Московский авиационный институт

(национальный исследовательский университет)

**Факультет информационных технологий и прикладной математики**

Кафедра вычислительной математики и программирования

**Лабораторная работа 2 по курсу ОOП:**

**основы программирования на языке С#**

АГРЕГАЦИЯ ПО ЗНАЧЕНИЮ И ВЛОЖЕНИЕМ

Работу выполнил:

Березнев Никита Вадимович

Группа: М8О-203Б-21

Подпись: \_\_\_\_\_\_\_\_

Дата: 29 октября 2022 г.

**2.1 Агрегация по значению**

**Агрегация** - отношение «часть-целое» между двумя равноправными объектами.

При **агрегации по** ссылке один класс имеет в качестве поля ссылку на другой класс, тем самым классы являются независимыми друг от друга, их можно менять, создавать, уничтожать отдельно друг от друга.



**Код программы:**

using System;

namespace CSharpLab2\_1

{

class Program

{

class A

{

private C c = new C();

private J j = new J();

public A()

{

c.c1 = 40;

}

~A() { }

public void mA()

{

Console.WriteLine("method of A");

}

public J jA

{

get { Console.Write("get j ->"); return j; }

}

public C cA

{

get { Console.Write("get c ->"); return c; }

}

}

class C

{

D d = new D();

E e = new E();

F f = new F();

public C() { this.c1 = 15; }

~C() { }

public void mC()

{

Console.WriteLine("method of C");

}

public D dA

{

get { Console.Write("get d ->"); return d; }

}

public E eA

{

get { Console.Write("get e ->"); return e; }

}

public F fA

{

get { Console.Write("get f ->"); return f; }

}

public int c1 { get; set; }

}

class J

{

K k = new K();

public J() { }

~J() { }

public void mJ()

{

Console.WriteLine("method of J");

}

public K kA

{

get { Console.Write("get k ->"); return k; }

}

}

//

class D

{

public D() { }

~D() { }

public void mD()

{

Console.WriteLine("Method of D");

}

}

class E

{

public E() { }

~E() { }

public void mE()

{

Console.WriteLine("Method of E");

}

}

class F

{

public F() { }

~F() { }

public void mF()

{

Console.WriteLine("Method of F");

}

}

class K

{

public K() { }

~K() { }

public void mK()

{

Console.WriteLine("Method of K");

}

}

static void Main(string[] args)

{

A a = new A();

a.mA();

a.cA.mC();

a.jA.mJ();

a.cA.dA.mD();

a.cA.eA.mE();

a.cA.fA.mF();

a.jA.kA.mK();

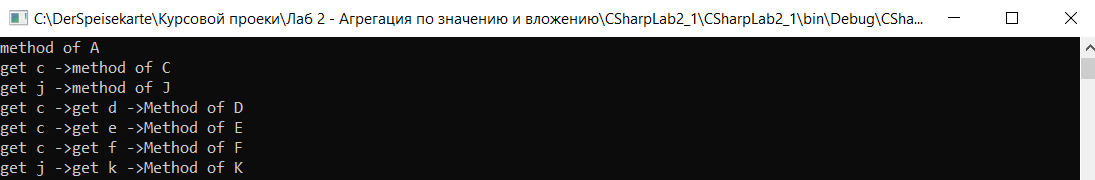
Console.ReadKey();

}

}

}

**Результат:**



**Вывод:**

При агрегации по значению все объекты класса существуют внутри объявленного класса. При таком виде агрегации невозможно удалить объекты, являющиеся частью объекта первого по иерархии класса. Например, *b, j —* части объекта *а* класса А(первый класс по иерархии); эти части создаются только при вызове конструктора класса A, а уничтожаются — при вызове деструктора А.

**2.2 Агрегация вложением**



**Код программы:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace CSharpLab2\_2

{

class Program

{

class A

{

public A() { j.j1 = 5; }

~A() { }

public class C

{

public C() { }

~C() { }

public class D

{

public D() { }

~D() { }

public void mD() { Console.WriteLine(" method of D"); }

}

public class E

{

public E() { }

~E() { }

public void mE() { Console.WriteLine(" method of E"); }

}

public class F

{

public F() { }

~F() { }

public void mF() { Console.WriteLine(" method of F"); }

}

public void mC() { Console.WriteLine(" method of C"); }

public D dA { get { Console.Write("get d -> "); return d; } }

private D d = new D();

public E eA { get { Console.Write("get e -> "); return e; } }

private E e = new E();

public F fA { get { Console.Write("get f -> "); return f; } }

private F f = new F();

}

public class J

{

public J() { this.j1 = 10; }

~J() { }

public class K

{

public K() { }

~K() { }

public void mK() { Console.WriteLine(" method of K"); }

}

public void mJ() { Console.WriteLine(" method of J"); }

public int j1 { set; get; }

}

public void mA() { Console.WriteLine(" method of A"); }

public C cA { get { Console.Write("get c ->"); return c; } }

public J jA { get { Console.Write("get j ->"); return j; } }

private C c = new C();

private J j = new J();

}

static void Main(string[] args)

{

A a = new A();

a.mA();

a.cA.mC();

a.jA.mJ();

a.cA.dA.mD();

a.cA.eA.mE();

a.cA.fA.mF();

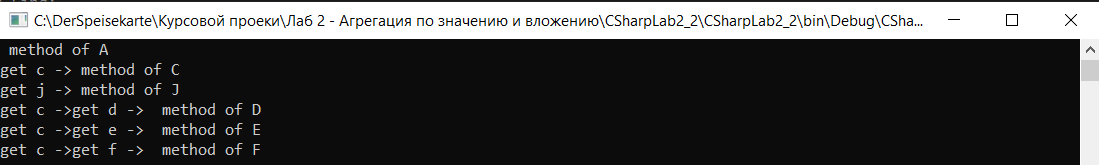
Console.ReadKey();

}

}

}

**Результат:**



**Вывод:**

При агрегации вложением определение классов происходит внутри классов, стоящих выше по иерархии. Все объекты создаваемого класса существуют внутри него самого. Как и в случае агрегации по значению уничтожение объектов, невозможно без уничтожения класса, стоящего выше по иерархии. Агрегация по ссылке намного лучше агрегации по значению или вложению, так как агрегация по ссылке расходует меньше памяти.