## Лабораторная работа №8. Работа с коллекциями

#### Цель работы

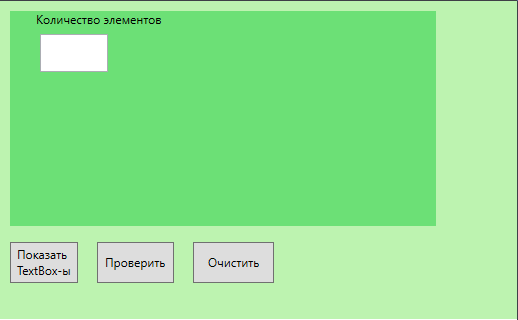
Освоение реализации обработки коллекций (массивов, списков).

#### Полезная информация

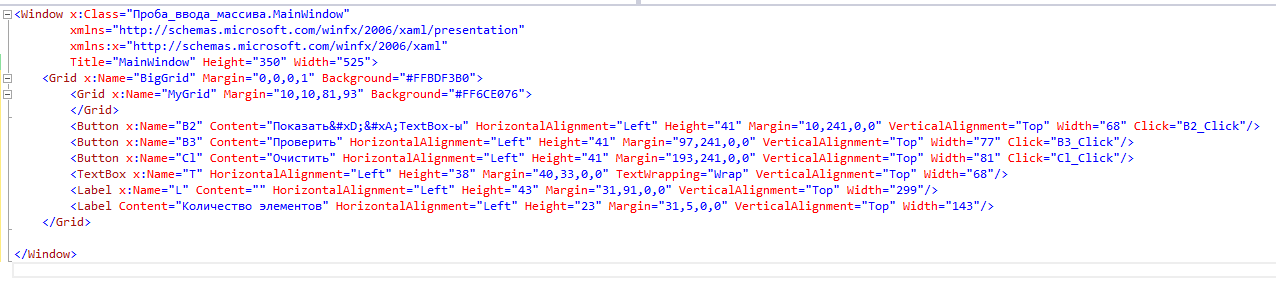
Ниже следует инструкция, в которой рассмотрена динамическая генерация элементов управления в приложении. В качестве примера здесь фигурируют текстовые поля (TextBox). Отметим, что это лишь пример. У вас могут оказаться другие элементы управления, другие действия с ними.

##### Добавление контейнера

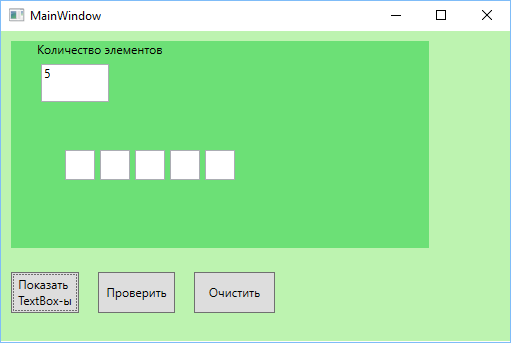
1. Расположите в окне разметки XAML еще один элемент Grid. Этот новый объект должен оказаться внутри уже существующего элемента Grid. Дайте новому элементу имя. В примерах в коде он будет называться MyGrid. Перед началом работы программы он будет пуст. Затем именно на нем будут размещены все генерируемые программой TextBox-ы. Все остальные элементы – TextBox для ввода количества элементов, метка или TextBox для вывода результата, будут расположены на основном элементе Grid.
2. Например, в конструкторе ваше приложение может выглядеть так:



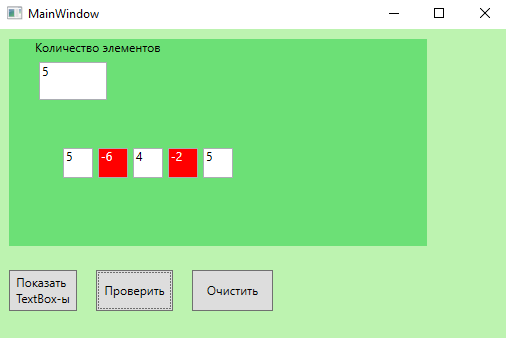
1. Элемент MyGrid чуть меньше основного элемента и выделен темным цветом, хотя и размер, и цвет могут совпадать. Сейчас этот объект пуст. Несмотря на то, что метка, содержащая текст «Количество элементов» и пустой TextBox кажутся расположены на MyGrid, это не так. Они расположены на основном элементе Grid, который называется BigGrid. Это можно увидеть (и отрегулировать) с помощь окна, содержащего код разметки:



1. Ввод массива (массивов) должен быть реализован следующим образом: в TextBox‑е задается количество элементов, после чего на MyGrid генерируется необходимое число TextBox-ов: Если требуется ввести два массива, то это количество удваивается.



1. Далее предполагается ввод значений и попытка произвести вычисления. Если данные некорректны, требуется выдать сообщение и выделить именно некорректные данные. Выделять можно разными способами: цветом фона или цветом текста. В примере выделены отрицательные числа, использованы оба способа выделения:



1. Предусмотрена также кнопка очистки, которая уберет все TextBox с MyGrid и очистит TextBox с количеством элементов.

##### Массив элементов управления

Объявлять можно не только массив чисел, но и массив других объектов. Например, мы будем работать с массивом TextBox-ов. Однако, с этим массивом мы будем работать при нажатии разных кнопок. Поэтому сам он и количество элементов в нем должны быть доступны разным обработчикам событий. Следовательно, мы не можем объявлять их внутри этих обработчиков, а должны объявить полями нашего основного класса MainWindow:

Изображение выглядит как текст

Автоматически созданное описание

Обратите внимание на то, где именно это сделано. Пока не выполняется выделение памяти под массив с помощью new, так как количество элементов n еще не задано, переменная n просто объявлена. Далее как переменной n, так и массивом TextBox-ов tb смогут пользоваться все методы класса.

##### Как в программе вывести нужное количество TextBox-ов?

Для этого нужны следующие действия:

1. Узнать, сколько потребуется TextBox-ов.
2. Выделить под них память.
3. Сгенерировать в цикле эти TextBox -ы, делая следующие действия:
4. 3.1. Выделить память под один TextBox.

3.2. Установить размеры и содержимое TextBox-а.

3.3. Добавить TextBox на элемент MyGrid.

3.4. Установить расположение TextBox-а.

Код для выполнения этих действий приведен ниже:

Изображение выглядит как текст

Автоматически созданное описание

В объекте pos указано расположение каждого TextBox-а, при этом первый параметр задает координату *х*, второй – *у*. Следует отметить, что размеры и смещение у вас могут быть свои.

##### Итак, у нас есть массив TextBox-ов. Как с ним работать дальше?

Дальше мы будем работать с обычным массивом, как и раньше. Все, что нужно – это скопировать содержимое наших TextBox-ов в этот массив. В коде ниже объявляется числовой массив, заполняется содержимым TextBox-ов, после чего выполняется проверка данных: все отрицательные числа выделяются белым цветом на красном фоне:

Изображение выглядит как текст

Автоматически созданное описание

##### Как выполнить очистку?

Итак, надо выполнить очистку, чтобы запустить программу по новой. Заполнять TextBox пустой строкой вы уже умеете. Но как убрать все сгенерированные, ставшие ненужными TextBox-ы? Нужно просто очистить решетку MyGrid:

Изображение выглядит как текст

Автоматически созданное описание

##### Как рисовать графические объекты в программе

Чтобы создать какой-нибудь элемент (прямоугольник, окружность, кнопку, TextBox не с помощью конструктора, а в коде, необходимо инициализировать объект заданного типа. Рассмотрим это на примере линий. Объявим объект класса Line.



Здесь объявляется объект «линия», которая называется myL (а могла бы и как-нибудь по-другому). После этого можно менять различные свойства линии, обращаясь к ним через точку. Например, строчка

myL.Stroke=Brushes.Red;

покрасит линию myL в красный цвет.

Для работы понадобятся следующие свойства:

|  |  |
| --- | --- |
| **Имя свойства** | **Что собой представляет** |
| X1 | Х координата первой точки |
| Y1 | Y координата первой точки |
| X2 | Х координата второй точки |
| Y2 | Y координата второй точки |
| Stroke | Цвет линии |
| StrokeThickness | Толщина линии |

Важно! Координаты нумеруются от левого верхнего угла. Именно там находится точка (0;0).

Однако, чтобы линия появилась на экране, недостаточно инициализировать объект и установить свойства. Линию потребуется добавить в контейнер, где размещаются все объекты. Когда вы создаете приложение, у вас уже есть элемент Window (окно) и элемент Grid (решетка). Grid и будет тем контейнером, где мы будем размещать объекты. Для этого его надо назвать. Пусть ему будет дано имя myG. Тогда строчка, добавляющая линию myL в контейнер myG буде выглядеть следующим образом:



Если необходимо рисовать линии в цикле, следовательно, инициализировать их, менять свойства и добавлять в контейнер надо также в цикле.

Аналогично линиям добавляются и другие элементы.

#### Порядок выполнения лабораторной работы

1. Реализуйте приложение в соответствии с вариантом. При реализации необходимо использовать одномерные и двумерные массивы, списки – в зависимости от ситуации. Набор структур каждый выбирает сам так, чтобы эффективно реализовать алгоритм приложения. При необходимости внесите изменения в библиотеку классов.
2. В приложении параметры объекта (или объектов) задаются пользователем. При этом пользователь выбирает, к какому подклассу будет относиться объект.
3. Предусмотреть возможность сбросить все параметры и начать работу с приложением заново.
4. Проверьте работоспособность вашего приложения.

#### Варианты

1. 1. Приложение должно реализовывать битву заданного количества драконов. Одна атака проводится по нажатию одной кнопки. Схема битвы соответствует нерегламентированной схватке «все со всеми». Атакует случайный дракон случайного дракона (но не себя). Дракон, которого «убили», выбывает из схватки. Он не может атаковать, и его тоже не могут атаковать. В удобной форме выводить статистику боя, показывая результат каждого удара. Предусмотреть кнопку «Закончить битву», после которой будет выдано состояние каждого из драконов и «Начать заново», после которой можно будет по новой вводить данные.
2. 1. Провести рыцарский турнир среди заданного количества рыцарей по схеме «на выбывание». Рыцари случайным образом разбиваются на пары, в парах определяются победители, после чего между собой бьются уже победители. Если в какой-то момент рыцарю не хватило пары, то он автоматически выходит победителем из этой схватки. Изобразить результаты турнира в форме «дерева» (смотреть картинку):



и выдать сообщение с поздравлением победителя. Предусмотреть возможность начать турнир заново.

1. 1. Вводятся параметры принцессы. Охарактеризовать состояние принцессы по ИМТ, учитывая вес, рост и возраст. При этом можно воспользоваться приведенной ниже таблицей или ее аналогом



* 1. В замке несколько башен с разными характеристиками (число башен и характеристики задаются пользователем). Принцесса помещается в одну из башен случайным образом. Приложение должно вывести список башен (или отобразить башни) выделив ту, в которой находится принцесса.

1. 1. Вводятся параметры заданного пользователем числа магов. Вывести ту стихию, которая является среди этих магов наиболее распространенной. Если таких несколько, вывести все наиболее распространенные.
2. 1. Вводятся параметры заданного пользователем числа артефактов. Вывести ту стихию, которая является среди артефактов наиболее редкой и повысить цену на случайное число процентов (от 5 до 20) за редкость. Вывести итоговый список артефактов. Если «самых редких» будет больше одного, то повышение цены коснется всех самых редких артефактов.
3. 1. Вводятся параметры заданного пользователем числа вкладов. Объединить все однотипные вклады в один. Вывести обновленный список вкладов.
4. 1. Пользователем вводятся события, происходившие в истории замка – летопись. Формат записи: год (число), событие (строка). Добавить летопись как поле класса. По запросу вывести информацию с летописью замка, но так, чтобы все события были выведены в хронологическом порядке. Если события произошли в один год, то более ранним считается то, которое было добавлено в летопись раньше.
5. 1. Вводятся параметры сделанных разными мастерами наборов рун. Каждый мастер изготавливал некоторое количество наборов из разного материала. Определить для каждого мастера самый частый и самый редкий для него материал (учесть, что таких материалов может оказаться несколько). Вывести самый частый и самый редкий материалы среди всех наборов.
6. 1. Вводятся параметры различных судов. Необходимо для погрузки выбрать то судно, скорость которого будет наибольшей (при условии, если груз на судно помещается). Таких судов может быть несколько, в этом случае показать каждое.
7. 1. Вводятся параметры лошадей. Вывести самую распространенную и самую редкую масти.
8. 1. Приложение должно реализовывать драку заданного количества разбойников. Одна атака проводится по нажатию одной кнопки. Удар наносит случайный разбойник случайному противнику-разбойнику (но не себе). Разбойник, которого «убили», выбывает из драки. Он не может атаковать, и его тоже не могут атаковать. В удобной форме выводить статистику драки, показывая результат каждого удара. Предусмотреть кнопку «Закончить схватку», после которой будет выдано состояние каждого из разбойников (HP и деньги) и «Начать заново», после которой можно будет по новой вводить данные.
9. 1. Вводятся параметры заданного числа питомцев. По каждой из характеристик выдайте среднее арифметическое значение и медиану среди всех питомцев. Медиана – это значение, которое будет в середине всех элементов, если их отсортировать. То есть, слева будет находиться такое же количество меньших элементов, сколько справа – больших. Если число элементов четных, то берется среднее значение между двумя «серединками». Например, для набора 6, 4, 8, 17, 3 медианой будет 6, для набора 6, 4, 8, 17, 3, 5 медианой будет среднее арифметическое между 5 и 6, т.е. 5,5.
10. 1. Вводятся параметры монстра. Разделать его тушу, отрезая каждый раз кусок случайного веса. Резать до тех пор, пока не останется последний кусок, не превышающий 5% от первоначальной массы монстра. Вывести информацию о расфасованных кусках (вес, стоимость). Подсчитать общую сумму выручки с монстра и среднее арифметическое весов кусков.
11. 1. Вводится информация о юнитах, доступных для найма в армию. Персонаж набирает себе юнитов, указывая количество юнитов каждого типа. Подсчитать общую атаку армии (с учетом атаки и количества каждого юнита), а также вклад в процентах каждого вида юнитов в общую атаку (учитывать только тех, кого наняли).
12. 1. Реализовать приложение «Лавка целителя». Вводятся данные об имеющихся у целителя аптечках. Подобрать для определенного персонажа (параметры которого вводятся пользователем) наиболее подходящую аптечку. Аптечка должна полностью вылечить персонажа (если это невозможно – максимально приблизить число его HP к максимальному), и при этом быть наиболее дешевой. Если окажется, что аптечка с бо́льшим числом HP, чем надо, будет дешевле – выбирать ее. Если есть несколько равнозначных аптечек, показать все наиболее подходящие.
13. 1. Вводятся параметры различных источников. Вывести параметр, для которого больше всего источников и параметр, для которого меньше всего. Если вариантов ответа несколько, выводить все.
14. 1. Вводятся параметры различных темниц. Определить самую наполненную темницу (в зависимости от числа заключенных и метража) и самую ненаполненную. Переселить часть узников из самой переполненной в самую ненаполненную. Часть берется так, чтобы в самой ненаполненной плотность узников не превысила ранее вычисленное среднее значение.
15. 1. Вводятся параметры имеющихся в лавке артефактов. Пользователь набирает артефакты в корзину, далее обеспечивается продажа. Скидка дается только на один артефакт. Случайным образом определяется, понравился ли покупатель хозяину лавки, не понравился, или хозяин остался нейтральным. В случае, если покупатель хозяину понравился, скидка дается на самый дорогой артефакт, если не понравился – на самый дешевый. При нейтральном отношении для скидки выбирается случайный артефакт, кроме самого дорогого и самого дешевого.
16. 1. Вводятся параметры всех таверн города. Рыцарь, победив дракона, ходит отмечать это событие по тавернам (у рыцаря есть определенная сумма денег, она вводится пользователем). Таверна, в которую попадает рыцарь, определяется случайным образом. Таверны могут повторяться, но никогда рыцарь не заходит в одну и ту же таверну два раза подряд. В каждой таверне он тратит деньги, может кого-то ограбить или быть ограбленным. Если в таверне нет свободных мест, то выбирается следующая таверна. Без денег зайти в таверну рыцарь не может. Поэтому закончит пировать он либо тогда, когда законатся деньги, либо когда свалится (определяется случайным образом). Вывести всю статистику похождений рыцаря, и сколько денег у него осталось в итоге.
17. 1. Вводятся параметры разных фрагментов местности, по которым надо пройти персонажу. Вычислить время, которое затратит персонаж на весь путь, с учетом вида местности каждого фрагмента. Выделить местность, на которую будет затрачено наибольшее количество времени и наименьшее.
18. 1. Вводятся параметры деревьев на участке. Вывести, каких деревьев на участке больше всего, и каких меньше. Вывести их параметры. Если самых частых или самых редких несколько – вывести все из них.
19. 1. Вводятся параметры лабиринта. Приложение должно изобразить лабиринт, в котором расставлены стены и препятствия. По нажатию кнопки перехода на следующий уровень показывается следующий уровень лабиринта. Стены на каждом уровне определяются уровнем. Цвет должен соответствовать указанному в параметре цвету. Препятствия расставляются случайным образом, но так, чтобы они не совпадали друг с другом, со стеной, а также с входом и выходом. Для реализации разбейте лабиринт на клетки, каждый уровень представьте матрицей. Клетка может быть свободна, может быть со стеной, может быть входом или выходом.
20. 1. Вводятся параметры напитков, имеющихся в лавке. Вывести вид упаковки, являющейся самой распространенной и вид упаковки, являющейся наиболее редкой. Если таких несколько, то обозначить все подходящие.
21. 1. Вводится информация о владельцах шахт, затем – о шахтах каждого владельца. Подсчитать ежедневный доход каждого из владельцев по каждому из ресурсов. Определить суммарный доход каждого владельца, а также среднее арифметическое доходов: по каждому из ресурсов, по каждому из владельцев.
22. 1. Вводятся параметры разных книг. Вывести название книги, отвечающей запросу пользователя. Пользователь выбирает параметр из ComboBox (можно реализовать с помощью двух выборов): книги с наибольшим числом страниц, книги с наибольшим процентом заполненности, книги с наименьшим числом страниц, книги с наименьшим процентом заполненности, среднее число страниц, средний процент заполненности.
    2. Вводятся параметры разных деревень (с указанием владельцев). Вывести таблицу владельцев в порядке убывания числа деревень. В каждой строке выводить владельца, число деревень, суммарный доход с них.
    3. Пользователем вводится список университетов. Далее пользователь выбирает навыки, которые хотел бы прокачать, составляя себе их список (выбор возможен только из тех навыков, которые есть в университетах, которые есть в списке). Подобрать для пользователя список университетов, в которые ему стоит пойти учиться. Если университеты обучают одному навыку, то необходимо выбирать сперва более высокий по уровню, а при равных уровнях – тот, что дешевле.

1. 1. Добавьте в базовый класс поле с названием здания. Далее вводится информация о зданиях. Вывести таблицу всех зданий в порядке от самого выгодного (с наибольшей прибылью) до самого невыгодного. В каждой строке выводить название и размер прибыли.
2. 1. Реализуйте рыцарский турнир, храня результаты в турнирной таблице. Турнир производится по схеме, где каждый рыцарь бьется с каждым (кроме, разумеется, себя) по одному разу. За победу дается 3 очка, за ничью – 1 очко, за проигрыш – 2 очка (результаты вводятся пользователем). Распределите выигрыш между рыцарями в зависимости от места. Выведите итоговую таблицу, расположив победившего в начале, занявшего второе место – за ним и т.д. В каждой строке выводится идентификатор рыцаря (можно номер), количество набранных очков, выигранная (проигранная) сумма.
3. 1. Формируется список блюд ресторана. Каждый раз калорийность и стоимость задаются для 100 г блюда. Пользователь выбирает блюда для банкета, указывая их вес, который он хочет заказать. Рассчитать стоимость банкета.
4. 1. Вводится информация о кланах. Отсортируйте кланы по уровню в порядке убывания. Для каждого клана укажите уровень, общую атаку, число человек в клане. Внутри уровня кланы сортируются по общей атаке, далее – по числу человек.
5. 1. Вводятся сведения о странах. В приложении должна быть кнопка «начать мировую войну». После этого страны должны быть разбиты на коалиции случайным образом. Число коалиций задается пользователем. Равномерного распределения не требуется. Далее в войне должно пройти заданное пользователем число локальных сражений между странами. Какая страна на какую нападает, определяется случайным образом, однако страны, относящиеся к одной коалиции, воевать друг с другом не могут. Вывести всю статистику произошедшего (по каждому локальному сражению) и итог передела мира для каждой страны: название, площадь, население, потери убитыми.
6. 1. Задается список городов. Далее задается время пути между каждыми двумя городами. Задается время только между теми городами, между которыми есть дорога непосредственно из одного города в другой. Пользователь указывает, с какого города он хочет начать и какие города он хотел бы посетить. Разработать маршрут так, чтобы путешественник посетил каждый город по одному разу, и чтобы время на путешествие было минимальным. Если таких маршрутов несколько, выдать один из них.
7. 1. В базовом классе должно быть поле, сколько времени надо затратить на достопримечательность (подразумевалось, что это «оптимальное время на посещение», если у вас это не так, измените поле). Вводится список достопримечательностей с их параметрами. Пользователь выбирает, какие достопримечательности он хочет посетить. Необходимо составить программу посещения города так, чтобы турист мог посетить все желаемые достопримечательности. Возможно, понадобится разбить на поездку на несколько дней. Для каждой достопримечательности укажите, во сколько турист должен начать ее посещение. Подразумевается, что все достопримечательности доступны с 9:00 до 20:00 ежедневно.
8. 1. Вводится информация о разных планетах. Вывести планеты, отсортировав их по убыванию значения, которое выберет пользователь. Это может быть масса, длительность суток или плотность.
9. 1. Вводится список инопланетян. Обработайте список так, чтобы имена инопланетян не повторялись. Если у инопланетян все параметры совпадают, то это означает, что информация продублировалась, и надо удалить лишних существ. Если же хотя бы один параметр отличается, то это просто совпадение имен. И надо изменить имя каждого такого инопланетянина, добавив к его имени порядковый номер. Выдайте итоговый список.
10. 1. Вводится список космических кораблей. При этом часть кораблей отправлена в полет. На ТО корабля требуется определенное время: 3 дня на каждый отсек плюс 2 дня на весь корабль. Составьте очередь на ТО, поставив в начало очереди те корабли, которым осталось меньше времени до прохождения ТО. Укажите дату начала и дату окончания ТО. При этом учтите, что часть кораблей сможет пройти ТО только тогда, когда вернется из полета.
11. 1. Вводится информация о космонавтах. Вывести список всех космонавтов, от космонавта с наибольшим числом часов полета, к космонавту с наименьшим. Выводится имя и число часов.
12. 1. Вводится информация о кометах. Вывести список элементов, которые встречаются в составе комет, от самого распространенного к самому редкому.
13. 1. Вводится информация об организмах. Распределить их по группам так, чтобы в каждой группе были организмы только одного типа (типы определялись в лабораторной работе №5).
14. 1. Вводится информация о наперсточниках. Вывести список всех языков общения с наперсточниками, от самого распространенного до самого редкого.
15. 1. Вводится информация о межзвездных маршрутах. Выведите самую распространенную и самую редкую цели маршрутов (если критерию соответствуют несколько, выводить все).
16. 1. Вводятся сообщения. Для каждой пары сообщений определить расстояние Левенштейна. Расстояние Левенштейна – это минимальное количество односимвольных операций (а именно вставки, удаления, замены), необходимых для превращения одной последовательности символов в другую. Подробнее смотреть тут: [Википедия. Расстояние Левенштейна](https://ru.wikipedia.org/wiki/%D0%A0%D0%B0%D1%81%D1%81%D1%82%D0%BE%D1%8F%D0%BD%D0%B8%D0%B5_%D0%9B%D0%B5%D0%B2%D0%B5%D0%BD%D1%88%D1%82%D0%B5%D0%B9%D0%BD%D0%B0) и вообще на просторах сети.
17. 1. Реализуйте игру «Виселица». При этом пользователь сам выбирает тип игры. Текущее состояние игры записывается в экземпляр класса и отображается пользователю. Графика может быть реализована схематично.
18. 1. Реализуйте игру. Пользователем вводится загадываемое слово, которое для игрока будет представлено закрытыми карточками для букв. Далее игроку выдаются 24 буквы для игры: обязательно туда должны входить нужные буквы в требуемом количестве. Остальные «фишки» должны быть дополнены случайными буквами. Чтобы отгадать слово, пользователь должен нажать на фишку. Если такая буква есть, то она встает на нужное место. В экземпляр класса записывается состояние игры, статистика выдается пользователю. Подсчитывать также число попыток (нажатий на «фишки»).
    2. Измените библиотеку классов так, чтобы рецепт мог состоять из неопределенного числа ингредиентов: содержать не три ингредиента, а список ингредиентов. Изменить все методы, работающие с этими ингредиентами. Реализовать приложение, которое будет пересчитывать для рецепта пропорции ингредиентов.
19. 1. Для двух персонажей вводится список загадчиков, к которым они попадали (для каждого персонажа – свой список). Вывести тех, к кому попали оба (проверять по именам).