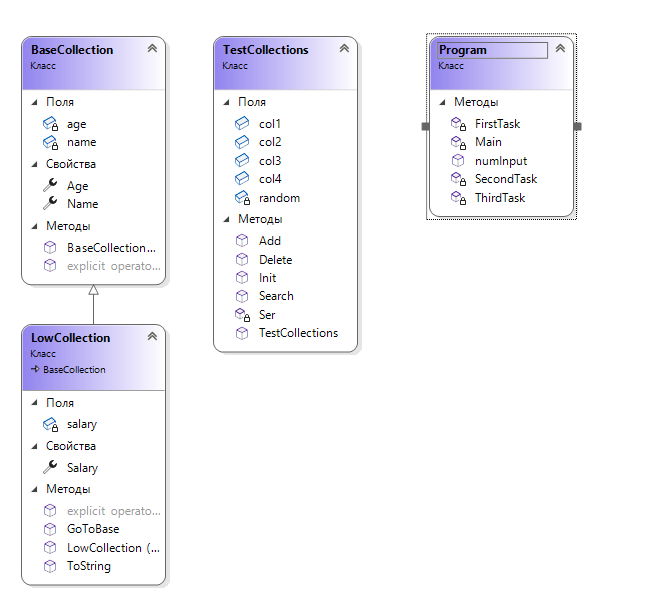
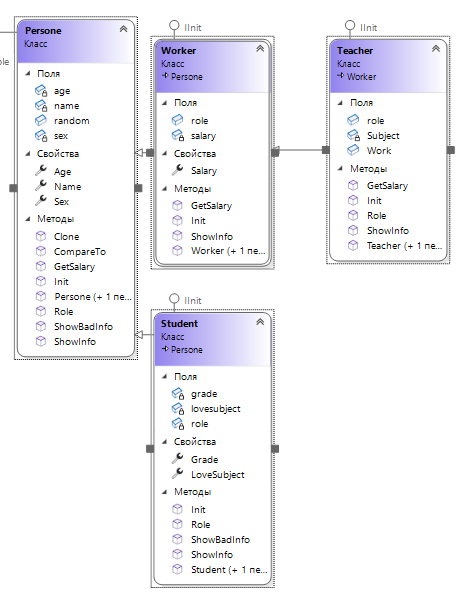
|  |
| --- |
| Пермский филиал федерального государственного автономного образовательного учреждения высшего образования  «Национальный исследовательский университет  «Высшая школа экономики»  *Факультет экономики, менеджмента и бизнес-информатики* |
|  |
| Полежаев Яков Вячеславович  **ЛАБОРАТОРНАЯ РАБОТА №11**  по направлению подготовки *38.03.05 Бизнес-информатика*  образовательная программа «Бизнес-информатика»   |  |  | | --- | --- | |  | Руководитель  Преподаватель кафедры ИТБ  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Марквирер В.Д. |   Пермь, 2022 |

# **Задача № 1 и 2**

## **Диаграмма классов**



**Рисунок 1 Диаграмма классов**



**Рисунок 2 Иерархия классов для подключаемой библиотеки**

## **Описание методов для работы с коллекцией.**

## **Задание 1.**

* 

Конструктор для создания необобщенной очереди (в данной программе без параметров). Однако, может принимать такие параметры как:

int Capacity (размер выделяемой памяти)

Float GrowFactor(на сколько умножается расширяемая коллекция)

ICollection c (и создана была коллекция по образу и подобию другой коллекции).

* 

Данный метод позволяет добавлять в коллекцию элементы в КОНЕЦ очереди и принимает в себя один единственный параметр типа Object.

* 

Данный метод позволяет удалять объект из начала очереди и возвращать его. После него сразу выполняется метод GetType() для получения типа возвращаемого объекта.

* 

Данный метод используется для проверки очереди на пустоту, чтобы не вызывалось исключение при Dequeu(). Он возвращает текущие количество элементов в очереди.

* 

Данный метод используется для копирования очереди в массив и последующей сортировки этого массива.



* 

После этого идет обратная запись уже упорядоченных элементов в очищенную очередь

* 

Данный метод позволяет определять есть ли объект в очереди. Он возвращает значение типа bool равное true, если объект есть в очереди, и false, если он не был найден.

* Для работы с иерархией была подключена библиотека при помощи dll из 10 лабораторной
* Все меню основано на кейсах switch и цикле while(flag), где flag принимает значение false лишь при кейсе 0.
* Для перебора коллекций используется цикл foreach.
* Была подключена библиотека System.Generic

## **Задание 2.**

## 

Конструктор для строго типизированной коллекции Dictionary или словарь, где в данной программе типом выступает Persone. Здесь в качестве ключа Tkey словарь принимает значение типа string, а в качестве Tvalue выступает обобщенный лист. Внутри фигурных скобок проходит инициализация словаря и в тоже время листа. Каждому ключу присваивается конкретное значение. Ключи не могут повторяться.

* 

Данный метод позволяет проверять словарь на пустоту возвращая true, если есть какие-то значение и false, если пустой. Может принимать функцию для проверки соответствия каждого элемента условию.

* 

В квадратных скобках указан ключ, по которому найдем список, в котором уже будем искать объект для удаления. Метод Find ()с лямбда выражением позволяет найти объект удовлетворяющий условиям и возвращает его первое вхождение.

* 

Метод Remove () принимает в себя единственный параметр типа Object, который является объектом для удаления.

* 

KeyValuePair используется при итерации словаря и позволяет получать или задавать пару Ключ-Значение.

* 

Здесь метод Add относится к коллекции List, но он может использоваться и для словаря. Для этого в параметры надо передать параметры Ключ – значение.

## **Пример демонстрационной программы**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace лаб\_11\_2\_

{

public class BaseCollection

{

int age;

string name;

public int Age { get =>this.age;

set

{

if(value >= 0)

{

this.age = value;

}

else{ this.age = 0;Console.WriteLine("Была ошибка при вводе возрасте. age = 0"); }

}

}

public string Name { get { return this.name; } set { this.name = value; } }

public BaseCollection()

{

age = 0;

name = "No name";

}

public BaseCollection(string name,int age)

{

this.Name = name;

this.Age = age;

}

public static explicit operator string(BaseCollection basa) { return basa.Name; }

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace лаб\_11\_2\_

{

internal class LowCollection: BaseCollection

{

int salary;

public int Salary { get => this.salary;

set

{

if(value < 0)

{

this.salary = 0;

Console.WriteLine("Ошибка при вводе зп");

}

else

{

this.salary = value;

}

}

}

public LowCollection(): base()

{

Salary = 0;

}

public LowCollection(string name,int age,int salary) : base(name, age)

{

this.Salary = salary;

}

public BaseCollection GoToBase()

{

return new BaseCollection(this.Name, this.Age);

}

public override string ToString()

{

return this.Name;

}

public static explicit operator string(LowCollection low) { return low.Name; }

}

}

**(метод Main будет продемонстрирован в конце)**

# **Задача 3**

## **Диаграмма классов (см. выше)**

## **Пример демонстрационной программы**

namespace лаб\_11\_2\_

{

internal class TestCollections

{

Random random = new Random();

public static List<LowCollection> col1;

public static List<string> col2;

public static Dictionary<BaseCollection, LowCollection> col3;

public static Dictionary<string, LowCollection> col4;

public TestCollections()

{

col1 = new List<LowCollection>(1000);

col2 = new List<string>(1000);

col3 = new Dictionary<BaseCollection, LowCollection>(1000);

col4 = new Dictionary<string, LowCollection>(1000);

}

public void Init()

{

for (int item = 0; item < 1000;item++)

{

LowCollection low = new LowCollection(random.Next(0,2000).ToString(), random.Next(0,100),random.Next(0,10000));

col1.Add(low);

col2.Add((string)low);

col3.Add(low,low);

int i = 0;

bool flag = false;

do

{

try

{

col4.Add((string)low, low);

flag = true;

}

catch (Exception) { i++; low.Name = low.Name + i; flag = false; }

}while(!flag);

}

}

public void Add()

{

Console.WriteLine("Куда добавим? 1) col1 2) col2 3) col3 4) col4");

int answer = Program.numInput();

bool flag = false;

do

{

switch (answer)

{

case 1:

col1.Add(new LowCollection());

flag = true;

break;

case 2:

col2.Add((string)new LowCollection());

flag = true;

break;

case 3:

col3.Add(new BaseCollection(),new LowCollection());

flag = true;

break;

case 4:

col4.Add((string)new BaseCollection(), new LowCollection());

flag = true;

break;

default:

{ flag = false;

Console.ForegroundColor = ConsoleColor.Red;

Console.WriteLine("Введено что-то не то");

Console.ForegroundColor = ConsoleColor.White;

break;

}

}

} while (!flag);

}

public void Delete()

{

Console.WriteLine("Где удалим? 1) col1 2) col2 3) col3 4) col4");

int answer = Program.numInput();

bool flag = false;

do

{

switch (answer)

{

case 1:

col1.Remove(new LowCollection());

flag = true;

break;

case 2:

col2.Remove((string)new LowCollection());

flag = true;

break;

case 3:

col3.Remove(new BaseCollection());

flag = true;

break;

case 4:

col4.Remove((string)new BaseCollection());

flag = true;

break;

default:

{

flag = false;

Console.ForegroundColor = ConsoleColor.Red;

Console.WriteLine("Введено что-то не то");

Console.ForegroundColor = ConsoleColor.White;

break;

}

}

} while (!flag);

}

private void Ser(LowCollection start)

{

Stopwatch stopwatch = new Stopwatch();

Console.Write("col1: ");

stopwatch.Start();

col1.Contains(start);

stopwatch.Stop();

Console.WriteLine(stopwatch.ElapsedTicks);

Console.Write("col2: ");

stopwatch.Restart();

col2.Contains((string)start);

stopwatch.Stop();

Console.WriteLine(stopwatch.ElapsedTicks);

Console.Write("col3: ");

stopwatch.Restart();

col3.ContainsKey(start);

stopwatch.Stop();

Console.Write("по ключу "+stopwatch.ElapsedTicks + ", ");

stopwatch.Restart();

col3.ContainsValue(start);

stopwatch.Stop();

Console.WriteLine("по значению "+stopwatch.ElapsedTicks);

Console.Write("col4: ");

stopwatch.Restart();

col4.ContainsKey((string)start);

stopwatch.Stop();

Console.Write("по ключу "+stopwatch.ElapsedTicks + ", ");

stopwatch.Restart();

col4.ContainsValue(start);

stopwatch.Stop();

Console.WriteLine("по значению "+stopwatch.ElapsedTicks);

Console.WriteLine(new String('-', 10));

}

public void Search()

{

LowCollection start = new LowCollection("Sam",27,1000);

LowCollection middle = new LowCollection("Tom",30,2234);

LowCollection end = new LowCollection("Manson",12,0);

LowCollection no = new LowCollection("Greg",19,500);

col1[0] = start;

col2[0] = (string)start;

col1[500] = middle;

col2[500] = (string)middle;

col1[999] = end;

col2[999] = (string)end;

col3[start] = start;

col3[middle] = middle;

col3[end] = end;

col4[(string)start] = start;

col4[(string)middle] = middle;

col4[(string)end] = end;

Console.WriteLine("Ищем первый элемент");

Ser(start);

Console.WriteLine("Ищем средний элемент");

Ser(middle);

Console.WriteLine("Ищем последний элемент");

Ser(end);

Console.WriteLine("Ищем не существующий элемент");

Ser(no);

Console.ReadKey();

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using PeopleLib;

using System.Collections;

namespace лаб\_11\_2\_

{

internal class Program

{

public static int numInput()

{

do

{

try

{

int num;

num = Convert.ToInt32(Console.ReadLine());

return num;

}

catch (FormatException)

{

Console.ForegroundColor = ConsoleColor.Red;

Console.WriteLine("Введено не число");

Console.ForegroundColor = ConsoleColor.White;

}

} while (true);

}

static void FirstTask()

{

bool flag = false;

int answer = 0;

int who = 0;

string Menu = "1. Удаление объектов\n" +

"2. Добавление объектов\n" +

"3. Подсчет объектов\n" +

"4. Вывод элементов\n" +

"5. Средний возраст\n" +

"6. вывод массива\n" +

"7. Создать клона\n" +

"8. сортировка и поиск\n" +

"0. Перейти к селедующему заданию\n";

Queue peoples = new Queue();

peoples.Enqueue(new Persone("Tom", 27, "м"));

peoples.Enqueue(new Student("Mark", 17, "м", 5, "Math"));

peoples.Enqueue(new Worker("Linda", 20, "ж", 30000));

peoples.Enqueue(new Teacher("Olga", 30, "ж", 20000, "English"));

do

{

Console.ReadKey();

Console.Clear();

Console.WriteLine("Задание 1) меню :");

Console.WriteLine(Menu);

answer = Program.numInput();

Console.Clear();

switch (answer)

{

case 1:

{

Console.WriteLine("Выбран 1. удаление объектов");

if (peoples.Count != 0)

{

Console.WriteLine();

Console.WriteLine($"удаляем {peoples.Dequeue().GetType()}");

Console.WriteLine("новый список: ");

foreach (Persone item in peoples)

{

Console.WriteLine(new string('-', 10));

try

{

Console.WriteLine(item.GetType());

}catch (ArgumentNullException) { Console.WriteLine("пусто"); }

}

}

flag = true;

break;

}

case 2:

Console.WriteLine("Выбран 2. создание объектов");

string Name, Sex, LoveSubject, Subject;

int Age, Grade, Salary;

Console.WriteLine();

flag = false;

do

{

Console.WriteLine("Кого будем создавать? 1)Человек,2)Работник,3)студент,4)учитель)");

who = Program.numInput();

Console.Clear();

switch (who)

{

case 1:

Console.WriteLine("Введите имя");

Name = Console.ReadLine();

Console.WriteLine("Введите возраст");

Age = Program.numInput();

Console.WriteLine("Введите пол ");

do

{

Sex = Console.ReadLine();

if (Sex == "м" || Sex == "ж")

{

break;

}

else

{

Console.ForegroundColor = ConsoleColor.Red;

Console.WriteLine("Ошибка ввода");

Console.ForegroundColor = ConsoleColor.White;

}

} while (true);

peoples.Enqueue(new Persone(Name, Age, Sex));

Console.WriteLine();

Console.WriteLine("новый список: ");

foreach (Persone item in peoples)

{

Console.WriteLine(new string('-', 10));

Console.WriteLine(item.GetType());

}

flag = true;

break;

case 2:

Console.WriteLine("Введите имя");

Name = Console.ReadLine();

Console.WriteLine("Введите возраст");

Age = Program.numInput();

Console.WriteLine("Введите пол ");

do

{

Sex = Console.ReadLine();

if (Sex == "м" || Sex == "ж")

{

break;

}

else

{

Console.ForegroundColor = ConsoleColor.Red;

Console.WriteLine("Ошибка ввода");

Console.ForegroundColor = ConsoleColor.White;

}

} while (true);

Console.WriteLine("Введите з.п");

Salary = Program.numInput();

peoples.Enqueue(new Worker(Name, Age, Sex, Salary));

Console.WriteLine();

Console.WriteLine("новый список: ");

foreach (Persone item in peoples)

{

Console.WriteLine(new string('-', 10));

Console.WriteLine(item.GetType());

}

flag = true;

break;

case 3:

Console.WriteLine("Введите имя");

Name = Console.ReadLine();

Console.WriteLine("Введите возраст");

Age = Program.numInput();

Console.WriteLine("Введите пол ");

do

{

Sex = Console.ReadLine();

if (Sex == "м" || Sex == "ж")

{

break;

}

else

{

Console.ForegroundColor = ConsoleColor.Red;

Console.WriteLine("Ошибка ввода");

Console.ForegroundColor = ConsoleColor.White;

}

} while (true);

Console.WriteLine("Введите класс ученика");

Grade = Program.numInput();

Console.WriteLine("Введите любимый предмет");

LoveSubject = Console.ReadLine();

peoples.Enqueue(new Student(Name, Age, Sex, Grade, LoveSubject));

Console.WriteLine();

Console.WriteLine("новый список: ");

foreach (Persone item in peoples)

{

Console.WriteLine(new string('-', 10));

Console.WriteLine(item.GetType());

}

flag = true;

break;

case 4:

Console.WriteLine("Введите имя");

Name = Console.ReadLine();

Console.WriteLine("Введите возраст");

Age = Program.numInput();

Console.WriteLine("Введите пол ");

do

{

Sex = Console.ReadLine();

if (Sex == "м" || Sex == "ж")

{

break;

}

else

{

Console.ForegroundColor = ConsoleColor.Red;

Console.WriteLine("Ошибка ввода");

Console.ForegroundColor = ConsoleColor.White;

}

} while (true);

Console.WriteLine("Введите з.п");

Salary = Program.numInput();

Console.WriteLine("Введите предмет преподавания");

Subject = Console.ReadLine();

peoples.Enqueue(new Teacher(Name, Age, Sex, Salary, Subject));

Console.WriteLine();

Console.WriteLine("новый список: ");

foreach (Persone item in peoples)

{

Console.WriteLine(new string('-', 10));

Console.WriteLine(item.GetType());

}

flag = true;

break;

default:

Console.ForegroundColor = ConsoleColor.Red;

Console.WriteLine("Введено что-то не то");

Console.ForegroundColor = ConsoleColor.White;

flag = false;

break;

}

} while (!flag);

break;

case 3:

flag = false;

Console.WriteLine("Вы выбрали 3.Подсчет объектов");

do

{

Console.WriteLine("Кого будем искать? 1)Человек,2)Работник,3)студент,4)учитель)");

who = Program.numInput();

int count = 0;

switch (who)

{

case 1:

foreach (Object item in peoples)

{

if (item is Persone)

{

count++;

}

}

Console.WriteLine($"В очереди всего {count}людей");

flag = true;

break;

case 2:

foreach (Object item in peoples)

{

if (item is Worker)

{

count++;

}

}

Console.WriteLine($"В очереди всего {count}работников");

flag = true;

break;

case 3:

foreach (Object item in peoples)

{

if (item is Student)

{

count++;

}

}

Console.WriteLine($"В очереди всего {count}учеников");

flag = true;

break;

case 4:

foreach (Object item in peoples)

{

if (item is Teacher)

{

count++;

}

}

Console.WriteLine($"В очереди всего {count}учителей");

flag = true;

break;

default:

Console.ForegroundColor = ConsoleColor.Red;

Console.WriteLine("Введено что-то не то");

Console.ForegroundColor = ConsoleColor.White;

flag = false;

break;

}

} while (!flag);

break;

case 4:

Console.WriteLine("Вы выбрали 4.Вывод элементов");

do

{

Console.WriteLine("Кого будем выводить? 1)Человек,2)Работник,3)студент,4)учитель)");

who = Program.numInput();

switch (who)

{

case 1:

foreach (Object item in peoples)

{

if (item is Persone)

{

Persone n = item as Persone;

n.ShowInfo();

Console.WriteLine(new string('-', 10));

}

}

flag = true;

break;

case 2:

foreach (Object item in peoples)

{

if (item is Worker)

{

Worker n = item as Worker;

n.ShowInfo();

Console.WriteLine(new string('-', 10));

}

}

flag = true;

break;

case 3:

foreach (Object item in peoples)

{

if (item is Student)

{

Student n = item as Student;

n.ShowInfo();

Console.WriteLine(new string('-', 10));

}

}

flag = true;

break;

case 4:

foreach (Object item in peoples)

{

if (item is Teacher)

{

Teacher n = item as Teacher;

n.ShowInfo();

Console.WriteLine(new string('-', 10));

}

}

flag = true;

break;

default:

Console.ForegroundColor = ConsoleColor.Red;

Console.WriteLine("Введено что-то не то");

Console.ForegroundColor = ConsoleColor.White;

flag = false;

break;

}

} while (!flag);

break;

case 5:

Console.WriteLine("Вы выбрали 5. Средний возраст");

do

{

Console.WriteLine("У кого будем считать? 1)Человек,2)Работник,3)студент,4)учитель)");

who = Program.numInput();

int count = 0;

int Summ = 0;

switch (who)

{

case 1:

foreach (Object item in peoples)

{

if (item is Persone)

{

Persone n = item as Persone;

Summ += n.Age;

count++;

}

}

Console.WriteLine($"средний возраст людей {Summ / count}");

flag = true;

break;

case 2:

foreach (Object item in peoples)

{

if (item is Worker)

{

Worker n = item as Worker;

Summ += n.Age;

count++;

}

}

Console.WriteLine($"средний возраст работающих {Summ / count}");

flag = true;

break;

case 3:

foreach (Object item in peoples)

{

if (item is Student)

{

Student n = item as Student;

Summ += n.Age;

count++;

}

}

Console.WriteLine($"средний возраст учеников {Summ / count}");

flag = true;

break;

case 4:

foreach (Object item in peoples)

{

if (item is Teacher)

{

Teacher n = item as Teacher;

Summ += n.Age;

count++;

}

}

Console.WriteLine($"средний возраст учителей {Summ / count}");

flag = true;

break;

default:

Console.ForegroundColor = ConsoleColor.Red;

Console.WriteLine("Введено что-то не то");

Console.ForegroundColor = ConsoleColor.White;

flag = false;

break;

}

} while (!flag);

break;

case 6:

Console.WriteLine("вы выбрали 6. вывод массива");

foreach (Object item in peoples)

{

Console.WriteLine(item.GetType());

}

flag = true;

break;

case 7:

Console.WriteLine("Создать клона");

var clone = peoples;

if (peoples.Count > 0)

{

Console.WriteLine("peoples");

foreach (Object item in peoples)

{

Console.WriteLine(item.GetType());

}

Console.WriteLine(new String('-', 10));

Console.WriteLine("clone");

foreach (Object item in clone)

{

Console.WriteLine(item.GetType());

}

Console.WriteLine(new String('-', 10));

peoples.Dequeue();

Console.WriteLine("Delete in peoples");

Console.WriteLine(new String('-', 10));

Console.WriteLine("peoples");

foreach (Object item in peoples)

{

Console.WriteLine(item.GetType());

}

Console.WriteLine(new String('-', 10));

Console.WriteLine("clone");

foreach (Object item in clone)

{

Console.WriteLine(item.GetType());

}

}

else { Console.WriteLine("Опыт с клонирование не возможен. Коллекция пустая"); }

flag = true;

break;

case 8:

Console.WriteLine("вы выбрали 8. сортировка и поиск");

Persone find = new Persone("Smit", 27, "М");

Persone clon= find as Persone;

peoples.Enqueue(find);

var arr = peoples.ToArray();

Array.Sort(arr);

peoples.Clear();

foreach (Object item in arr) { peoples.Enqueue(item); }

bool v = peoples.Contains(clon);

if (v == true)

{

Console.WriteLine("да, есть рабочий");

clon.ShowInfo();

}

else { Console.WriteLine("такого человека нет"); }

break;

case 0:

flag = false;

break;

default:

{

Console.ForegroundColor = ConsoleColor.Red;

Console.WriteLine("Введено что-то не то");

Console.ForegroundColor = ConsoleColor.White;

flag = true;

break;

}

}

} while (flag);

}

static void SecondTask()//второе задание!

{

string Role;

bool flag = false;

int answer = 0;

int who = 0;

string Menu = "1. Удаление объектов\n" +

"2. Добавление объектов\n" +

"3. Подсчет объектов\n" +

"4. Вывод элементов\n" +

"5. Средний возраст\n" +

"6. вывод массива\n" +

"7. Создать клона\n" +

"8. сортировка и поиск\n" +

"0. перейти к следующему заданию\n";

Dictionary<string, List<Persone>> peoples = new Dictionary<string, List<Persone>>()

{

["JustPersone"] = new List<Persone> {new Persone("Tom", 27, "м") },

["Worker"] = new List<Persone> { new Worker("Linda", 20, "ж", 30000) },

["Student"] = new List<Persone> { new Student("Mark", 17, "м", 5, "Math") },

["Teacher"] = new List<Persone> { new Teacher("Olga", 30, "ж", 20000, "English") }

};

do

{

Console.ReadKey();

Console.Clear();

Console.WriteLine("Задание 2) меню :");

Console.WriteLine(Menu);

answer = Program.numInput();

Console.Clear();

string Name, Sex, LoveSubject, Subject;

switch (answer)

{

case 1:

{

Console.WriteLine("Выбран 1. удаление объектов");

if (peoples.Any() == true)

{

Console.WriteLine();

Console.WriteLine("Кого удаляем?");

Console.WriteLine("Напишите роль(JustPersone,Worker,Student,Teacher)");

Role = "";

do

{

flag = false;

Role = Console.ReadLine();

if (Role == "JustPersone" || Role == "Worker" || Role == "Student" || Role == "Teacher")

{

flag = true;

}

else { Console.WriteLine("такой роли нет"); }

} while (!flag);

Console.WriteLine("Напишите имя");

Name = Console.ReadLine();

var s = peoples[$"{Role}"].Find(item => item.Name == $"{Name}");

if(s != null)

{

Console.Write("Удаляем:");

s.ShowInfo();

peoples[$"{Role}"].Remove(s);

}

else

{

Console.ForegroundColor = ConsoleColor.Red;

Console.WriteLine("Такого человека нет");

Console.ForegroundColor = ConsoleColor.White;

}

Console.WriteLine("новый список: ");

foreach(KeyValuePair<string,List<Persone>> item in peoples)

{

foreach(Persone item2 in item.Value)

try

{

s = item2 as Persone;

s.ShowInfo();

Console.WriteLine(new string('-', 10));

}

catch (NullReferenceException) { Console.WriteLine("пусто"); }

}

}

flag = true;

break;

}

case 2:

Console.WriteLine("Выбран 2. создание объектов");

int Age, Grade, Salary;

Console.WriteLine();

flag = false;

do

{

Console.WriteLine("Кого будем создавать? 1)Человек,2)Работник,3)студент,4)учитель)");

who = Program.numInput();

Console.Clear();

switch (who)

{

case 1:

Console.WriteLine("Введите имя");

Name = Console.ReadLine();

Console.WriteLine("Введите возраст");

Age = Program.numInput();

Console.WriteLine("Введите пол ");

do

{

Sex = Console.ReadLine();

if (Sex == "м" || Sex == "ж")

{

break;

}

else

{

Console.ForegroundColor = ConsoleColor.Red;

Console.WriteLine("Ошибка ввода");

Console.ForegroundColor = ConsoleColor.White;

}

} while (true);

peoples["JustPersone"].Add(new Persone(Name, Age, Sex));

Console.WriteLine();

Console.WriteLine("новый список: ");

foreach (KeyValuePair<string, List<Persone>> item in peoples)

{

foreach (Persone item2 in item.Value)

{

item2.ShowInfo();

Console.WriteLine(new string('-', 10));

}

}

flag = true;

break;

case 2:

Console.WriteLine("Введите имя");

Name = Console.ReadLine();

Console.WriteLine("Введите возраст");

Age = Program.numInput();

Console.WriteLine("Введите пол ");

do

{

Sex = Console.ReadLine();

if (Sex == "м" || Sex == "ж")

{

break;

}

else

{

Console.ForegroundColor = ConsoleColor.Red;

Console.WriteLine("Ошибка ввода");

Console.ForegroundColor = ConsoleColor.White;

}

} while (true);

Console.WriteLine("Введите з.п");

Salary = Program.numInput();

peoples["Worker"].Add(new Worker(Name, Age, Sex, Salary));

Console.WriteLine();

Console.WriteLine("новый список: ");

foreach (KeyValuePair<string, List<Persone>> item in peoples)

{

foreach (Persone item2 in item.Value)

{

item2.ShowInfo();

Console.WriteLine(new string('-', 10));

}

}

flag = true;

break;

case 3:

Console.WriteLine("Введите имя");

Name = Console.ReadLine();

Console.WriteLine("Введите возраст");

Age = Program.numInput();

Console.WriteLine("Введите пол ");

do

{

Sex = Console.ReadLine();

if (Sex == "м" || Sex == "ж")

{

break;

}

else

{

Console.ForegroundColor = ConsoleColor.Red;

Console.WriteLine("Ошибка ввода");

Console.ForegroundColor = ConsoleColor.White;

}

} while (true);

Console.WriteLine("Введите класс ученика");

Grade = Program.numInput();

Console.WriteLine("Введите любимый предмет");

LoveSubject = Console.ReadLine();

peoples["Student"].Add(new Student(Name, Age, Sex, Grade, LoveSubject));

Console.WriteLine();

Console.WriteLine("новый список: ");

foreach (KeyValuePair<string, List<Persone>> item in peoples)

{

foreach (Persone item2 in item.Value)

{

item2.ShowInfo();

Console.WriteLine(new string('-', 10));

}

}

flag = true;

break;

case 4:

Console.WriteLine("Введите имя");

Name = Console.ReadLine();

Console.WriteLine("Введите возраст");

Age = Program.numInput();

Console.WriteLine("Введите пол ");

do

{

Sex = Console.ReadLine();

if (Sex == "м" || Sex == "ж")

{

break;

}

else

{

Console.ForegroundColor = ConsoleColor.Red;

Console.WriteLine("Ошибка ввода");

Console.ForegroundColor = ConsoleColor.White;

}

} while (true);

Console.WriteLine("Введите з.п");

Salary = Program.numInput();

Console.WriteLine("Введите предмет преподавания");

Subject = Console.ReadLine();

peoples["Teacher"].Add(new Teacher(Name, Age, Sex, Salary, Subject));

Console.WriteLine();

Console.WriteLine("новый список: ");

foreach (KeyValuePair<string, List<Persone>> item in peoples)

{

foreach (Persone item2 in item.Value)

{

item2.ShowInfo();

Console.WriteLine(new string('-', 10));

}

}

flag = true;

break;

default:

Console.ForegroundColor = ConsoleColor.Red;

Console.WriteLine("Введено что-то не то");

Console.ForegroundColor = ConsoleColor.White;

flag = false;

break;

}

} while (!flag);

break;

case 3:

flag = false;

Console.WriteLine("Вы выбрали 3.Подсчет объектов");

do

{

Console.WriteLine("Кого будем искать? 1)Человек,2)Работник,3)студент,4)учитель)");

who = Program.numInput();

int count = 0;

switch (who)

{

case 1:

count = peoples["Persone"].Count();

Console.WriteLine($"В словаре всего {count}людей");

flag = true;

break;

case 2:

count = peoples["Worker"].Count();

Console.WriteLine($"В словаре всего {count}работников");

flag = true;

break;

case 3:

count = peoples["Student"].Count();

Console.WriteLine($"В словаре всего {count}учеников");

flag = true;

break;

case 4:

count = peoples["Teacher"].Count();

Console.WriteLine($"В словаре всего {count}учителей");

flag = true;

break;

default:

Console.ForegroundColor = ConsoleColor.Red;

Console.WriteLine("Введено что-то не то");

Console.ForegroundColor = ConsoleColor.White;

flag = false;

break;

}

} while (!flag);

break;

case 4:

Console.WriteLine("Вы выбрали 4.Вывод элементов");

do

{

Console.WriteLine("Кого будем выводить? 1)Человек,2)Работник,3)студент,4)учитель)");

who = Program.numInput();

switch (who)

{

case 1:

foreach (Object item in peoples["Persone"])

{

Persone n = item as Persone;

n.ShowInfo();

Console.WriteLine(new string('-', 10));

}

flag = true;

break;

case 2:

foreach (Object item in peoples["Worker"])

{

Worker n = item as Worker;

n.ShowInfo();

Console.WriteLine(new string('-', 10));

}

flag = true;

break;

case 3:

foreach (Object item in peoples["Student"])

{

Student n = item as Student;

n.ShowInfo();

Console.WriteLine(new string('-', 10));

}

flag = true;

break;

case 4:

foreach (Object item in peoples["Teacher"])

{

Teacher n = item as Teacher;

n.ShowInfo();

Console.WriteLine(new string('-', 10));

}

flag = true;

break;

default:

Console.ForegroundColor = ConsoleColor.Red;

Console.WriteLine("Введено что-то не то");

Console.ForegroundColor = ConsoleColor.White;

flag = false;

break;

}

} while (!flag);

break;

case 5:

Console.WriteLine("Вы выбрали 5. Средний возраст");

do

{

Console.WriteLine("У кого будем считать? 1)Человек,2)Работник,3)студент,4)учитель)");

who = Program.numInput();

int count = 0;

int Summ = 0;

switch (who)

{

case 1:

foreach (Object item in peoples["JustPersone"])

{

Persone n = item as Persone;

Summ += n.Age;

count++;

}

Console.WriteLine($"средний возраст простых людей {Summ / count}");

flag = true;

break;

case 2:

foreach (Object item in peoples["Worker"])

{

Worker n = item as Worker;

Summ += n.Age;

count++;

}

Console.WriteLine($"средний возраст работающих без инфы о мес.раб. {Summ / count}");

flag = true;

break;

case 3:

foreach (Object item in peoples["Student"])

{

Student n = item as Student;

Summ += n.Age;

count++;

}

Console.WriteLine($"средний возраст учеников {Summ / count}");

flag = true;

break;

case 4:

foreach (Object item in peoples["Teacher"])

{

Teacher n = item as Teacher;

Summ += n.Age;

count++;

}

Console.WriteLine($"средний возраст учителей {Summ / count}");

flag = true;

break;

default:

Console.ForegroundColor = ConsoleColor.Red;

Console.WriteLine("Введено что-то не то");

Console.ForegroundColor = ConsoleColor.White;

flag = false;

break;

}

} while (!flag);

break;

case 6:

Console.WriteLine("вы выбрали 6. вывод массива");

foreach (KeyValuePair<string, List<Persone>> item in peoples)

{

foreach (Persone item2 in item.Value)

{

item2.ShowInfo();

Console.WriteLine(new string('-', 10));

}

}

flag = true;

break;

case 7:

Console.WriteLine("Создать клона");

var clone = peoples;

if (peoples.Any())

{

peoples["Worker"].Add(new Worker("smile", 30, "м", 150000));

Console.WriteLine("peoples");

foreach (KeyValuePair <string,List<Persone>> item in peoples)

{

foreach(Object obj in item.Value)

{

Console.WriteLine(obj.GetType());

}

}

Console.WriteLine(new String('-', 10));

Console.WriteLine("clone");

foreach (KeyValuePair<string, List<Persone>> item in clone)

{

foreach (Object obj in item.Value)

{

Console.WriteLine(obj.GetType());

}

}

peoples["Worker"].Remove(peoples["Worker"].Find(item => item.Name == "smile"));

Console.WriteLine(new String('-', 10));

Console.WriteLine("Delete in peoples");

Console.WriteLine(new String('-', 10));

Console.WriteLine("peoples");

foreach (KeyValuePair<string, List<Persone>> item in peoples)

{

foreach (Object obj in item.Value)

{

Console.WriteLine(obj.GetType());

}

}

Console.WriteLine(new String('-', 10));

Console.WriteLine("clone");

foreach (KeyValuePair<string, List<Persone>> item in clone)

{

foreach (Object obj in item.Value)

{

Console.WriteLine(obj.GetType());

}

}

}

flag = true;

break;

case 8:

Console.WriteLine("вы выбрали 8. сортировка и поиск");

foreach(KeyValuePair<string,List<Persone>> item in peoples)

{

item.Value.Sort();

}

Console.WriteLine("Кого ищем?");

Console.WriteLine("Напишите роль(JustPersone,Worker,Student,Teacher)");

Role = "";

do

{

flag = false;

Role = Console.ReadLine();

if (Role == "JustPersone" || Role == "Worker" || Role == "Student" || Role == "Teacher")

{

flag = true;

}

} while (!flag);

Console.WriteLine("Напишите имя");

Name = Console.ReadLine();

var arr = peoples[Role].Find(item => item.Name == Name);

if(arr != null)

{

arr.ShowInfo();

}

else { Console.WriteLine("такого человека нет"); }

break;

case 0:

flag = false;

break;

default:

{

Console.ForegroundColor = ConsoleColor.Red;

Console.WriteLine("Введено что-то не то");

Console.ForegroundColor = ConsoleColor.White;

flag = true;

break;

}

}

} while (flag);

}

static void ThirdTask()//третье задание

{

TestCollections collections = new TestCollections();

collections.Init();

bool flag = false;

do

{

Console.Clear();

Console.WriteLine("1. удаление \n" +

"2. добавление\n" +

"3. поиск\n" +

"0. закончить\n");

int answer = Program.numInput();

switch (answer)

{

case 1:

Console.WriteLine("1. удаление");

collections.Delete();

flag = true;

break;

case 2:

Console.WriteLine("2. добавление");

collections.Add();

flag = true;

break;

case 3:

Console.WriteLine("3. поиск");

collections.Search();

flag = true;

break;

case 0:

flag = false;

break;

default:

Console.ForegroundColor = ConsoleColor.Red;

Console.WriteLine("Введено что-то не то");

Console.ForegroundColor = ConsoleColor.White;

break;

}

} while (flag);

}

static void Main(string[] args)

{

FirstTask();

SecondTask();

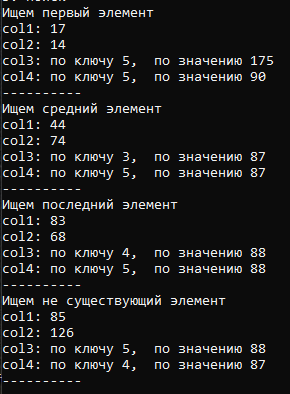
ThirdTask();

}

}

}

## **Полученные результаты.**



Где:

col1 = new List<LowCollection>(1000); (лист с объектами)

col2 = new List<string>(1000);(лист типа стринг)

col3 = new Dictionary<BaseCollection, LowCollection>(1000);(словарь с объектом базового класса в качестве Tkey и производным классом как TValue)

col4 = new Dictionary<string, LowCollection>(1000); (словарь с типом стринг в качестве Tkey и производным классом как TValue)

## **Анализ полученных результатов.**

В основном словари показывают наиболее быстрые результаты так как данный класс реализуется как отсортированная хэш-таблица, что ускоряет процесс поиска взятием хэш-кода каждого ключа. Хэш высчитывается только один раз и затем складывается в скрытое поле классе. Однако, словари упорядочены по ключам, а не значениям, следовательно, поиск по значению может уступать листу. Стоит добавить, что словарь проводит поиск линейно, а лист логарифмически.

Разницу между string и object в качестве к сожалению, мне не понятна, возможно все зависит от размера получаемых хэш-значений, или от процесса их получения.