<Person> persons = new Queue<Person>();

persons.Enqueue(new Person() { Name = "Tom" });

persons.Enqueue(new Person() { Name = "Bill" });

persons.Enqueue(new Person() { Name = "John" });

// получаем первый элемент без его извлечения

Person pp = persons.Peek();

Console.WriteLine(pp.Name);

Console.WriteLine("Сейчас в очереди {0} человек", persons.Count);

// теперь в очереди Tom, Bill, John

foreach (Person p in persons)

{

Console.WriteLine(p.Name);

}

// Извлекаем первый элемент в очереди - Tom

Person person = persons.Dequeue(); // теперь в очереди Bill, John

Console.WriteLine(person.Name);

Console.ReadLine();

}

}

class Person

{

public string Name { get; set; }

}

}

#### Stack<T>

static void UseGenericStack()

{

Stack<Person> stackOfPeople = new Stack<Person>();

stackOfPeople.Push(new Person

{ FirstName = "Homer", LastName = "Simpson", Age = 47 });

stackOfPeople.Push(new Person

{ FirstName = "Marge", LastName = "Simpson", Age = 45 });

stackOfPeople.Push(new Person

{ FirstName = "Lisa", LastName = "Simpson", Age = 9 });

// Просмотреть верхний элемент, вытолкнуть его и просмотреть снова .

Console.WriteLine("First person is: {0}", stackOfPeople.Peek());

Console.WriteLine("Popped off {0}", stackOfPeople.Pop());

Console.WriteLine("\nFirst person is: {0}", stackOfPeople.Peek());

Console.WriteLine("Popped off {0}", stackOfPeople.Pop());

Console.WriteLine("\nFirst person item is: {0}", stackOfPeople.Peek());

Console.WriteLine("Popped off {0}", stackOfPeople.Pop());

try

{

Console.WriteLine("\nFirst person is: {0}", stackOfPeople.Peek());

Console.WriteLine("Popped off {0}", stackOfPeople.Pop());

}

catch (InvalidOperationException ex)

{

Console.WriteLine("\nError! {0}", ex.Message); // Ошибка Стек пуст.

}

}

#### LinkedList<T>

class Program

{

static void Main()

{

// Создадим связный список

LinkedList<string> link = new LinkedList<string>();

// Добавим несколько элементов

link.AddFirst("Alex");

link.AddFirst("Djek");

link.AddFirst("Bob");

link.AddFirst("Doug");

// Отобразим элементы в прямом направлении

LinkedListNode<string> node;

Console.WriteLine("Элементы коллекции в прямом направлении: ");

for (node = link.First; node != null; node = node.Next)

Console.Write(node.Value + "\t");

// Отобразить элементы в обратном направлении

Console.WriteLine("\n\nЭлементы коллекции в обратном направлении: ");

for (node = link.Last; node != null; node = node.Previous)

Console.Write(node.Value + "\t");

Console.ReadLine();

}

}

#### Dictionary<T, V>

Dictionary<int, string> countries = new Dictionary<int, string>(5);

countries.Add(1, "Russia");

countries.Add(3, "Great Britain");

countries.Add(2, "USA");

countries.Add(4, "France");

countries.Add(5, "China");

foreach (KeyValuePair<int, string> keyValue in countries)

{

Console.WriteLine(keyValue.Key + " - " + keyValue.Value);

}

// получение элемента по ключу

string country = countries[4];

// изменение объекта

countries[4] = "Spain";

// удаление по ключу

countries.Remove(2);

#### LINQ

#### Фильтрация

var resultfilter = (from emp in employes

where emp.Name.Contains("a")

select emp).ToList();

#### Порядок

var result4 = (from emp in employes

where emp.Age >= 30

orderby emp.Salary

select emp).ToList();

#### Группировка

var query = (from emp in employes

group emp by emp.DepartmentId into custGroup

//where custGroup.Count() > 2

orderby custGroup.Key

select custGroup).ToList();

#### Соединение

var result0 = (from emp in employes

join dep in departments on emp.DepartmentId equals dep.Id

select new { emp, dep }).ToList();

#### Выбор (Проецирование)

var result6 = (from emp in employes

join dep in departments on emp.DepartmentId equals dep.Id into gr

from g in gr.DefaultIfEmpty(new Department() { Id = 0, Name = "Нет" })

select new { emp.Name, DepartmentName = g.Name }).ToList();