№9 Делегаты, события и лямбда выражения

Задание

1. Используя делегаты (множественные) и события промоделируйте ситуации, приведенные в таблице ниже. Можете добавить новые типы (классы), если существующих недостаточно. При реализации методов везде где возможно использовать лямбда-выражения.

2. Создайте пять методов пользовательской обработки строки (например, удаление знаков препинания, добавление символов, замена на заглавные, удаление лишних пробелов и т.п.). Используя стандартные типы делегатов (Action, Func) организуйте алгоритм последовательной обработки строки написанными вами методами.

Далее приведен перечень заданий.

Вариант	Задание
1, 2, 15	Создать класс *Пользователь* с событиями *Переместить* (можно задать смещение) и *Сжать* (коэффициент сжатия). В main создать некоторое количество объектов *разного типа.* Часть объектов подписать на одно событие, часть на два (часть можете не подписывать). Проверить состояния объектов после наступления событий.
3, 4	Создать класс *Директор* с событиями *Повысить* (можно задать величину) и *Штраф*. В main создать некоторое количество объектов *токарей и студентов-заочников (или т.п.).* Часть объектов подписать на одно событие, часть на два (часть можете не подписывать). Проверить состояния зарплаты после наступления событий.
5, 6	Создать класс *Игра* с событиями *Атака* и *Лечить*. В main создать некоторое количество игровых объектов. Подпишите объекты на события произвольным образом. Реакция на события у разных объектов может быть разной (без изменения, увеличивается/уменьшается уровень жизни). Проверить состояния игровых объектов после наступления событий, возможно не однократном.
7, 8,	Создать класс *Boss* с событиями *Upgrade* и *Turn-on* (под определенным напряжением, учтите что техника или человек могут сломаться). В main создать некоторое количество объектов (техники или полу-техники). Подпишите объекты на события произвольным образом. Реакция на события у разных объектов будет разная.

	Проверить результаты изменения состояния объектов после наступления событий, возможно не однократном.
9, 10	Создать класс *Программист* с событиями *Удалить* и *Мутировать*. В main создать некоторое количество объектов (списков). Подпишите объекты на события произвольным образом. Реакция на события может быть следующая: удаление первого/последнего элемента списка, случайное перемещение строк и т.п. Проверить результаты изменения состояния объектов после наступления событий, возможно не однократном
11,12	Создать класс *Пользователь* с событиями *upgrade* и *work*. В main создать некоторое количество объектов (ПО). Подпишите объекты на события произвольным образом. Реакция на события может быть следующая: обновление версии, вывод сообщений и т.п. Проверить результаты изменения объектов после наступления событий.
13,14	Создать класс *Программист* с событиями *Переименовать* и *Новое свойство*. В main создать некоторое количество объектов (языков программирования). Подпишите объекты на события произвольным образом. Реакция на события может быть следующая: изменение имени/версии, добавление новых операций, технологий или понятий. Проверить результаты изменения состояния объектов после наступления событий, возможно не однократном

Вопросы

1. Что такое делегат? Как определить делегат?
2. Назначение делегатов.
3. Какие есть способы присваивания делегату адреса метода?
4. Поясните назначение метода Invoke.
5. Что такое групповая адресация делегата?
6. Как создать событие?
7. Как события связаны с делегатами? Опишите и поясните схему взаимодействия.
8. Что такое лямбда-выражения? Приведите пример лямбда-выражения с несколькими параметрами.
9. Что такое ковариантность и контравариантность делегатов? Что это дает?
10. Поясните разницу между встроенными делегатами Action и Func.