ЛАБОРАТОРНАЯ РАБОТА 11. ОПЕРАЦИИ КЛАССОВ. ПЕРЕГРУЗКА ОПЕРАЦИЙ.

1. Цель и содержание

Цель лабораторной работы: научиться осуществлять перегрузку операторов относительно пользовательских типов.

Задачи лабораторной работы:

– изучить операции, подлежащие перегрузке;

– получить практические навыки перегрузки операторов.

2. Формируемые компетенции

Лабораторная работа направлена на формирование следующих компетенций:

– способность к проектированию базовых и прикладных информационных технологий (ПК-11);

– способность разрабатывать средства реализации информационных технологий (методические, информационные, математические, алгоритмические, технические и программные) (ПК-12).

3. Теоретическая часть

2.1 Операции. Большинство операций пришло в C# из языков C, C++:

Категория	Операции
Арифметические	+, -, *, /, %
Логические	&, |, ^, ~, &&, ||, !
Конкатенация строк	+
Инкремент и декремент	++, --
Сравнение	<, >, >=, >=, ==, !=
Присвоение	+=, -=, /=, *=, %=, =, &=, |=, ^=, <<=, >>=

Доступ к члену класса	.
Индексация	[ ]
Приведение	( )
Условная (тернарная) операция	? :
Создание объектов	new
Информация о типе	sizeof, is, as, typeof,
Контроль исключения, связанного с переполнением	checked, unchecked

2.2 Перегрузка операций относительно класса. Рассмотрим простой класс, например, для представления объекта «Вектор» (для простоты возьмем вектор на плоскости). Пример 1:

```
class Vector
{
    // Поля данных представляют стороны параллелепипеда
    public double X, Y;

    // Конструкторы
    public Vector()
    {

    }

    public Vector(double XCoord, double YCoord)
    {
        X = XCoord;
        Y = YCoord;
    }

}
```

Класс создан, но в математике можно выполнять сложение (вычитание) векторов и умножение вектора на число, а также находить скалярное произведение векторов. Необходимо добавить эту возможность для нашего класса «Вектор». Это значит, что операции, указанные в примере 2 должны иметь смысл.

Пример 2:

```
Vector a = new Vector(4, 10);
Vector b = new Vector(10, 15);

// вычисление суммы векторов
Vector SumVector = a + b;

// Вычисление произведения вектора на число
Vector MultNum = 2 * a;

// вычисление скалярного произведения
double Skalar = a * b;
```

В данной реализации (пример 1) такие операции в коде невозможны. Добавим в класс перегруженные операции.

Пример 3:

```
class Vector
{
    // Поля данных представляют стороны параллелепипеда
    public double X, Y;

    // Конструкторы
    public Vector()
    {

    }

    public Vector(double XCoord, double YCoord)
    {
        X = XCoord;
        Y = YCoord;
    }

    // Перегрузка операции сложения
    public static Vector operator +(Vector left, Vector right)
    {
        return new Vector(left.X + right.X, left.Y + right.Y);
    }

    // Перегрузка операции умножения вектора на число
    public static Vector operator *(double k, Vector vek)
    {
        return new Vector(k*vek.X, k*vek.Y);
    }

    // Перегрузка операции умножения двух векторов (скалярное умножение)
    public static double operator *(Vector left, Vector right)
    {
        return left.X * right.X + left.Y * right.Y;
    }

}
```

Перегрузка операции похожа на объявление метода класса, но существуют отличия: используется ключевое слово static; используется ключевое слово operator и символ операции (+, *, - и т.д.) вместо имени метода.

После того как операции перегружены, возможно выполять следующие действия:

```
class Program
{
    static void Main(string[] args)
    {
        Vector a = new Vector(4, 10);
        Vector b = new Vector(10, 15);

        // вычисление суммы векторов
        Vector SumVector = a + b;

        // Вычисление произведения вектора на число
        Vector MultNum = 2 * a;

        // вычисление скалярного произведения
        double Skalar = a * b;

        // Проведение сложных расчетов
        Vector RES = 2 * a + 3 * b + (a * b) * a;


        Console.ReadKey();
    }
}
```

4. Оборудование и материалы

Для выполнения лабораторной работы рекомендуется использовать персональный компьютер со следующими характеристиками: 64-разрядный (x64) процессор с тактовой частотой 1 ГГц и выше, оперативная память – 1 Гб и выше, свободное дисковое пространство – не менее 1 Гб, графическое устройство DirectX 9. Программное обеспечение: операционная система WINDOWS 7 и выше, Microsoft Visual Studio 20112 и выше.

5. Указания по технике безопасности

Техника безопасности при выполнении лабораторной работы определяется общепринятой для пользователей персональных компьютеров. Самостоятельно не производить ремонт персонального компьютера, установку

и удаление программного обеспечения; в случае неисправности персонального компьютера сообщить об этом обслуживающему персоналу лаборатории; не касаться электрических розеток металлическими предметами; рабочее место пользователя персонального компьютера должно содержаться в чистоте; не разрешается возле персонального компьютера принимать пищу, напитки.

6. Методика и порядок выполнения работы

1. Создайте консольное приложение.

2. Совместно с преподавателем спроектируйте и разработайте класс, относительно которого перегрузите операции +, - /, *, %, ==, > (каждый студент разрабатывает свой класс).

Индивидуальные задания (возможные варианты классов):

«Вектор в пространстве», «Матрица», «Сотрудник», «Компьютер», «Товар», «Объект недвижимости», «Комплексное число», «Куб», «Автомобиль», «Студент», «Книга», «Дисциплина (предмет)».

Возможно предложение своего класса или доработка класса из лабораторной работы №13.

7. Содержание отчета и его форма

Отчет по лабораторной работе должен содержать:

1. Номер и название лабораторной работы.

2. Цели лабораторной работы.

3. Ответы на контрольные вопросы.

4. Экранные формы и листинг программного кода, показывающие порядок выполнения лабораторной работы, и результаты, полученные в ходе её выполнения.

Отчет о выполнении лабораторной работы в письменном виде сдается преподавателю.

8. Контрольные вопросы

1. Что такое конструктор?

2. Что такое перегрузка операторов? Когда применяется данный механизм?

3. Допустим, что для класса создана перегруженная операция сложения. Может ли быть создана еще одна операция сложения для данного класса?

4. Какие операции нельзя перегрузить?

5. Какие операции необходимо перегружать попарно?

9. Список литературы

Для выполнения лабораторной работы, при подготовке к защите, а также для ответа на контрольные вопросы рекомендуется использовать следущие источники: [3], [5].

ЛАБОРАТОРНАЯ РАБОТА 12. ПОСТРОЕНИЕ ИЕРАРХИИ КЛАССОВ

1. Цель и содержание

Цель лабораторной работы: изучить механизм организации наследования классов.

Задачи лабораторной работы:

– научиться объявлять производные классы;

– научиться создавать иерархии классов;

– научиться использовать механизм полиморфизма.

2. Формируемые компетенции

Лабораторная работа направлена на формирование следующих компетенций:

– способность к проектированию базовых и прикладных информационных технологий (ПК-11);

– способность разрабатывать средства реализации информационных технологий (методические, информационные, математические, алгоритмические, технические и программные) (ПК-12).

3. Теоретическая часть

2.1 Наследование реализации. Наследование реализации (implementation inheritance) означает, что тип происходит от базового типа, получая от него все поля-члены и функции-члены.

Синтаксис наследования реализации: