**МЕЖГОСУДАРСТВЕННОЕ УЧРЕЖДЕНИЕ**

**ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ**

**«БЕЛОРУССКО-РОССИЙСКИЙ УНИВЕРСИТЕТ»**

**Кафедра «Автоматизированные системы управления»**

**ВИЗУАЛЬНЫЕ СРЕДСТВА РАЗРАБОТКИ ПРОГРАММИРАММНЫХ ПРИЛОЖЕНИЙ**

**Лабораторные работы**

**Объектно-ориентированное программирование.**

Могилев 2022

**Лабораторная работа №1**

**Классы, свойства, индексаторы.**

**Цель работы:** получить практические навыки по разработке программ с использованием классов.

**1 Задание:**

Создать класс согласно своего варианта. Создать соответствующий тип-*перечисление* (enum).

Класс должен содержать:

* 4-5 скрытых полей;
* одно из полей должно быть типа *перечисление;*
* на каждое поле должно быть свойство;
* одно из полей должно быть класса DateTime, для хранения даты рождения или создания объекта;
* на поле класса DateTime свойство должно быть только для чтения;
* свойство типа int c методами get и set для получения и изменения года рождения (создания) в закрытом поле типа DateTime;
* должно быть поле для хранения себестоимости (для класса Person – оклад);
* статическое поле, в котором будет храниться НДС (20%) (для класса Person – подоходный налог (12%));
* Создайте свойство, которое возвращает стоимость (себестоимость, увеличенная на НДС) или для класса Person – зарплату (оклад уменьшенный на подоходный налог);
* хотя бы одно свойство должно иметь проверку вводимых данных;
* конструктор по умолчанию, 2 конструктора с параметрами;
* конструктор копирования;
* индексатор, который принимает в качестве параметра (индекса) число (некую сумму) и возвращает Тrue, если передаваемая сумма больше стоимости (зарплаты) и False в противном случае;
* переопределенный метод ToString().

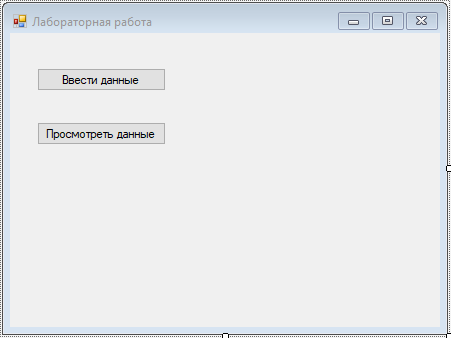
**2 Порядок выполнения работы**

1. Создайте новый проект Windows Forms.

2. Задайте имя форме и измените текст в названии окна.

3. Добавьте на форму две кнопки.

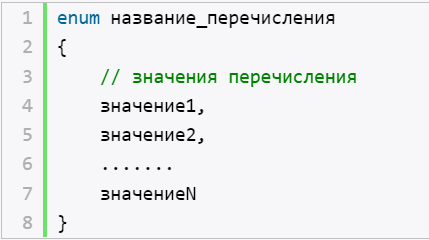
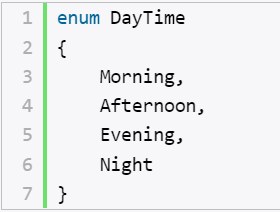
4. Задайте им соответствующие названия и текст.



5. Добавьте в проект класс. Сразу задайте ему необходимое имя. Объявите класс как **public**.

6. Создайте соответствующие поля.

7. Одно из полей должно быть типа перечисление. Сделайте его тоже **public**. Перечисление создается вне класса следующим образом:

8. Создайте конструктор с параметрами.

9. Создайте конструктор без параметров, который будет вызывать конструктор с параметрами и устанавливать какие-то «дежурные» значения.

10. Создайте конструктор с произвольным количеством параметров.

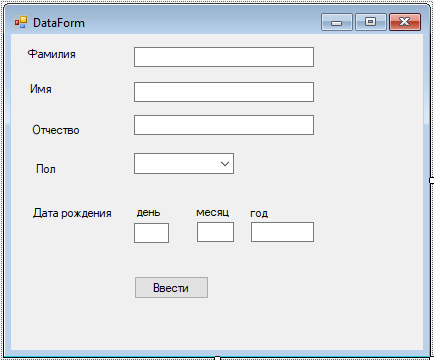
11. На все поля напишите свойства и свойства по заданию.

12. Свойство для работы с полем типа **enum**, должно принимать и возвращать тип String.

14. Добавьте в проект еще одну форму для ввода и просмотра данных.

(Для пункта, где будет использоваться перечисление добавьте ComboBox)

*Например:*



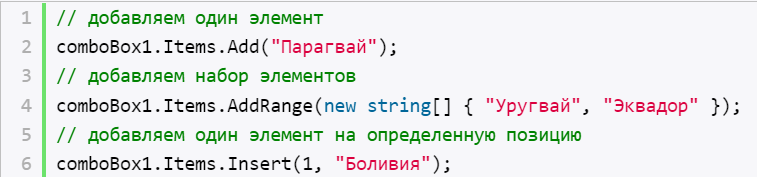
15. В классе главной формы создайте поле типа вашего класса (по заданию).

16. На главной форме щелкните по кнопке «Ввести данные» и в сгенерированном событии организуйте создание объекта класса, а так же создание и открытие второй формы.

17. Для того, что бы формы могли обмениваться данными между собой, организуем передачу объекта класса через конструктор. Для этого напишем еще один конструктор для второй формы. В качестве параметра он будет получать объект класса и вызывать конструктор по умолчанию.

В классе второй формы объявите поле типа вашего класса. В конструкторе объект, полученный в качестве параметра, положите в поле класса. Теперь в него можно записывать данные или читать их.

18. **ComboBox** представляет собой простой список. Ключевым свойством этого элемента является свойство **Items**, которое как раз и хранит набор всех элементов списка. Для того чтобы в ComboBox добавить записи, которые должны отображаться для выбора воспользуйтесь методом **Add** (новые элементы помещаются в конец списка), **AddRange** (позволяет добавить сразу несколько элементов, например, массив) или **Insert** (вставка производится по определенному индексу списка):



Программным путем добавить список в ComboBox лучше всего в конструкторе формы.

19. В обработчик события нажатия кнопки «Ввести» добавьте проверки и ввод данных из TextBox-ов в поля класса. Для проверки выбора записи в ComboBox-е воспользуйтесь свойством SelectedItem (оно возвращает тип Object, так что не забудьте привести его к нужному вам типу). Добавьте в функцию логическую переменную, которая будет хранить результат проверок. Если хотя бы одно поле не заполнено или заполнено неверно отобразите на форме сообщение об ошибке (добавьте Label и управляйте его отображением с помощью свойства Visible). Если все данные введены верно, они присваиваются полям класса и вторая форма закрывается.

20. Теперь организуем просмотр данных объекта.

21. На главной форме на щелчок кнопки «Просмотр данных» напишите создание и открытие второй формы. Но, прежде чем передавать в конструктор объект класса необходимо проверить, не пустой ли он.

22. Можно создать еще одну форму для просмотра данных объекта, осуществляя вывод не в TextBox-ы, а на Label-ы.

Другим вариантом будет вывод данных в ту же форму, где мы их вводили, но сделать поля недоступными для изменения. Для этого можно воспользоваться свойством ReadOnly для TextBox (true) и свойством Enabled для ComboBox (false). Для того чтобы форма знала, для чего ее открыли (для ввода или для просмотра) можно добавить еще один параметр логического типа в конструктор второй формы. Если форма открыта для просмотра, то в TextBox-ы выводятся значения полей объекта и они закрываются от доступа. Можно сделать кнопку «Ввести» невидимой, а добавить кнопку «Закрыть», для закрытия формы.

**3 Варианты заданий:**

|  |  |
| --- | --- |
| **№ варианта** | **Класс** |
| 1 | Телефон |
| 2 | Человек |
| 3 | Животное |
| 4 | Транспортное средство |
| 5 | Текст |
| 6 | Жилое здание |
| 7 | Человек |
| 8 | Лекарственное средство |
| 9 | Компьютер |
| 10 | Товар |

**Лабораторная работа №2**

**Наследование. Перегрузка методов и операторов. Полиморфизм.**

**Цель работы:** получить навыки использования механизмов наследования и полиморфизма классов, закрепить навыки перегрузки методов и операторов.

**1 Задание**

Построить иерархию классов предметной области согласно варианту.

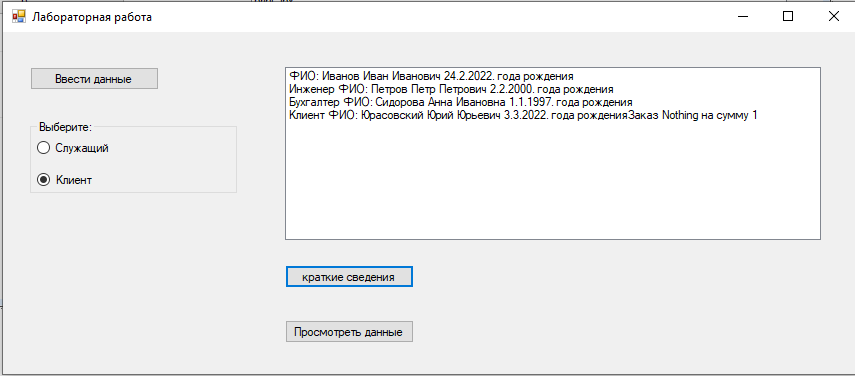
* В классах описать конструкторы с параметрами и конструкторы по умолчанию, свойства для установки и получения значений полей классов, методы для описания поведения объектов. Каждый из создаваемых классов должен иметь не менее двух методов, трех свойств, трех конструкторов.
* Для каждого созданного класса переопределить один и тот же виртуальный метод родительского класса.
* Создать массив из объектов обоих производных классов и продемонстрировать полиморфизм в действии.

**2** **Порядок выполнения работы**

1. Создайте два производных класса, согласно своему варианту. В каждом должно быть несколько дополнительных полей и соответствующих свойств. Напишите конструктор по умолчанию, с параметрами и конструктор, принимающий в качестве параметра объект базового типа.
2. В базовом классе реализуйте виртуальный метод (метод, который присущ всем объектам базового класса и его наследникам, но отличается для разных классов.)
3. Переопределите виртуальный метод определенный в базовом классе для каждого производного класса. Он может использовать метод базового класса и добавлять свой функционал или быть реализован совершенно по-другому, по вашему усмотрению.
4. На главной форме объявите массив объектов базового класса. Для этого удобней всего использовать List (см. пример лекции).
5. Добавьте на главную форму панель groupBox. В нее вложите переключатели RadioButton для определения типа нового создаваемого объекта. Проверить, включен ли переключатель можно с помощью логического свойства Checked.
6. Далее, прежде чем создать новую форму для ввода новых объектов, сначала создайте объект нужного класса (базового или производного) и положите его в созданный ранее массив. А затем передавайте его в конструктор формы.
7. Продумайте ввод данных специфических для разных производных классов. Для проверки типа объекта используйте конструкцию:

person is Employee

1. На главную форму добавьте элемент управления **ListBox.** Элемент ListBox представляет собой простой список. Ключевым свойством этого элемента является свойство Items, так же как и у элемента ComboBox.
2. Пример главной формы:



1. Организуйте обход в цикле всех объектов массива базового типа и вызов для них виртуального перегруженного метода. Результат отобразите в ListBox.
2. Чтобы очистить сразу весь список, примените метод Clear:

listBox1.Items.Clear();

|  |  |
| --- | --- |
|  |  |

1. Протестируйте все методы классов.

**3 Варианты заданий:**

|  |  |  |
| --- | --- | --- |
| **№варианта** | **Класс родитель** | **Производные классы** |
| 1 | Телефон | Смартфон, стационарный |
| 2 | Человек | Студент, преподаватель |
| 3 | Животное | Млекопитающее, птица |
| 4 | Транспортное средство | Грузовик, легковой автомобиль |
| 5 | Текст | Статья, заметка |
| 6 | Жилое здание | Коттедж, многоквартирный дом |
| 7 | Человек | Пациент, врач |
| 8 | Лекарственное средство | Таблетки, мазь |
| 9 | Компьютер | Ноутбук, десктоп |
| 10 | Товар | Продукт, домашняя техника |

**4 Контрольные вопросы**

1. Каким образом реализуется перегрузка конструкторов?
2. Можно ли перегрузкой операторов отменить очередность выполнения операции?
3. Какие операции требуют, чтобы левый операнд был объектом класса?
4. В чем заключена основная задача наследования?
5. Как осуществляется доступ к элементам производных и базовых классов?
6. Что такое полиморфизм? В чем он заключается и как используется при работе с классами?

**Лабораторная работа №3**

**Перегрузка методов и операций.**

**Цель работы:** получить практические навыки по разработке программ с использованием классов.

**Задание:**

В родительском классе:

* определите 2 метода, соответствующие функционалу класса;
* определите метод Equals() так, чтобы объекты считались равными, если равны значения полей объектов, а не ссылки объектов;
* переопределить операции == и != , с использованием метода Equals;
* переопределить операции < и >, <= и >=.

В классах наследниках:

* переопределите один из методов родительского класса;
* переопределить метод Equals;
* переопределить операции == и != ;
* переопределить операции < и >, <= и >=.

Протестируйте все методы классов, а также метод Equals().

**Лабораторная работа №4**

**Интерфейсы. Итераторы.**

**Цель работы:** ознакомление с основами объектного подхода в языке C#, приобрести практические навыки по работе с интерфейсами, закрепить навыки использования наиболее часто используемых интерфейсов в языке C#.

**1 Порядок выполнения работы**

1.1 В программе, разработанной в предыдущей лабораторной работе:

* определить интерфейс ISell (для классов Person интерфейс IDismiss (уволить)), который будет содержать:
  + свойство «дата продажи» (увольнения);
  + логическое свойство «продан»(уволен);
  + свойство «владелец» (причина);
  + метод Sell() (Dismiss());
* реализовать интерфейс ISell (IDismiss) во всех классах. Метод Sell() (Dismiss()) пусть пока просто изменяет значения свойств.
* реализовать в классе-родителе интерфейс IComparable для выполнения сортировки по цене или зарплате;
* реализовать в классе-родителе интерфейс IComparer <«класс-родитель»> для выполнения сортировки по дате создания или рождения;
* реализовать во вспомогательном классе интерфейс IComparer <«класс-наследник»>, для сортировки по признаку, принадлежащему классу-наследнику;
* создать класс, который будет содержать в себе массив родительского класса, в котором будут храниться все объекты.
* определить в этом классе методы по поиску и по всем видам сортировок;
* реализовать в классе-родителе интерфейс System.Collections.IEnumerable для перебора объектов, по отдельному признаку;
* определить вспомогательный класс, реализующий интерфейс System.Collections.IEnumerator для производного класса;

1.2 В качестве отчета предоставить работающее программное обеспечение.

1.3 Защитить лабораторную работу.

**2 Контрольные вопросы**

1. Что такое интерфейс?
2. Какие элементы языка C# могут быть членами интерфейса?
3. Сколько интерфейсов может наследовать класс?
4. Где должны быть реализованы методы интерфейса?
5. Можно ли реализовать множественный интерфейс?
6. Как проявляется принцип полиморфизма при использовании интерфейсов?

**Лабораторная работа №5**

**Создание интерфейса пользователя. Логирование.**

**1 Порядок выполнения работы:**

1.1 В программе, разработанной в предыдущей лабораторной работе добавить следующий функционал:

1. Главная форма при запуске приложения предлагает войти в систему как администратор или как пользователь.
   1. Если будет выбран администратор, то форма предлагает выбрать фамилию из списка и ввести пароль. Если пароль введен верно программа переходит на страницу с личными данными.
   2. Если будет выбран пользователь, то программа переходит на форму просмотра информации.
2. Форма просмотра информации предполагает возможность просмотреть список общий или по критериям. Некоторые данные к просмотру не доступны ( например зарплата или стоимость). Но есть возможность проверить достаточна ли сумма для покупки. Можно осуществлять сортировку по различным критериям и поиск по названию, по дате и еще как минимум 2 параметрам.
3. Форма с личными данными отображает данные пользователя (администратора) и позволяет их изменять (после выбора соответствующего режима). Измененные данные после нажатия кнопки «принять изменения» записываются в файл. Есть кнопка для перехода на страницу работы с базой.
4. Форма работы с базой предполагает возможность просмотреть полный список (общий или по критериям). Можно осуществлять сортировку по различным критериям и поиск по названию, по дате, цене (зарплате) и еще как минимум 2 параметрам. Можно добавлять новые объекты (товар) в список.
5. Если залогинился бухгалтер, то кроме всего прочего у него есть возможность изменить цену или зарплату. Он также может просмотреть список проданных товаров (уволенных сотрудников).
6. Отдел кадров может добавлять новых сотрудников в список или увольнять их. И также может просмотреть список уволенных сотрудников. (Если у вас товар, то его добавлением или продажей будет заведовать отдел продаж.)
7. Если в систему вошел начальник, он может менять любые параметры, а также он может добавлять новые объекты (новых сотрудников) в список или продать любой объект (или уволить сотрудника). Может просмотреть список проданных товаров (уволенных сотрудников).

Для выполнения этой работы понадобится

* новый класс – администратор. У администратора есть имя, должность, пароль и еще несколько полей по своему усмотрению. Должность должна быть типа перечисление. В зависимости от должности будут открываться дополнительные возможности по управлению базой.
* новый класс, который будет содержать в себе массив родительского класса, в котором будут храниться все объекты. В этом классе надо реализовать все функции по управлению базой – поиск, сортировки, выборки, добавление новых объектов и перевод их в статус проданных (уволенных).
* новый класс – журнал. В нем будут храниться записи о всех действиях с базой кроме сортировок. (этот класс и работа с ним будет описана в лабораторной работе «Делегаты и события».

1.2 В качестве отчета предоставить работающее программное обеспечение.

1.3 Защитить лабораторную работу.

**Лабораторная работа №6**

**Делегаты и события.**

**Цель работы:** научиться разрабатывать программы с использованием делегатов и событий.

**1 Порядок выполнения работы**

1.1 В программе, разработанной в лабораторной работе № 12 требуется определить класс, содержащий типизированную коллекцию, 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.

Для этого определить класс Journal, который можно использовать для накопления информации об изменениях в коллекциях. Класс Journal хранит информацию в списке объектов. Каждый элемент списка содержит информацию об отдельном изменении, которое произошло в коллекции.

1.2 В качестве отчета предоставить работающее программное обеспечение.

1.3 Защитить лабораторную работу.

**2 Контрольные вопросы**

1. Назовите этапы создания и применения делегатов.
2. Как осуществляется вызов методов с помощью делегата?
3. В чём заключается механизм события?
4. Каков порядок создания пользовательского события?
5. Как используются методы класса в роли обработчика события?
6. Какой синтаксис должны иметь .NET-совместимые обработчики событий?