**Лабораторная работа № 5**

**Тема:** Разработка спецификаций структурных единиц класса. Организация иерархии наследования.

**Цель работы:** получить практические навыки разработки иерархии классов, методов, свойств классов.

*Указания:*

Требуется описать базовый класс и 1-3 наследованных класса. Каждый разрабатываемый класс должен находиться в **отдельном модуле** (файле) в составе проекта. Для добавления нового модуля в проект выберите пункт меню   
**«Проект» → «Добавить класс»** и укажите название класса.

Класс должен содержать: поля, конструкторы, свойства, методы (или переопределенные методы).

*Разработка структур данных должна производиться по следующему плану:*

1. Объявите базовый класс и опишите его поля. **Поля** должны быть скрыты (уровень доступа private или protected).
2. Опишите конструктор базового класса (конструкторов может быть и несколько, если требуется создавать объекты разными способами).
3. Опишите методы базового класса. **Методы,** которые представляют интерфейсную часть класса, должны быть открыты (уровень доступа public), так как вызов их будет происходить из другого класса (класса Program, где находится главная функция).
4. Продумайте, какие методы должны быть виртуальными и будут переопределены в дочерних классах. Опишите эти методы, пометив их словом virtual.
5. По условию задачи необходимо будет получать или задавать значения некоторых полей. Опишите **свойства**, связанные с этими полями, доступные для чтения (get) и (или) для установки (set). Свойства, также как и методы, будут иметь публичный доступ. Это позволит получать или устанавливать значение скрытого поля из основной программы.
6. Объявите **дочерний класс** и опишите его дополнительные поля. Дочерний класс наследует поля и методы родительского класса, поэтому те же поля, что и у родительского класса, объявлять в дочернем классе не нужно.
7. Опишите конструктор дочернего класса (т.к. у каждого класса должен быть свой конструктор).
8. Опишите новые и переопределенные методы дочернего класса. Переопределённые методы пометьте служебным словом override.
9. Для проверки функционала описанных классов **в главной функции** создайте **по одному объекту каждого класса** и покажите вызов методов. Убедитесь, что переопределенные методы вызываются правильно, в соответствие с типом объекта, к которому применяются.

**Пример описания классов:**

namespace Monsters

{

class Monster // базовый класс Монстр

{

private string name; // поля класса Монстр

private int power;

private int health;

protected static Random rnd = new Random();

public Monster() // конструктор по умолчанию

{

this.name = "Noname";

this.power = 100;

this.health = 100;

}

public Monster(string name) // конструктор с одним параметром

{

this.name = name;

this.power = rnd.Next(1, 200);

this.health = rnd.Next(1, 200);

}

public Monster(string name, int power, int health) // конструктор

{

this.name = name;

this.power = power;

this.health = health;

}

public virtual void Show()// метод для вывода информации на экран

{

if (this is Demon)

Console.Write("\tДемон: {0}\t", name);

if (this is Monster)

Console.Write("\tМонстр: {0}\t", name);

Console.Write("Сила: {0}\tЗдоровье: {1}", power, health);

}

}

class Demon : Monster // порождённый класс Демон

{

private int brain; // поле класса Демон, он дополнительно имеет ум

public Demon(string name, int power, int health, int brain): base(name, power, health) // конструктор для Демона

{

this.brain = brain;

}

public override void Show() // переопределенный метод для вывода информации о Демоне

{

base.Show();

Console.Write("\tУм: {0}", brain);

}

public int Brain // свойство для проверки ума

{

get

{

return brain;

}

}

}

class Program

{

static void Main()

{

…// здесь создание объектов и вызов методов

}

}

}

**ВАРИАНТЫ:**

**Вариант 1**

Создать абстрактный класс Transport (транспортное средство) и на его основе два производных класса Plain (самолёт) и Ship (корабль). Классы должны иметь возможность получать параметры (название, количество мест, скорость, год выпуска, для самолёта – высота, для корабля – порт приписки (перечисляемый тип)), а также иметь метод вывода на экран сведений о транспортном средстве. Создать по одному объекту каждого класса и показать вызов методов для них.

**Вариант 2**

Создать абстрактный класс Point (точка), содержащий поля X, Y с координатами точки, и на его основе класс ColoredPoint (цветная точка) с дополнительным полем Color (цвет) и Line (линия) с дополнительными полями Length (длина линии) и Ugol (угол наклона линии к оси X) и дочерний класс Circle (окружность), причем окружность характеризуется точкой своего центра и радиусом. Описать методы для вывода на экран сведений об объекте, получения и установки координат точки, а также для изменения цвета и получения текущего цвета и методы прорисовки всех видов объектов в графическом режиме.

**Вариант 3**

Создать базовый класс Plain (самолёт), содержащий поля: тип самолёта, количество пассажиров. Создать производный класс Fight (рейс), содержащий дополнительные поля: номер рейса, название пункта назначения. Описать методы для вывода на экран сведений об объекте, получения и установки значений некоторых полей. Описать класс Airport (аэропорт), содержащий включаемое поле массив рейсов. Создать метод добавления рейса, метод удаления рейса, а также метод вывода всего списка на экран. В основной программе продемонстрировать создание объекта аэропорт и работу со списком рейсов.

**Вариант 4**

Создать базовый класс Woker (работник), содержащий поля: ФИО работника, должность (перечисляемый тип), год поступления на работу. Создать производный класс Teacher (преподаватель) с дополнительными полями нагрузка (то есть количество часов на учебный год) и категория. Описать методы для вывода на экран сведений об объекте, получения и установки значений некоторых полей. Организовать расчёт заработной платы: для простых работников с учётом 8-ми часового рабочего дня и почасовой ставки для конкретной должности, для преподавателей – с учётом нагрузки и категории. Описать класс Personnel (персонал) с включенным полем массив работников, и методами добавления и удаления работников в массив объектов, а также вывода на экран списка персонала и зарплаты каждого работника.

**Вариант 5**

Создать базовый класс Tovar (товар), содержащий поля: код товара, название, цена, количество на складе. Создать конструктор класса, метод для вывода на экран сведений об объекте товар, а также методы для увеличения количества товара на 1 и уменьшения количества товара на 1.

Создать производный класс Product (продукт) c дополнительным полем срок реализации продукта (то есть сколько суток он годен). Создать конструктор класса, метод для вывода на экран сведений об объекте.

Описать класс Order (заказ), у которого имеются следующие поля: номер заказа, дата и время заказа, статус заказа (формируется или готов), а также включаемое поле массив товаров с указанием количества каждой позиции. Создать метод добавления товара в заказ (товары выбираются из заранее созданного списка имеющихся товаров), метод увеличения или уменьшения количества данного товара в заказе на 1, метод удаления товара из заказа, а также метод окончательного формирования заказа с расчётом полной стоимости заказа, и метод вывода заказа на экран. В основной программе продемонстрировать создание заказа.

**Вариант 6**

Создать базовый класс Note (запись), содержащий поля: ФИО человека, номер телефона. Создать производный класс Friend (друг), содержащий дополнительные поля: электронная почта, дата рождения. Описать методы для вывода на экран записей, получения и установки значений некоторых полей. Описать класс AdressBook (телефонная книга), содержащий включаемое поле массив записей Note. Создать метод добавления записи в телефонную книгу, метод удаления записи из телефонной книги, а также метод вывода всего списка на экран. В основной программе продемонстрировать создание телефонной книги.

**Вариант 7**

Создать базовый класс Circle (круг), содержащий поле: радиус. Создать производные классы Cone (конус) и Cylinder (цилиндр), содержащие дополнительное поле: высота. Описать методы для вывода на экран сведений об объекте, получения и установки значений полей, методы вычисления площади поверхности и объёма фигур для производных классов. Создать по одному объекту каждого класса и показать вызов методов для них.

**Вариант 8**

Создать базовый класс Rectangle (прямоугольник), содержащий поля: длина, ширина. Создать производные классы Parallelogram (параллелограмм) и Trapeze (равнобедренная трапеция), содержащие дополнительное поле: угол наклона. Описать методы для вывода на экран сведений об объекте, получения и установки значений полей, методы вычисления площади фигур. Создать по одному объекту каждого класса и показать вызов методов для них.

**Вариант 9**

Создать базовый класс Point (точка), содержащий поля: x, y (координаты точки). Создать производный класс Circle (круг), содержащий дополнительное поле: радиус, а от класса Circle производный класс Ellipse (эллипс), содержащий ещё одно дополнительное поле: второй радиус. Описать методы для вывода на экран сведений об объекте, получения и установки значений полей, методы вычисления площади фигур. Создать по одному объекту каждого класса и показать вызов методов для них.

**Вариант 10**

Создать базовый класс Rectangle (прямоугольник), содержащий поля: длина, ширина. Создать производные классы Parallelepiped (параллелепипед) и Pyramid (пирамида), содержащие дополнительное поле: высота. Описать методы для вывода на экран сведений об объекте, получения и установки значений полей, методы вычисления площади поверхности и объёма фигур. Создать по одному объекту каждого класса и показать вызов методов для них.

**Вариант 11**

Создать базовый класс Patch (путь), содержащий поля: пункт назначения, длина пути (в км). Создать производный класс Marsh (туристический маршрут), содержащий дополнительные поля: название маршрута (перечисляемый тип), уровень сложности, количество дней в пути. Описать методы для вывода на экран сведений о маршрутах, получения и установки значений некоторых полей. Создать по два объекта каждого класса и показать вызов методов для них.

**Вариант 12**

Создать базовый класс Note (запись), содержащий поля: ФИО человека, номер телефона. Создать производный класс Woker (сотрудник), содержащий дополнительные поля: электронная почта, должность (перечисляемый тип). Описать методы для вывода на экран записей, получения и установки значений некоторых полей. Создать по одному объекту каждого класса и показать вызов методов для них.

**Вариант 13**

Создать базовый класс Book (книга), содержащий поля: ФИО автора, название, год издания (перечисляемый тип), категория. Создать производный класс ElectronBook (электронная книга), содержащий дополнительное поле: электронный источник. Описать методы для вывода на экран сведений о книгах, получения и установки значений некоторых полей. Описать класс Library (библиотека), содержащий включаемое поле массив книг. Создать метод добавления книги, метод удаления книги, а также метод вывода всего списка на экран. В основной программе продемонстрировать создание объекта библиотека и работу со списком книг.

**Вариант 14**

Создать абстрактный класс Person (человек), содержащий поля: ФИО человека, год рождения. Создать производные классы Doctor (доктор) и Patient (пациент), содержащие дополнительные поля: для обоих производных классов отделения больницы (перечисляемый тип), для докторов специализация, для пациентов дата поступления в больницу и дата выписки (если пациент выписался из больницы). Описать методы для вывода на экран сведений о врачах и пациентах больницы, получения и установки значений некоторых свойств. Создать по одному объекту каждого класса и показать вызов методов для них.

**Вариант 15**

Создать базовый класс Triangle (равносторонний треугольник), содержащий поле: сторона треугольника. Создать производные классы Tetrahedron (тетраэдр), Octahedron (октаэдр), Icosuhedron (икосаэдр). Описать методы для вывода на экран названия правильного многогранника и сведений об объекте, получения значений полей, методы вычисления площади поверхности фигур. Создать по одному объекту каждого класса и показать вызов методов для них.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Многогранник** | | **Вершины** | **Рёбра** | **Грани** |
| тетраэдр | [Тетраэдр](https://commons.wikimedia.org/wiki/File:Tetrahedron.svg?uselang=ru) | 4 | 6 | 4 |
| [октаэдр](https://ru.wikipedia.org/wiki/%D0%9E%D0%BA%D1%82%D0%B0%D1%8D%D0%B4%D1%80) | [Октаэдр](https://commons.wikimedia.org/wiki/File:Octahedron.svg?uselang=ru) | 6 | 12 | 8 |
| [икосаэдр](https://ru.wikipedia.org/wiki/%D0%98%D0%BA%D0%BE%D1%81%D0%B0%D1%8D%D0%B4%D1%80) | [Икосаэдр](https://commons.wikimedia.org/wiki/File:Icosahedron.svg?uselang=ru) | 12 | 30 | 20 |

**Вариант 16**

Создать абстрактный класс Car (автомобиль), содержащий поля: госномер, марка (перечисляемый тип), цвет. Создать класс Avtostoyanka (автостоянка), содержащий поля: фамилия владельца, автомобиль (включаемое поле), признак присутствия автомобиля на стоянке. Описать методы для вывода на экран сведений об автомобилях, получения и установки значений некоторых полей. Создать по два объекта каждого класса и показать вызов методов для них.

**Вариант 17**

Создать базовый класс BaseString (простая строка), содержащий поля: поле для хранения символов строки, длина строки. В базовом классе описать методы: конструктор без параметров, конструктор, принимающий в качестве параметра строку, метод получения длины строки, проверка на равенство двух строк, метод очистки строки (сделать строку пустой). Создать производный класс OctString (восьмеричная строка). Строки данного класса могут содержать только символы “01234567”. Если в составе инициализирующей строки встретятся другие символы, то строка принимает пустое значение. В производном классе описать методы: конструктор, принимающий в качестве параметра строку, перевод восьмеричного числа в десятичную систему счисления. Создать по одному объекту каждого класса и показать вызов методов для них.

**Вариант 18**

Создать базовый класс BaseString (простая строка), содержащий поля: поле для хранения символов строки, длина строки. В базовом классе описать методы: конструктор без параметров, конструктор, принимающий в качестве параметра строку, метод получения длины строки, проверка на равенство двух строк, метод очистки строки (сделать строку пустой). Создать производный класс BinString (двоичная строка). Строки данного класса могут содержать только символы “01”. Если в составе инициализирующей строки встретятся другие символы, то строка принимает пустое значение. В производном классе описать методы: конструктор, принимающий в качестве параметра строку, перевод двоичного числа в десятичную систему счисления. Создать по одному объекту каждого класса и показать вызов методов для них.

**Вариант 19**

Создать базовый класс BaseString (простая строка), содержащий поля: поле для хранения символов строки, длина строки. В базовом классе описать методы: конструктор без параметров, конструктор, принимающий в качестве параметра строку, метод получения длины строки, метод очистки строки (сделать строку пустой). Создать производный класс ComplexString (комплексная строка). Строки данного класса состоят из двух частей, разделённых знаком «i». Первая половина строки задаёт значение действительной части числа, вторая половина после знака «i» – значение мнимой части. Каждая часть инициализирующей строки может содержать десятичные цифры и знак + или –, задающие знак числа, только в первой позиции строки. Если в составе инициализирующей строки встретятся другие символы, то строка принимает пустое значение.

В производном классе описать методы: дополнительные поля, хранящие действительную и мнимую часть числа в виде целочисленных значений, конструктор, принимающий строку, конструктор, принимающий на вход два целочисленных значения (действительную и мнимую часть числа в целом виде), проверка на равенство двух комплексных чисел, сложение, вычитание, умножение комплексных чисел. Создать по одному объекту каждого класса и показать вызов методов для них.

**Вариант 20**

Создать базовый класс Simetric (правильный многоугольник), содержащий поля: количество вершин, сторона многоугольника. Создать производные классы Tangle (равносторонний треугольник), Kvadrat (квадрат), FiveSimetric (правильный пятиугольник), SixSimetric (правильный шестиугольник). Описать методы для вывода на экран названия фигуры, сведений об объекте, методы вычисления площади фигур, радиуса описанной и вписанной окружности. Создать по одному объекту каждого класса и показать вызов методов для них.

Площадь правильного многоугольника с числом сторон n и длиной стороны a : S = \frac{n}{4}\ a^2 \mathop{\mathrm{}}\, \operatorname{ctg} \frac{\pi}{n}

Длина стороны многоугольника через радиус описанной и вписанной окружности: a =  2R \sin \frac{\pi}{n} =  2r \mathop{\mathrm{tg}}\, \frac{\pi}{n}