Пермский филиал федерального государственного автономного образовательного учреждения высшего образования   
«Национальный исследовательский университет   
«Высшая школа экономики»

*Факультет экономики, менеджмента и бизнес-информатики*

Шешин Николай Андреевич

**Отчет по лабораторной работе №11**

студента образовательной программы «Программная инженерия»

по направлению подготовки *09.03.04 Программная инженерия*

Руководитель

доцент кафедры

информационных

технологий в бизнесе

к.т.н. О.Л. Викентьева

Пермь, 2020 год

**Постановка задачи 1 и 2**

**Порядок выполнения работы.**

### 2.1. Задание 1.

* 1. Создать коллекцию, в которую добавить объекты созданной иерархии классов.
  2. Используя меню, реализовать в программе добавление и удаление объектов коллекции.
  3. Разработать и реализовать три запроса (количество элементов определенного вида, печать элементов определенного вида и т.п.).
  4. Выполнить перебор элементов коллекции с помощью метода foreach.
  5. Выполнить клонирование коллекции.
  6. Выполнить сортировку коллекции (если коллекция не отсортирована) и поиск заданного элемента в коллекции.

При работе с коллекцией использовать объекты из иерархии классов, разработанной в работе №10.

### 2.2. Задание 2.

* 1. Создать обобщенную коллекцию, в которую добавить объекты созданной иерархии классов.
  2. Используя меню, реализовать в программе добавление и удаление объектов коллекции.
  3. Разработать и реализовать три запроса (количество элементов определенного вида, печать элементов определенного вида и т.п.).
  4. Выполнить перебор элементов коллекции с помощью метода foreach.
  5. Выполнить клонирование коллекции.
  6. Выполнить сортировку коллекции (если коллекция не отсортирована) и поиск заданного элемента в коллекции.

При работе с коллекцией использовать объекты из иерархии классов, разработанной в работе №10.

### 2.3. Задание 3.

1. Создать иерархию классов (базовый – производный) в соответствии с вариантом (см. лаб. раб. №10).
2. В производном классе определить свойство, которое возвращает ссылку на объект базового класса (это свойство должно возвращать ссылку на объект базового класса, а не ссылку на вызывающий объект производного класса). Например, для иерархии классов Person-Student в классе производном классе Student можно определить свойство

public Person BasePerson

{

get

{

return new Person(name, age);//возвращает объект базового класса

}

}

1. Определить класс TestCollections, который содержит поля следующих типов

Коллекция\_1<TKey> ;

Коллекция\_1<string> ;

Коллекция\_2<TKey, TValue> ;

Коллекция\_2<string, TValue> .

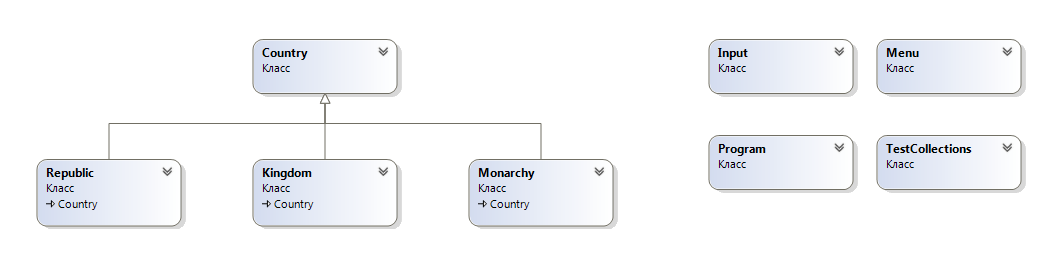
где тип ключа TKey и тип значения TValue связаны отношением базовый-производный (см. задание 1), Коллекция\_1 и Коллекция\_2 – коллекции из пространства имен System.Collections.Generic.

1. Написать конструктор класса TestCollections, в котором создаются коллекции с заданным числом элементов.
2. Предусмотреть автоматическую генерацию элементов коллекции таким образом, что каждый объект (Student) содержит подобъект базового класса (Person).
3. Все четыре коллекции должны содержать одинаковое число элементов. Каждому элементу из коллекции Коллекция\_1<TKey> должен отвечать элемент в коллекции Коллекция\_2<TKey, TValue> с равным значением ключа. Список Коллекция\_1<string> состоит из строк, которые получены в результате вызова метода ToString() для объектов TKey из списка Коллекция\_1<TKey>. Каждому элементу списка Коллекция\_1<string> отвечает элемент в Коллекция\_2 <string, TValue> с равным значением ключа типа string.
4. Для четырех разных элементов – первого, центрального, последнего и элемента, не входящего в коллекцию – надо измерить время поиска элемента в коллекциях Коллекция\_1<TKey> и Коллекция\_1<string> с помощью метода Contains; элемента по ключу в коллекциях Коллекция\_2< TKey, TValue> и Коллекция\_2 <string, TValue > с помощью метода ContainsKey; значения элемента в коллекции Коллекция\_2< TKey, TValue > с помощью метода ContainsValue. Обратите внимание на то, что искать нужно сами элементы, а не ссылки на них!
5. Предусмотреть методы для работы с коллекциями (добавление и удаление элементов).

**Вариант**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | List <T> | Dictionary <K,T> | List <T> | Dictionary <K,T> |

**Диаграмма классов для заданий 1, 2**



**Описание методов, используемых для работы с коллекцией**

***List<T>***

* ***Add() – добавляет элемент в List.***
* ***RemoveAt() – удаляет элемент из List***

***Dictionary<T1, T2>***

* ***TryAdd() – метод для попытки добавления в Dictionary***
* ***Remove() – удаляет элемент из Dictionary***
* ***ContainsKey() – проверяет нахождение элемента в Dictionary по ключу***

**Листинг программы**

***Program.cs***

using System;

using System.Collections.Generic;

using System.Collections;

namespace lab11

{

class Menu

{

public delegate void Request();

private List<String> options = new List<string>();

private List<Request> requests = new List<Request>();

public Menu(List<string> options, List<Request> requests)

{

this.options = options;

this.requests = requests;

}

public void Show()

{

int curOption = -1;

while (curOption != 0)

{

Console.Clear();

for (int i = 0; i < options.Count; i++)

{

Console.WriteLine((i + 1).ToString() + ") " + options[i]);

}

Console.WriteLine("0) Выход");

curOption = Input.Int("Выберите опцию:", 0, requests.Count + 1);

if (curOption == 0)

break;

requests[curOption - 1]();

Console.WriteLine("Нажмите любую клавишу");

Console.ReadKey();

}

}

}

class Program

{

public static Program program = new Program();

#region task1Funcs

public List<Country> countries = new List<Country>();

public void PrintList()

{

int i = 0;

Console.Clear();

if (countries.Count == 0)

{

Console.WriteLine("List пуст.");

return;

}

foreach (Country country in countries)

{

Console.Write(++i + ") ");

country.Print();

}

}

public void AddToList()

{

Console.Clear();

Console.WriteLine("1) Королевство");

Console.WriteLine("2) Монархия");

Console.WriteLine("3) Республика");

Console.WriteLine("4) Неизвестно");

Console.WriteLine("0) Выход");

int option = Input.Int("Выберите тип государства: ", 0, 5);

switch (option)

{

case 1:

{

Kingdom kingdom = new Kingdom();

kingdom.In();

countries.Add(kingdom);

break;

}

case 2:

{

Monarchy monarchy = new Monarchy();

monarchy.In();

countries.Add(monarchy);

break;

}

case 3:

{

Republic republic = new Republic();

republic.In();

countries.Add(republic);

break;

}

case 4:

{

Country country = new Country();

country.In();

countries.Add(country);

break;

}

case 0:

{

return;

}

}

}

public void DeleteFromList()

{

Console.Clear();

if (countries.Count == 0)

{

Console.WriteLine("List пуст.");

return;

}

int index = Input.Int($"Введите номер нужно элемента: (от 1 до { countries.Count})", 1, countries.Count);

countries.RemoveAt(index - 1);

}

public void FindKings()

{

int i = 0;

if (countries.Count == 0)

{

Console.WriteLine("List пуст.");

return;

}

Console.Clear();

foreach (Country country in countries)

{

if (country is Kingdom)

{

Console.WriteLine(++i + $") {(country as Kingdom).GetKing()}");

}

}

if (i == 0)

{

Console.WriteLine("Ничего не найдено");

}

}

public void FindNamesOfRepublics()

{

int i = 0;

if (countries.Count == 0)

{

Console.WriteLine("List пуст.");

return;

}

Console.Clear();

foreach (Country country in countries)

{

if (country is Republic)

{

Console.WriteLine(++i + $") {country.GetName()}");

}

}

if (i == 0)

{

Console.WriteLine("Ничего не найдено");

}

}

public void FindPopulationElevenMillion()

{

int i = 0;

if (countries.Count == 0)

{

Console.WriteLine("List пуст.");

return;

}

Console.Clear();

foreach (Country country in countries)

{

if (country.GetPopulation() > 11e6)

{

i++;

}

}

Console.WriteLine("Кол-во таких государств: " + i);

}

public void SortListByName()

{

if (countries.Count == 0)

{

Console.WriteLine("List пуст.");

return;

}

for (int i = 0; i < countries.Count - 1; i++)

{

if (String.Compare(countries[i].GetName(), countries[i + 1].GetName()) == 1)

{

string tmp = countries[i].GetName();

countries[i].SetName(countries[i + 1].GetName());

countries[i + 1].SetName(tmp);

i = (i > 1) ? i - 2 : i;

}

}

PrintList();

}

public void FindInListByName()

{

Console.Clear();

Console.WriteLine("Введите название государства: ");

string nameToFind = Console.ReadLine();

for (int i = 0; i < countries.Count; i++)

{

if (nameToFind == countries[i].GetName())

{

Console.WriteLine("Номер страны в списке - " + (i + 1));

return;

}

}

Console.WriteLine("Страна не найдена.");

}

#endregion

#region task2Funcs

public Dictionary<int, Country> countriesD = new Dictionary<int, Country>();

public void PrintDictionary()

{

Console.Clear();

if (countriesD.Count == 0)

{

Console.WriteLine("Dictionary пуст.");

return;

}

foreach (int i in countriesD.Keys)

{

Console.Write($"К году { i } относится: ");

countriesD[i].Print();

}

}

public void AddToDictionary()

{

Console.Clear();

Console.WriteLine("1) Королевство");

Console.WriteLine("2) Монархия");

Console.WriteLine("3) Республика");

Console.WriteLine("4) Неизвестно");

Console.WriteLine("0) Выход");

int option = Input.Int("Выберите тип государства: ", 0, 5);

switch (option)

{

case 1:

{

Kingdom kingdom = new Kingdom();

int dateIndex;

kingdom.In();

dateIndex = Input.Int("Введите год появления государства: ", 0, 2021);

if (!countriesD.TryAdd(dateIndex, kingdom))

{

Console.WriteLine("Ошибка добавления");

}

break;

}

case 2:

{

Monarchy monarchy = new Monarchy();

monarchy.In();

int dateIndex = Input.Int("Введите год появления государства: ", 0, 2021);

if (!countriesD.TryAdd(dateIndex, monarchy))

{

Console.WriteLine("Ошибка добавления");

}

break;

}

case 3:

{

Republic republic = new Republic();

republic.In();

int dateIndex = Input.Int("Введите год появления государства: ", 0, 2021);

if (!countriesD.TryAdd(dateIndex, republic))

{

Console.WriteLine("Ошибка добавления");

}

break;

}

case 4:

{

Country country = new Country();

country.In();

int dateIndex = Input.Int("Введите год появления государства: ", 0, 2021);

if (!countriesD.TryAdd(dateIndex, country))

{

Console.WriteLine("Ошибка добавления");

}

break;

}

case 0:

{

return;

}

}

}

public void DeleteFromDictionary()

{

Console.Clear();

if (countriesD.Count == 0)

{

Console.WriteLine("Dictionary пуст.");

return;

}

int dataIndex = Input.Int($"Введите год появления государства:" );

if (!countriesD.Remove(dataIndex))

{

Console.WriteLine("Государство не было найдено");

}

}

public void FindCountriesInFirstThousand()

{

Console.Clear();

if (countriesD.Count == 0)

{

Console.WriteLine("Dictionary пуст.");

return;

}

foreach (int i in countriesD.Keys)

{

if (i < 1001 && i > 0)

{

countriesD[i].Print();

}

}

}

public void FindCountriesInSecondThousand()

{

Console.Clear();

if (countriesD.Count == 0)

{

Console.WriteLine("Dictionary пуст.");

return;

}

foreach (int i in countriesD.Keys)

{

if (i < 2001 && i > 1000)

{

countriesD[i].Print();

}

}

}

public void FindMonarchies()

{

Console.Clear();

if (countriesD.Count == 0)

{

Console.WriteLine("Dictionary пуст.");

return;

}

foreach (int i in countriesD.Keys)

{

if (countriesD[i] is Monarchy)

{

countriesD[i].Print();

}

}

}

public void SortDictionaryByYear()

{

if (countriesD.Count == 0)

{

Console.WriteLine("Dictionary пуст.");

return;

}

Dictionary<int, Country> helpDictionary = new Dictionary<int, Country>();

for (int i = 0; i < 2021; i++)

{

if (countriesD.ContainsKey(i))

{

helpDictionary.Add(i, countriesD[i]);

}

}

countriesD = helpDictionary;

PrintDictionary();

}

public void FindInDictionaryByName()

{

Console.Clear();

Console.WriteLine("Введите название государства: ");

string nameToFind = Console.ReadLine();

foreach (int i in countriesD.Keys)

{

if (nameToFind == countriesD[i].GetName())

{

countriesD[i].Print();

return;

}

}

Console.WriteLine("Страна не найдена.");

}

#endregion

#region task3Funcs

public void StartCheckContain()

{

TestCollections testCollections = new TestCollections();

testCollections.ContainTime1();

testCollections.ContainTime2();

testCollections.ContainTime3();

testCollections.ContainTime4();

testCollections.ContainTime5();

testCollections.ContainTime6();

}

#endregion

public static void Main(string[] args)

{

Menu task1 = new Menu(new List<string>(new string[] { "Вывести коллекцию", "Добавить страну в коллекцию", "Удалить страну из коллекции", "Вывести имена всех королей",

"Вывести названия всех республик", "Вывести количество государств с населением больше 11000000 людей",

"Отсортировать коллекцию по названию", "Найти элемент в коллекции по названию"}),

new List<Menu.Request>() { program.PrintList, program.AddToList, program.DeleteFromList, program.FindKings,

program.FindNamesOfRepublics, program.FindPopulationElevenMillion,

program.SortListByName, program.FindInListByName });

Menu task2 = new Menu(new List<string>(new string[] { "Вывести обобщенную коллекцию", "Добавить страну в обобщенную коллекцию", "Удалить страну из обобщенной коллекции",

"Вывести государства, появившиеся в 1-ом тысячелетии", "Вывести государства, появивишиеся во 2-ом тысячелетии",

"Вывести все монархии", "Отсортировать обобщенный список по имени", "Найти элемент в обобщенном списке по имени"}),

new List<Menu.Request>() { program.PrintDictionary, program.AddToDictionary, program.DeleteFromDictionary,

program.FindCountriesInFirstThousand, program.FindCountriesInSecondThousand, program.FindMonarchies,

program.SortDictionaryByYear, program.FindInDictionaryByName});

Menu mainMenu = new Menu(new List<string>(new string[] { "Задание 1", "Задание 2"}), new List<Menu.Request>(new Menu.Request[] { task1.Show, task2.Show }));

mainMenu.Show();

}

}

}

***Country.cs***

using System;

using System.Collections;

using System.Collections.Generic;

using System.Text;

namespace lab11

{

class Country

{

protected string name;

protected int population;

public Country(string name, int population)

{

this.name = name;

this.population = population;

}

public Country()

{

}

public string GetName()

{

return name;

}

public void SetName(string name)

{

this.name = name;

}

public int GetPopulation()

{

return population;

}

virtual public void In()

{

Console.WriteLine("Введите название государства: ");

name = Console.ReadLine();

population = Input.Int("Введите кол-во жителей: ");

}

virtual public void Print()

{

Console.WriteLine($"Страна - {name}, численность населения - {population}.");

}

}

class Kingdom : Country

{

protected string king;

public Kingdom(string name, int population, string king) : base(name, population)

{

this.king = king;

}

public Kingdom()

{

}

public Country BaseCountry

{

get

{

return new Country(name, population);

}

}

public string GetKing()

{

return king;

}

override public void In()

{

Console.WriteLine("Введите название королевства: ");

name = Console.ReadLine();

population = Input.Int("Введите кол-во жителей: ");

Console.WriteLine("Введите короля: ");

king = Console.ReadLine();

}

override public void Print()

{

Console.WriteLine($"Королевство - {name}, король - {king}, численность населения - {population}.");

}

}

class Monarchy : Country

{

protected string imperor;

public Monarchy(string name, int population, string imperor) : base(name, population)

{

this.imperor = imperor;

}

public Monarchy()

{

}

public Country BaseCountry

{

get

{

return new Country(name, population);

}

}

public string GetImperor()

{

return imperor;

}

override public void In()

{

Console.WriteLine("Введите название монархии: ");

name = Console.ReadLine();

population = Input.Int("Введите кол-во жителей: ");

Console.WriteLine("Введите императора: ");

imperor = Console.ReadLine();

}

override public void Print()

{

Console.WriteLine($"Монархия - {name}, император - {imperor}, численность населения - {population}.");

}

}

class Republic : Country

{

protected string president;

public Republic(string name, int population, string president) : base(name, population)

{

this.president = president;

}

public Republic()

{

}

public Country BaseCountry

{

get

{

return new Country(name, population);

}

}

public string GetPresident()

{

return president;

}

override public void In()

{

Console.WriteLine("Введите название республики: ");

name = Console.ReadLine();

population = Input.Int("Введите кол-во жителей: ");

Console.WriteLine("Введите президента: ");

president = Console.ReadLine();

}

override public void Print()

{

Console.WriteLine($"Республика - {name}, президент - {president}, численность населения - {population}.");

}

}

}

***Input.cs***

using System;

using System.Collections.Generic;

using System.Text;

namespace lab11

{

public class Input

{

public static int Int(string message = "Введите число: ", int min = int.MinValue, int max = int.MaxValue, string messageError = "Неверный ввод!")

{

int number;

Console.WriteLine(message);

while (!int.TryParse(Console.ReadLine(), out number) || number < min || number >= max)

{

Console.WriteLine(messageError);

}

return number;

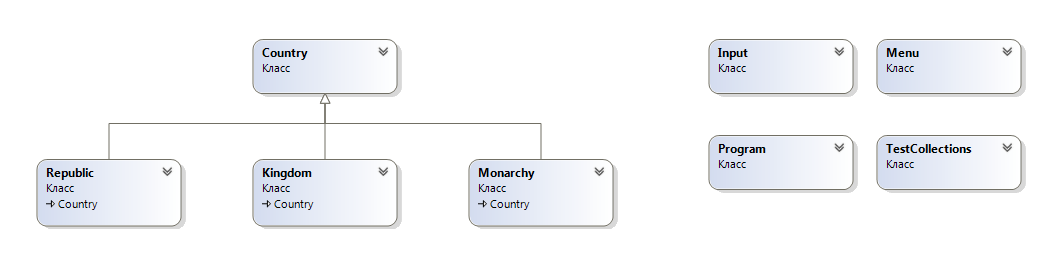
}

}

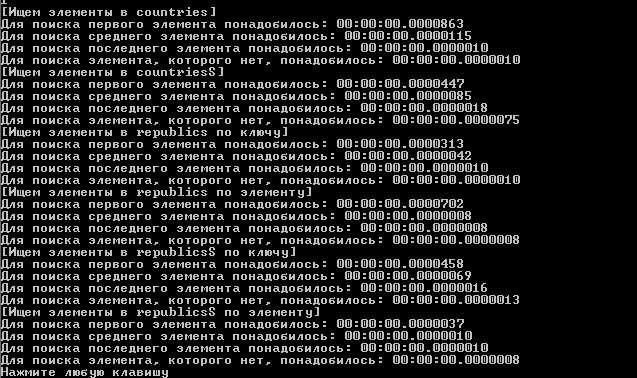
}

**Постановка задачи 3**

**Диаграмма классов**



**Результат программы**



**Анализ**

**TestCollections.cs (в демонстрационной программе запускаются все методы ContainTime по очереди)**

using System;

using System.Collections.Generic;

using System.Text;

namespace lab11

{

class TestCollections

{

int count;

List<Country> countries = new List<Country>();

List<string> countriesS = new List<string>();

Dictionary<Country, Republic> republics = new Dictionary<Country, Republic>();

Dictionary<string, Republic> republicsS = new Dictionary<string, Republic>();

public TestCollections()

{

List<Republic> list = new List<Republic>();

countries = new List<Country>();

countriesS = new List<string>();

republics = new Dictionary<Country, Republic>();

republicsS = new Dictionary<string, Republic>();

list.Add(new Republic("Афганистан", 31500000, "Ашраф Гани"));

list.Add(new Republic("Албания", 2876591, "Илир Мета"));

list.Add(new Republic("Алжир", 38087012, "Абдельмаджид Теббун"));

list.Add(new Republic("Аргентина", 43417000, "Альберто Фернандес"));

list.Add(new Republic("Германия", 83019200, "Франк-Вальтер Штайнмайер"));

for (int i = 0; i < list.Count; i++)

{

Republic republic = list[i];

countries.Add(republic as Country);

countriesS.Add((republic as Country).ToString());

republics.Add(republic as Country, republic);

republicsS.Add((republic as Country).ToString(), republic);

}

}

public TestCollections(int count)

{

this.count = count;

countries = new List<Country>();

countriesS = new List<string>();

republics = new Dictionary<Country, Republic>();

republicsS = new Dictionary<string, Republic>();

for (int i = 0; i < this.count; i++)

{

Republic republic = new Republic();

republic.In();

countries.Add(republic as Country);

countriesS.Add((republic as Country).ToString());

republics.Add(republic as Country, republic);

republicsS.Add((republic as Country).ToString(), republic);

}

}

public void Add(Republic republic)

{

countries.Add(republic as Country);

countriesS.Add((republic as Country).ToString());

republics.Add(republic as Country, republic);

republicsS.Add((republic as Country).ToString(), republic);

}

public void Delete(Republic republic)

{

countries.Remove(republic);

countriesS.Remove((republic as Country).ToString());

republics.Remove(republic as Country);

republicsS.Remove((republic as Country).ToString());

}

public void ContainTime1()

{

Console.WriteLine("[Ищем элементы в countries]");

System.Diagnostics.Stopwatch stopWatch = new System.Diagnostics.Stopwatch();

stopWatch.Start();

countries.Contains(new Republic("Афганистан", 31500000, "Ашраф Гани") as Country);

stopWatch.Stop();

Console.WriteLine("Для поиска первого элемента понадобилось: " + stopWatch.Elapsed);

stopWatch = new System.Diagnostics.Stopwatch();

stopWatch.Start();

countries.Contains(new Republic("Алжир", 38087012, "Абдельмаджид Теббун") as Country);

stopWatch.Stop();

Console.WriteLine("Для поиска среднего элемента понадобилось: " + stopWatch.Elapsed);

stopWatch = new System.Diagnostics.Stopwatch();

stopWatch.Start();

countries.Contains(new Republic("Германия", 83019200, "Франк-Вальтер Штайнмайер") as Country);

stopWatch.Stop();

Console.WriteLine("Для поиска последнего элемента понадобилось: " + stopWatch.Elapsed);

stopWatch = new System.Diagnostics.Stopwatch();

stopWatch.Start();

countries.Contains(new Republic("Гер", 83019200, "Франк-Вальтер Штайнмайер") as Country);

stopWatch.Stop();

Console.WriteLine("Для поиска элемента, которого нет, понадобилось: " + stopWatch.Elapsed);

}

public void ContainTime2()

{

Console.WriteLine("[Ищем элементы в countriesS]");

System.Diagnostics.Stopwatch stopWatch = new System.Diagnostics.Stopwatch();

stopWatch.Start();

countriesS.Contains((new Republic("Афганистан", 31500000, "Ашраф Гани") as Country).ToString());

stopWatch.Stop();

Console.WriteLine("Для поиска первого элемента понадобилось: " + stopWatch.Elapsed);

stopWatch = new System.Diagnostics.Stopwatch();

stopWatch.Start();

countriesS.Contains((new Republic("Алжир", 38087012, "Абдельмаджид Теббун") as Country).ToString());

stopWatch.Stop();

Console.WriteLine("Для поиска среднего элемента понадобилось: " + stopWatch.Elapsed);

stopWatch = new System.Diagnostics.Stopwatch();

stopWatch.Start();

countriesS.Contains((new Republic("Германия", 83019200, "Франк-Вальтер Штайнмайер") as Country).ToString());

stopWatch.Stop();

Console.WriteLine("Для поиска последнего элемента понадобилось: " + stopWatch.Elapsed);

stopWatch = new System.Diagnostics.Stopwatch();

stopWatch.Start();

countriesS.Contains((new Republic("Гер", 83019200, "Франк-Вальтер Штайнмайер") as Country).ToString());

stopWatch.Stop();

Console.WriteLine("Для поиска элемента, которого нет, понадобилось: " + stopWatch.Elapsed);

}

public void ContainTime3()

{

Console.WriteLine("[Ищем элементы в republics по ключу]");

System.Diagnostics.Stopwatch stopWatch = new System.Diagnostics.Stopwatch();

stopWatch.Start();

republics.ContainsKey(new Republic("Афганистан", 31500000, "Ашраф Гани") as Country);

stopWatch.Stop();

Console.WriteLine("Для поиска первого элемента понадобилось: " + stopWatch.Elapsed);

stopWatch = new System.Diagnostics.Stopwatch();

stopWatch.Start();

republics.ContainsKey(new Republic("Алжир", 38087012, "Абдельмаджид Теббун") as Country);

stopWatch.Stop();

Console.WriteLine("Для поиска среднего элемента понадобилось: " + stopWatch.Elapsed);

stopWatch = new System.Diagnostics.Stopwatch();

stopWatch.Start();

republics.ContainsKey(new Republic("Германия", 83019200, "Франк-Вальтер Штайнмайер") as Country);

stopWatch.Stop();

Console.WriteLine("Для поиска последнего элемента понадобилось: " + stopWatch.Elapsed);

stopWatch = new System.Diagnostics.Stopwatch();

stopWatch.Start();

republics.ContainsKey(new Republic("Гер", 83019200, "Франк-Вальтер Штайнмайер") as Country);

stopWatch.Stop();

Console.WriteLine("Для поиска элемента, которого нет, понадобилось: " + stopWatch.Elapsed);

}

public void ContainTime4()

{

Console.WriteLine("[Ищем элементы в republics по элементу]");

System.Diagnostics.Stopwatch stopWatch = new System.Diagnostics.Stopwatch();

stopWatch.Start();

republics.ContainsValue(new Republic("Афганистан", 31500000, "Ашраф Гани"));

stopWatch.Stop();

Console.WriteLine("Для поиска первого элемента понадобилось: " + stopWatch.Elapsed);

stopWatch = new System.Diagnostics.Stopwatch();

stopWatch.Start();

republics.ContainsValue(new Republic("Алжир", 38087012, "Абдельмаджид Теббун"));

stopWatch.Stop();

Console.WriteLine("Для поиска среднего элемента понадобилось: " + stopWatch.Elapsed);

stopWatch = new System.Diagnostics.Stopwatch();

stopWatch.Start();

republics.ContainsValue(new Republic("Германия", 83019200, "Франк-Вальтер Штайнмайер"));

stopWatch.Stop();

Console.WriteLine("Для поиска последнего элемента понадобилось: " + stopWatch.Elapsed);

stopWatch = new System.Diagnostics.Stopwatch();

stopWatch.Start();

republics.ContainsValue(new Republic("Гер", 83019200, "Франк-Вальтер Штайнмайер"));

stopWatch.Stop();

Console.WriteLine("Для поиска элемента, которого нет, понадобилось: " + stopWatch.Elapsed);

}

public void ContainTime5()

{

Console.WriteLine("[Ищем элементы в republicsS по ключу]");

System.Diagnostics.Stopwatch stopWatch = new System.Diagnostics.Stopwatch();

stopWatch.Start();

republicsS.ContainsKey((new Republic("Афганистан", 31500000, "Ашраф Гани") as Country).ToString());

stopWatch.Stop();

Console.WriteLine("Для поиска первого элемента понадобилось: " + stopWatch.Elapsed);

stopWatch = new System.Diagnostics.Stopwatch();

stopWatch.Start();

republicsS.ContainsKey((new Republic("Алжир", 38087012, "Абдельмаджид Теббун") as Country).ToString());

stopWatch.Stop();

Console.WriteLine("Для поиска среднего элемента понадобилось: " + stopWatch.Elapsed);

stopWatch = new System.Diagnostics.Stopwatch();

stopWatch.Start();

republicsS.ContainsKey((new Republic("Германия", 83019200, "Франк-Вальтер Штайнмайер") as Country).ToString());

stopWatch.Stop();

Console.WriteLine("Для поиска последнего элемента понадобилось: " + stopWatch.Elapsed);

stopWatch = new System.Diagnostics.Stopwatch();

stopWatch.Start();

republicsS.ContainsKey((new Republic("Гер", 83019200, "Франк-Вальтер Штайнмайер") as Country).ToString());

stopWatch.Stop();

Console.WriteLine("Для поиска элемента, которого нет, понадобилось: " + stopWatch.Elapsed);

}

public void ContainTime6()

{

Console.WriteLine("[Ищем элементы в republicsS по элементу]");

System.Diagnostics.Stopwatch stopWatch = new System.Diagnostics.Stopwatch();

stopWatch.Start();

republicsS.ContainsValue(new Republic("Афганистан", 31500000, "Ашраф Гани"));

stopWatch.Stop();

Console.WriteLine("Для поиска первого элемента понадобилось: " + stopWatch.Elapsed);

stopWatch = new System.Diagnostics.Stopwatch();

stopWatch.Start();

republicsS.ContainsValue(new Republic("Алжир", 38087012, "Абдельмаджид Теббун"));

stopWatch.Stop();

Console.WriteLine("Для поиска среднего элемента понадобилось: " + stopWatch.Elapsed);

stopWatch = new System.Diagnostics.Stopwatch();

stopWatch.Start();

republicsS.ContainsValue(new Republic("Германия", 83019200, "Франк-Вальтер Штайнмайер"));

stopWatch.Stop();

Console.WriteLine("Для поиска последнего элемента понадобилось: " + stopWatch.Elapsed);

stopWatch = new System.Diagnostics.Stopwatch();

stopWatch.Start();

republicsS.ContainsValue(new Republic("Гер", 83019200, "Франк-Вальтер Штайнмайер"));

stopWatch.Stop();

Console.WriteLine("Для поиска элемента, которого нет, понадобилось: " + stopWatch.Elapsed);

}

}

}