Лабораторная работа 3. Полиморфизм.

Программирование полиморфных методов. Перегрузка операций

**Цель работы:**

Познакомиться с программированием полиморфных методов и перегрузкой операций в объектно-ориентированном программировании средствами языка C#.

**Необходимые теоретические сведения**

**Полиморфизм**

Полиморфизм – одна из основных составляющих объектно-ориентированного программирования, позволяющая определять в наследуемом классе методы, которые будут общими для всех наследующих классов, при этом наследующий класс может определять специфическую реализацию некоторых или всех этих методов. Главный принцип полиморфизма: «один интерфейс, несколько методов». Благодаря ему, можно пользоваться методами, не обладая точными знаниями о типе объектов. Основным инструментом для реализации принципа полиморфизма является использование виртуальных методы и абстрактных классов.

**Виртуальные методы**

Метод, при определении которого в наследуемом классе было указано ключевое слово virtual, и который был переопределен в одном или более наследующих классах, называется виртуальным методом. Следовательно, каждый наследующий класс может иметь собственную версию виртуального метода.

Выбор версии виртуального метода, которую требуется вызвать, осуществляется в соответствии с типом объекта, на который ссылается ссылочная переменная, во время выполнения программы. Другими словами, именно тип объекта, на который указывает ссылка (а не тип ссылочной переменной), определяет вызываемую версию виртуального метода. Таким образом, если класс содержит виртуальный метод и от этого класса были наследованы другие классы, в которых определены свои версии метода, при ссылке переменной типа наследуемого класса на различные типы объектов вызываются различные версии виртуального метода.

При определении виртуального метода в составе наследуемого класса перед типом возвращаемого значения указывается ключевое слово virtual, а при переопределении виртуального метода в наследующем классе используется модификатор override. Виртуальный метод не может быть определен с модификатором static или abstract.

Переопределять виртуальный метод не обязательно. Если наследующий класс не предоставляет собственную версию виртуального метода, то используется метод наследуемого класса.

Переопределение метода положено в основу концепции динамического выбора вызываемого метода - выбора вызываемого переопределенного метода осуществляется во время выполнения программы, а не во время компиляции.

Синтаксис:

virtual тип имя (список\_параметров){тело\_метода};

**Абстрактные классы**

В абстрактном классе определяются лишь общие предназначения методов, которые должны быть реализованы в наследующих классах, но сам по себе этот класс не реализует один, или несколько подобных методов, называемых абстрактными (для них определены только некоторые характеристики, такие как тип возвращаемого значения, имя и список параметров).

При объявлении абстрактного метода используется модификатор abstract. Абстрактный метод автоматически становится виртуальным, так что модификатор virtual при объявлении метода не используется.

Абстрактный класс предназначен только для создания иерархии классов, нельзя создать объект абстрактного класса.

Пример:

abstract class Animal

{

public string Name;

protected int Weight;

private int Type;

abstract void Feed();

public int Animal(int W, int T, string N)

{

Weight=W;

Type=T;

Name=N;

}

public int GetWeight(){return Weight;}

}

class Predator:Animal

{

private int Speed;

override void Feed(int Food){

Weight += Food;}

}

Контрольные вопросы

1) Что понимается под термином «полиморфизм»?

2) В чем состоит основной принцип полиморфизма?

3) В чем состоит значение основного принципа полиморфизма?

4) Какие механизмы используются в языке C# для реализации концепции

полиморфизма?

5) Что понимается под термином «виртуальный метод»?

6) Какое ключевое слово языка C# используется для определения виртуального метода?

7) В чем состоит особенность виртуальных методов в производных (дочерних) классах?

8) В какой момент трансляции программы осуществляется выбор версии виртуального метода?

9) Какие условия определяют выбор версии виртуального метода?

10) Какое ключевое слово (модификатор) языка C# используется для определения виртуального метода в базовом (родительском) классе?

11) Какое ключевое слово (модификатор) языка C# используется для определения виртуального метода в производном (дочернем) классе?

12) Какие модификаторы недопустимы для определения виртуальных методов?

13) Что означает термин «переопределенный метод»?

14) В какой момент трансляции программы осуществляется выбор вызываемого переопределенного метода?

15) Приведите синтаксис виртуального метода в общем виде. Проиллюстрируйте его фрагментом программы на языке C#.

16) Что понимается под термином «абстрактный класс»?

17) В чем заключаются особенности абстрактных классов?

18) Какой модификатор языка C# используется при объявлении абстрактных методов?

19) Являются ли абстрактные методы виртуальными?

20) Используется ли модификатор virtual языка C# при объявлении абстрактных методов?

21) Возможно ли создание иерархии классов посредством абстрактного класса?

22) Возможно ли создание объектов абстрактного класса?

23) Приведите синтаксис абстрактного класса в общем виде. Проиллюстрируйте его фрагментом программы на языке C#.

**Задание 1**

Расширить иерархию классов из лабораторной работы №2 с использованием виртуального класса в качестве основы иерархии. Показать пример

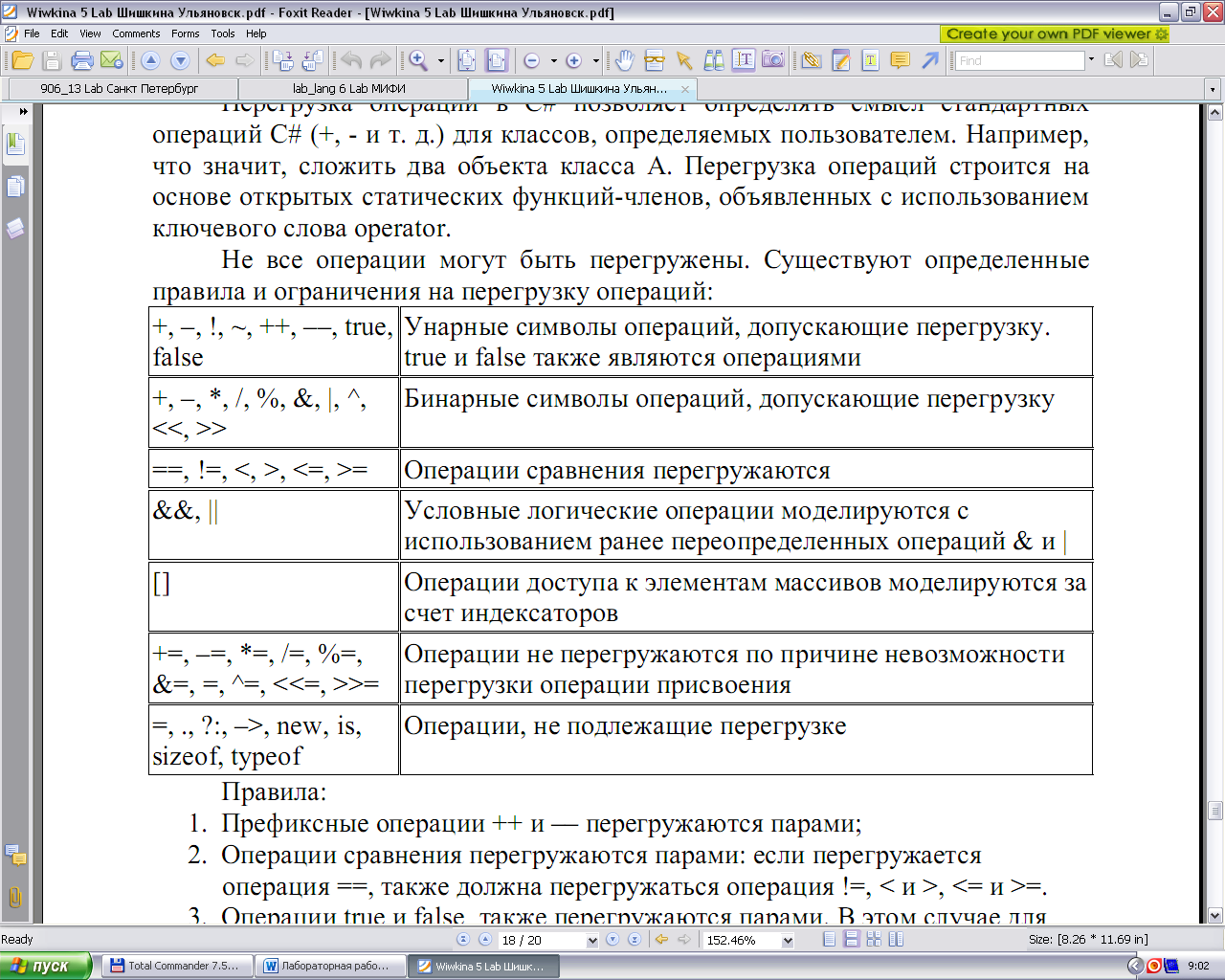
использования полиморфизма методов.

**Переопределение операций**

**Теоретические основы**

Перегрузка операций в C# позволяет определять смысл стандартных операций C# (+, - и т. д.) для классов, определяемых пользователем. Например, что значит, сложить два объекта класса А. Перегрузка операций строится на основе открытых статических функций-членов, объявленных с использованием ключевого слова operator.

Не все операции могут быть перегружены. Существуют определенные правила и ограничения на перегрузку операций:



**Правила:**

1. Префиксные операции ++ и –– перегружаются парами;

2. Операции сравнения перегружаются парами: если перегружается операция ==, также должна перегружаться операция !=, < и >, <= и >=.

3. Операции true и false также перегружаются парами. В этом случае для объекта класса определяются критерии истинности. Необходимо следить, чтобы критерии истинности, определенные в операции true и в операции false, не противоречили друг другу.

Синтаксис:

public static <тип возвращаемого значения> operator <операция>(<параметры>)

Пример

class Program

{

public static Program operator ++(Program par1)

{

par1.x++;

return par1;

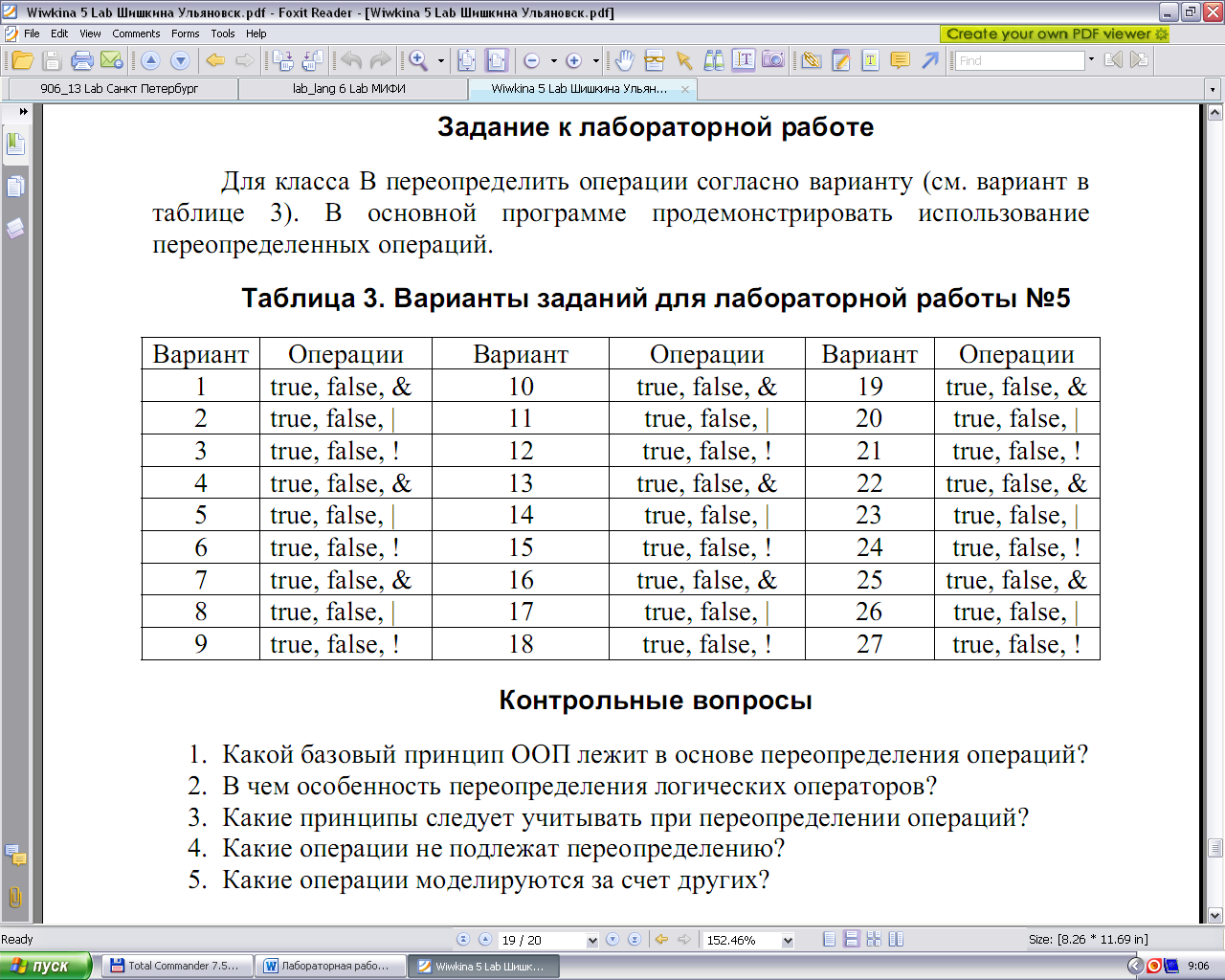
}

}

**Задание к лабораторной работе**

Для класса В переопределить операции согласно варианту (см. вариант в таблице ). В основной программе продемонстрировать использование переопределенных операций.

Таблица. Варианты заданий для лабораторной работы



**Контрольные вопросы**

1. Какой базовый принцип ООП лежит в основе переопределения операций?

2. В чем особенность переопределения логических операторов?

3. Какие принципы следует учитывать при переопределении операций?

4. Какие операции не подлежат переопределению?

**Порядок выполнения работы:**

1) Изменить иерархию классов и реализовать ее на С#.

2) Показать на примере одного из методов, присутствующих в каждом классе, свойство полиморфизма.

3) Подготовить отчет в твердой копии и в электронном виде.