# Отчет по практической работе №13 «Разработка программы, управляемой событиями»

## Задание

**Часть 1.**

1. Создать иерархию классов (см. лаб. 9). Для каждого класса реализовать конструктор без параметров, с параметрами, свойства для доступа к полям объектов, метод для автоматического формирования объектов. Перегрузить метод ToString() для формирования строки со значениями всех полей класса.
2. Создать класс MyCollection как производный класс от класса Сollection<MyClass>. Реализовать в классе методы для заполнения коллекции (элементы коллекции формируются автоматически), добавления элементов коллекции, удаления элементов коллекции, сортировки элементов коллекции по заданному полю, очистки коллекции. Реализовать итератор для доступа к элементам коллекции. Реализовать свойство Length (только для чтения), содержащее текущее количество элементов коллекции.
3. Написать демонстрационную программу иллюстрирующую работу всех методов класса MyCollection. Использовать исключительные ситуации для обработки ошибочных ситуаций.

**Часть 2.**

1. Определить класс MyNewCollection производный от класса MyCollection, который с помощью событий извещает об изменениях в коллекции.

* Коллекция состоит из объектов ссылочных типов. Коллекция изменяется при удалении/добавлении элементов или при изменении одной из входящих в коллекцию ссылок, например, когда одной из ссылок присваивается новое значение. В этом случае в соответствующих методах или свойствах класса бросаются события.
* При изменении данных объектов, ссылки на которые входят в коллекцию, значения самих ссылок не изменяются. Этот тип изменений не порождает событий.
* Для событий, извещающих об изменениях в коллекции, определяется свой делегат. События регистрируются в специальных классах-слушателях.

1. Для событий определить делегат CollectionHandler с сигнатурой:

void CollectionHandler (object source, CollectionHandlerEventArgs args);

1. Определить класс CollectionHandlerEventArgs, производный от класса System.EventArgs, который содержит

* открытое автореализуемое свойство типа string с названием коллекции, в которой произошло событие;
* открытое автореализуемое свойство типа string с информацией о типе изменений в коллекции;
* открытое автореализуемое свойство для ссылки на объект, с которым связаны изменения;
* конструкторы для инициализации класса;
* перегруженную версию метода string ToString() для формирования строки с информацией обо всех полях класса.

1. В новую версию класса MyNewCollection добавить

* открытое автореализуемое свойство типа string с названием коллекции;
* метод bool Remove (int j) для удаления элемента с номером j ; если в списке нет элемента с номером j, метод возвращает значение false;
* индексатор (с методами get и set) с целочисленным индексом для доступа к элементу с заданным номером.

1. В класс MyNewCollection добавить два события типа CollectionHandler.

* CollectionCountChanged, которое происходит при добавлении нового элемента в коллекцию или при удалении элемента из коллекции; через объект CollectionHandlerEventArgs cобытие передает имя коллекции, строку с информацией о том, что в коллекцию был добавлен новый элемент или из нее был удален элемент, ссылку на добавленный или удаленный элемент;
* CollectionReferenceChanged, которое происходит, когда одной из ссылок, входящих в коллекцию, присваивается новое значение; через объект CollectionHandlerEventArgs событие передает имя коллекции, строку с информацией о том, что был заменен элемент в коллекции, и ссылку на новый элемент.

1. Событие CollectionCountChanged бросают следующие методы класса MyNewCollection

* AddDefaults();
* Add (object[] ) ;
* Remove (int index).

1. Событие CollectionReferenceChanged бросает метод set индексатора, определенного в классе MyNewCollection.
2. Определить класс Journal, который можно использовать для накопления информации об изменениях в коллекциях типа MyNewCollection. Класс Journal хранит информацию в списке объектов типа JournalEntry. Каждый элемент списка содержит информацию об отдельном изменении, которое произошло в коллекции. Класс JournalEntry содержит

* открытое автореализуемое свойство типа string с названием коллекции, в которой произошло событие;
* открытое автореализуемое свойство типа string с информацией о типе изменений в коллекции;
* открытое автореализуемое свойство типа string c данными объекта, с которым связаны изменения в коллекции;
* конструктор для инициализации полей класса;
* перегруженную версию метода string ToString().

1. Класс Journal содержит

* коллекцию элементов типа JournalEntry (закрытое поле);
* обработчики событий CollectionCountChanged и CollectionReferenceChanged, которые добавляют новый элемент JournalEntry в коллекцию; для инициализации JournalEntry используется информация из объекта CollectionHandlerEventArgs, который передается вместе с событием;
* перегруженную версию метода string ToString() для формирования строки с информацией обо всех элементах массива.

1. Написать демонстрационную программу, в которой:

* создать две коллекции MyNewCollection.
* Создать два объекта типа Journal, один объект Journal подписать на события CollectionCountChanged и CollectionReferenceChanged из первой коллекции, другой объект Journal подписать на события CollectionReferenceChanged из обеих коллекций.

1. Внести изменения в коллекции MyNewCollection

* добавить элементы в коллекции;
* удалить некоторые элементы из коллекций;
* присвоить некоторым элементам коллекций новые значения.

1. Вывести данные обоих объектов Journal.

## 1. Анализ задачи

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **№** | **Описание функции** | **Входные данные** | **Классы входных данных** | | **Выходные данные** |
| **Основные функции** | | | | | |
| 1. | Добавление элемента в коллекцию | Коллекция класса Time, целый номер элемента в коллекции, объект класса «время» | | Номер элемента не превышает число элементов в коллекции + 1, и больше 0 | Коллекция класса Time с добавленным на указанное место элементом |
| Номер элемента превышает число элементов в коллекции + 1, или меньше или равен 0 | Сообщение об ошибке |
| 2. | Заполнение коллекции случайными элементами | Коллекция класса Time, целое число элементов | | Число элементов больше 0 | Коллекция класса Time, состоящая из введенного числа случайных элементов |
| Число элементов меньше или равно 0 | Сообщение об ошибке |
| 3. | Удаление элемента из коллекции | Коллекция класса Time, натуральный номер элемента в коллекции | | Элемент с таким значением номера существует | Коллекция класса Time с удаленным с указанного места элементом |
| Элемент с таким значением номера не существует | Сообщение об ошибке |
| 4. | Сортировка коллекции по числу часов | Коллекция класса Time | |  | Отсортированная коллекция класса Time |
| 5. | Сортировка коллекции по числу минут | Коллекция класса Time | |  | Отсортированная коллекция класса Time |
| 6. | Вывод коллекции на экран | Коллекция класса Time | |  | Выведенная на экран коллекция класса Time |
| 7. | Очистка коллекции | Коллекция класса Time | |  | Пустая коллекция класса Time |
| 8. | Просмотр журнала событий | Объект класса Journal | | Журнал пуст | Сообщение «Журнал пуст» |
| Журнал не пуст | Выведенный на экран журнал |

## 2. Проектирование

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **Функция** | **Входные данные** | **Выходные данные** |
| **Основные функции** | | | |
| 1. | AddElement | List<Time>, Time t | List<Time>, сообщение об ошибке |
| 2 | Fill | List<Time>, int n | List<Time>, сообщение об ошибке |
| 3 | RemoveByNumber | List<Time>, int n | List<Time>, сообщение об ошибке |
| 4 | SortByHours | List<Time> | List<Time> |
| 5 | SortByMinutes | List<Time> | List<Time> |
| 6 | Display | List<Time> | List<Time> |
| 7 | DisplayJournal | Journal j | Journal j, сообщение «Журнал пуст» |

## 3. Листинг программы

## Person.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Lab13

{

class Program

{

public static int ReadInteger(int LowerLimit, int HigherLimit, string ErrorMessage)

{

int count;

bool confirmed;

do

{

confirmed = Int32.TryParse(Console.ReadLine(), out count);

if ((count < LowerLimit) || (count > HigherLimit) || (confirmed != true))

{

Console.WriteLine(ErrorMessage);

}

} while ((count < LowerLimit) || (count > HigherLimit) || (confirmed != true));

return count;

}

static void WorkCollection(MyNewCollection c, Journal j)

{

int s = 0;

do

{

Console.WriteLine("Что бы вы хотели сделать далее?");

Console.WriteLine("1 - Заполнить коллекцию случайными элементами");

Console.WriteLine("2 - Добавить элемент в коллекцию");

Console.WriteLine("3 - Удалить элемент из коллекции");

Console.WriteLine("4 - Подсчитать число элементов в коллекции");

Console.WriteLine("5 - Отсортировать коллекцию по числу часов");

Console.WriteLine("6 - Отсортировать коллекцию по числу минут");

Console.WriteLine("7 - Очистить коллекцию");

Console.WriteLine("8 - Просмотреть коллекцию");

Console.WriteLine("9 - Просмотреть журнал событий для коллекции");

Console.WriteLine("10 - Завершить работу с коллекцией");

s = ReadInteger(1, 10, "Вы ввели неправильное число. Введите число заново.");

switch (s)

{

case 1:

{

Console.WriteLine("Введите число элементов, которые вы хотите добавить в коллекцию");

c.Fill(ReadInteger(1, 100, "Вы ввели неправильное число. Введите число заново."));

Console.WriteLine("Коллекция успешно заполнена.");

break;

}

case 2:

{

Console.WriteLine("Введите число часов в добавляемом элементе");

int h = ReadInteger(0, 23, "Вы ввели неправильное число. Введите число заново.");

Console.WriteLine("Введите число минут в добавляемом элементе");

int m = ReadInteger(0, 59, "Вы ввели неправильное число. Введите число заново.");

c.Add(new Time(h, m));

Console.WriteLine("Элемент успешно добавлен");

break;

}

case 3:

{

Console.WriteLine("Введите номер удаляемого элемента");

if (c.RemoveByNumber(ReadInteger(0, 999999, "")) == true)

{

Console.WriteLine("Элемент успешно удален");

}

else

{

Console.WriteLine("Элемента с таким номером нет в коллекции");

};

break;

}

case 4:

{

Console.WriteLine($"В коллекции всего {c.Count()} элементов");

break;

}

case 5:

{

c.SortHours();

Console.WriteLine("Коллекция успешно отсортирована");

break;

}

case 6:

{

c.SortMinutes();

Console.WriteLine("Коллекция успешно отсортирована");

break;

}

case 7:

{

c.Clear();

Console.WriteLine("Коллекция успешно очищена");

break;

}

case 8:

{

c.Display();

break;

}

case 9:

{

j.Display();

break;

}

case 10:

{

break;

}

default:

{

Console.WriteLine("Вы ввели неправильное число. Введите число заново.");

break;

}

}

} while (s != 10);

}

static void Main(string[] args)

{

MyNewCollection c1 = new Lab13.MyNewCollection("Коллекция 1");

MyNewCollection c2 = new Lab13.MyNewCollection("Коллекция 2");

Journal j1 = new Lab13.Journal();

c1.CollectionCountChanged += new CollectionHandler(j1.CollectionCountChanged);

c1.CollectionReferenceChanged += new CollectionHandler(j1.CollectionReferenceChanged);

Journal j2 = new Lab13.Journal();

c1.CollectionReferenceChanged += new CollectionHandler(j2.CollectionReferenceChanged);

c2.CollectionReferenceChanged += new CollectionHandler(j2.CollectionReferenceChanged);

Console.WriteLine("Здравствуйте. Вас приветствует программа, демонстрирующая работу обработчиков событий.");

Console.WriteLine("Сейчас в программе созданы две коллекции типа <Time>, с которыми вы можете работать.");

Console.WriteLine("Также в ней созданы по одному журналу событий для каждой из этих коллекций, которые вы в любой момент можете просмотреть. Первый журнал подписан на изменения числа элементов и изменения ссылок на элементы из первой коллекции; второй журнал подписан на события изменения числа элементов в обеих коллекциях");

int s = 0;

do

{

Console.WriteLine("Что бы вы хотели сделать далее?");

Console.WriteLine("1 - Работать с первой коллекцией");

Console.WriteLine("2 - Работать со второй коллекцией");

Console.WriteLine("3 - Завершить работу с программой");

s = ReadInteger(1,3, "Вы ввели неправильное число. Введите число заново.");

switch (s)

{

case 1:

{

WorkCollection(c1, j1);

break;

}

case 2:

{

WorkCollection(c2, j2);

break;

}

case 3:

{

break;

}

default:

{

Console.WriteLine("Вы ввели неправильное число. Введите число заново.");

break;

}

}

} while (s != 3);

}

}

}

## 4. Тестирование

1) Выбор операции

|  |  |  |  |
| --- | --- | --- | --- |
|  | *Входные данные* | *Ожидаемый результат* | *Результат* |
| **Т1** | u = 0 | Натуральное число введено неверно. | Натуральное число введено неверно. Повторите ввод |
| **Т2** | u = 1 | Начало работы с коллекцией 1 | Начало работы с коллекцией 1 |
| **Т3** | u = 2 | Начало работы с коллекцией 2 | Начало работы с коллекцией 2 |
| **Т4** | u = 3 | Выход из программы | Выход из программы |
| **Т5** | u = 4 | Натуральное число введено неверно. | Натуральное число введено неверно. Повторите ввод |
| **Т6** | u = abc | Натуральное число введено неверно. | Натуральное число введено неверно. Повторите ввод |
| **Т7** | u = 1.5 | Натуральное число введено неверно. | Натуральное число введено неверно. Повторите ввод |

2) Ввод номера элемента коллекции из M элементов

|  |  |  |  |
| --- | --- | --- | --- |
|  | *Входные данные* | *Ожидаемый результат* | *Результат* |
| **Т1** | N = 1 | Продолжение работы с программой | Продолжение работы с программой |
| **Т2** | N = M-1 | Продолжение работы с программой | Продолжение работы с программой |
| **Т3** | N = M | Продолжение работы с программой | Продолжение работы с программой |
| **Т4** | N = M+1 | Натуральное число введено неверно. | Натуральное число введено неверно. Повторите ввод |
| **Т5** | N = 0 | Натуральное число введено неверно. | Натуральное число введено неверно. Повторите ввод |