Лабораторная работа 4

Делегаты, лямбда-выражения, события

Информация для всех вариантов

В лабораторной работе требуется определить класс-коллекцию **XCollection**, содержащий в себе элементы типа **X**. Для хранения элементов использовать стандартные коллекции.

**XCollection** изменяется при добавлении, удалении элементов **X** (событие **count change**) или изменении одного из элементов (ссылки) в коллекции (событие **reference** **change**). В этом случае в соответствующих методах или свойствах класса **XCollection** вызываются(**invoke**) специально объявленные делегаты**.**

Для хранения и вызова в **XCollection** методов «**подписанных»** классов (в нашем случаем это будут методы класса **Journal**) необходимо использовать **закрытые** делегаты и **открытые** события. Подписку и отписку методов проводить только через **события** и их служебные методы (add, remove), никакого прямого доступа к закрытым делегатам быть не должно.

Для сортировки элементов коллекции необходимо использовать метод **Sort (лямбда-выражение с полем сортировки)**. Использование лямбда-выражений для других целей приветствуется.

Определить класс хранения информации о событии **XListHandlerEventArgs**, производный от класса **EventArgs** включающий в себя

* открытое автосвойство типа **string** с названием коллекции, в которой произошло событие;
* открытое автосвойство типа **string** с информацией о типе изменений в коллекции; (типы изменений назвать самостоятельно)
* открытое автосвойство типа **X** для хранения 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.

Объявить новый тип данных - делегат **XListHandler** с сигнатурой:

delegate void XListHandler

(object source, XListHandlerEventArgs args);

**Определить класс XCollection, который содержит в себе**

* закрытое поле типа **List**<**X**>;
* автосвойство типа **string** с названием коллекции;
* метод **bool** **Add** **(int j, X record)** для добавления элемента **record** в позицию **j** списка List<**X**>; если в списке нет элемента с номером **j**, метод ничего не добавляет и возвращает значение **false**;
* метод **bool** **Remove** **(int j)** для удаления элемента с номером j из списка List<**X**>; если в списке нет элемента с номером j, метод возвращает значение **false**;
* индексатор **this** типа **X** (с методами **get** и **set**) с целочисленным индексом для доступа к элементу списка List<**X**> с заданным номером.

Названия событий в **XCollection** (тип делегата **XListHandler)**

* **onXCountChanged -** происходит при добавлении нового элемента в коллекцию или при удалении элемента из коллекции; через объект **XListHandlerEventArgs** cотбытие 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 **X**;
* **onXReferenceChanged -** происходит, когда одной из ссылок, входящих в коллекцию, присваивается новое значение; через объект **XListHandlerEventArgs** событие передает имя коллекции, строку с информацией о том, что был заменен элемент в коллекции, и ссылку на новый элемент **X**.

Событие **XCountChanged** вызывают следующие методы класса **XCollection**

* **AddDefaults(); -** заполнение коллекции произвольными заранее заданными данными
* **AddX (params X[] ) ; -** добавление в коллекцию указанных элементов
* **Remove (int j)** – удаление j элемента

\*Событие **XReferenceChanged** вызывается в методе set индексатора, определенного в классе **XCollection**.

Определить класс **Journal**, который можно использовать для накопления информации об изменениях в коллекциях типа **XCollection**. 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 автосвойство типа **string** с названием коллекции, в которой произошло событие;
* открытое автосвойство типа **date** с информацией о времени, когда произошло событие;
* открытое автосвойство типа **string** с информацией о типе изменений в коллекции;
* открытое автосвойство типа **string** c данными объекта **X**, с которым связаны изменения в коллекции;
* конструктор для инициализации полей класса;
* перегруженную версию метода string **ToString**().

Класс **Journal** содержит

* закрытое поле типа **List**<**JournalEntry**>;
* **реализует** обработчики **XCountChanged** и **XReferenceChanged** (для подписки на события в **XCollection)** которые при вызове добавляют новый элемент **JournalEntry** в список **List**<**JournalEntry**>; для инициализации **JournalEntry** используется информация из объекта **XListHandlerEventArgs**, который передается вместе с событием;
* перегруженную версию метода string **ToString**() для формирования строки с информацией обо всех элементах списка **List**<**JournalEntry**>.

В методе **Main**()

1. Создать две коллекции **XCollection**.  
2. Создать объекта типа **Journal** и подписать егона события **onXCountChanged** и **onXReferenceChanged** всех коллекций **XCollection**3. Внести изменения в коллекциях **XCollection**

* добавить элементы в коллекции;
* удалить некоторые элементы из коллекций;
* присвоить некоторым элементам коллекций новые значения;
* провести сортировку коллекции по разным полям
* провести сортировку журнала по типам операции

4. Вывод программы должен показать работу всех используемых технологий

Варианты заданий

Имя класса и поля класса Х для XCollection взять из лабораторной работы №3

Например для варианта 1 это будет класс Student с полями

**□ фамилия и инициалы;**

**□ номер группы;**

**□ успеваемость (массив из пяти элементов).**