

Московский авиационный институт
(национальный исследовательский университет)

**Институт информационных технологий и
прикладной математики**

Кафедра вычислительной математики и программирования

**Лабораторная работа №2 по курсу ООП:
Агрегация по значению и вложением на C#**

Работу выполнила:

М8О-209Б-19 Офицерова Т.И.

Группа

ФИО

Подпись

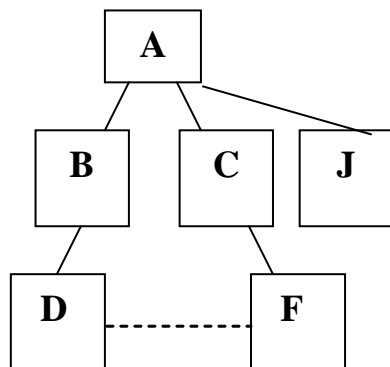
Вариант

Руководитель: _____/Кузнецова С.В./

Подпись:

Дата: 19 сентября 2020

Вариант графа



граф 10.

По значению

Текст программы

```
using System;

namespace lab2
{
    class A
    {
        private B b = new B();
        private C c = new C();
        private J j = new J();
        public A() //конструктор A
        {
            b.id = 33;
        }
        public void mA() //тест метод A
        {
            Console.WriteLine("method of A");
        }
        public B bA //свойство связи с объектом класса B
        {
            get
            {
                Console.Write("get b ->");
                return b;
            }
        }
        public C cA //свойство для связи с объектом класса C
        {
            get
            {
                Console.Write("get c ->");
                return c;
            }
        }
        public J jA //свойство для связи с объектом класса J
        {
            get
            {
                Console.Write("get j ->");
                return j;
            }
        }
    }
}
```

```

    }
}
class B
{
    private D d = new D();
    public int id;
    public B() { } //конструктор
    public void mB() //тест метод B
    {
        Console.WriteLine("method of B");
    }
    public D dA //свойство связи с объектом класса D
    {
        set
        {
            Console.WriteLine("set d");
            d = value;
        }
        get
        {
            Console.Write("get d -> ");
            return d;
        }
    }
}
class C
{
    private F f = new F();
    public C() { } //конструктор
    public void mC() //тест метод C
    {
        Console.WriteLine("method of C");
    }
    public F fA //свойство связи с объектом класса F
    {
        set
        {
            Console.WriteLine("set f");
            f = value;
        }
        get
        {
            Console.Write("get f -> ");
            return f;
        }
    }
}
class D
{
    public D() { } //пустой конструктор
    public void mD() //тестовый метод D
    {
        Console.WriteLine("Method of D");
    }
}
class F
{
    public F() { } //пустой конструктор
    public void mF() //тестовый метод F
    {
        Console.WriteLine("Method of F");
    }
}
class J
{

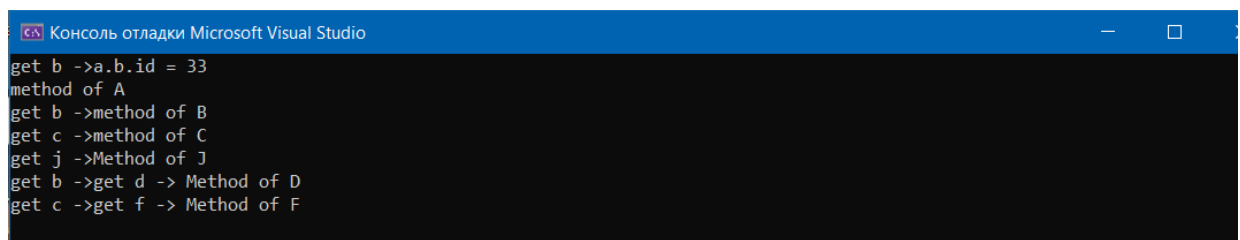
```

```

        public J() { } //пустой конструктор
        public void mJ() //тестовый метод J
        {
            Console.WriteLine("Method of J");
        }
    }
}
class Program
{
    static void Main(string[] args)
    {
        //Создание объекта класса A
        A a = new A();
        //обход вершин
        a.mA();
        a.bA.mB();
        a.cA.mC();
        a.jA.mJ();
        a.bA.dA.mD();
        a.cA.fA.mF();
    }
}
}

```

Результат работы



```

Консоль отладки Microsoft Visual Studio
get b -> a.b.id = 33
method of A
get b -> method of B
get c -> method of C
get j -> Method of J
get b -> get d -> Method of D
get c -> get f -> Method of F

```

Примеры

Человек и его сердце, компания и филиал, гитара и гриф

Вложением

Текст программы

```

using System;
using System.Collections.Generic;
using System.Text;

namespace lab2
{
    class A
    {
        private B b = new B();
        private C c = new C();
        private J j = new J();
        public A() //конструктор
        {
            this.b.id = 10;
        }
        public class B
        {
            private D d = new D();
            public int id;
            public B() { } //конструктор
        }
    }
}

```

```

public class D
{
    public D() { } //пустой конструктор
    public void mD() //тест метод D
    {
        Console.WriteLine("Method of D");
    }
} //end of D
public void mB() //тест метод B
{
    Console.WriteLine("Method of B");
}
public D dA //свойство связи с объектом класса D
{
    set
    {
        Console.WriteLine("set d");
        d = value;
    }
    get
    {
        Console.Write("get d -> ");
        return d;
    }
}
} //end of B
public class C
{
    private F f = new F();
    public C() { } //конструктор
    public class F
    {
        public F() { } //пустой конструктор
        public void mF() //тест метод F
        {
            Console.WriteLine("Method of F");
        }
    } //end of F
    public void mC() //тест метод C
    {
        Console.WriteLine("Method of C");
    }
    public F fA //свойство связи с объектом класса D
    {
        set
        {
            Console.WriteLine("set f");
            f = value;
        }
        get
        {
            Console.Write("get f -> ");
            return f;
        }
    }
} //end of C
public class J
{
    public J() { } //пустой конструктор
    public void mJ() //тест метод J
    {
        Console.WriteLine("Method of J");
    }
} //end of J
public void mA()

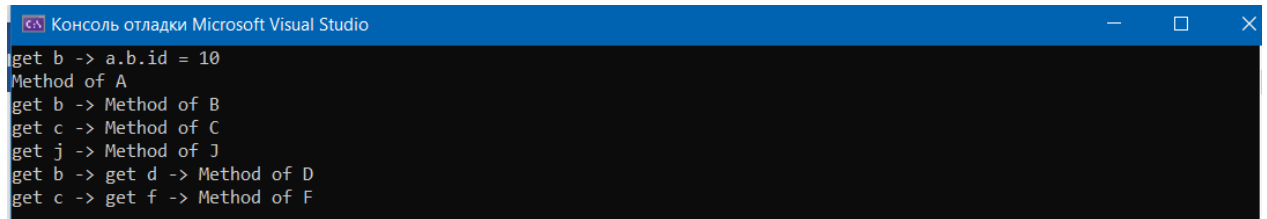
```

```

    {
        Console.WriteLine("Method of A");
    }
    public B bA//свойство связи с объектом класса B
    {
        set
        {
            Console.WriteLine("set b");
            b = value;
        }
        get
        {
            Console.Write("get b -> ");
            return b;
        }
    }
    public C cA//свойство связи с объектом класса C
    {
        set
        {
            Console.WriteLine("set c");
            c = value;
        }
        get
        {
            Console.Write("get c -> ");
            return c;
        }
    }
    public J jA//свойство связи с объектом класса D
    {
        set
        {
            Console.WriteLine("set j");
            j = value;
        }
        get
        {
            Console.Write("get j -> ");
            return j;
        }
    }
} //end of A
class Program
{
    static void Main(string[] args)
    {
        //Создание объекта класса A
        A a = new A();
        Console.WriteLine("a.b.id = {0}", a.bA.id);
        //обход вершин
        a.mA();
        a.bA.mB();
        a.cA.mC();
        a.jA.mJ();
        a.bA.dA.mD();
        a.cA.fA.mF();
    }
}

```

Результат работы

A screenshot of the Microsoft Visual Studio debug console window. The title bar is blue and contains the text 'Консоль отладки Microsoft Visual Studio' along with standard window control buttons. The console area has a black background with white text. It displays several lines of C# code and their corresponding outputs. The code includes property access like 'a.b.id', method calls like 'Method of A', and chained calls like 'get d -> Method of D'.

```
get b -> a.b.id = 10
Method of A
get b -> Method of B
get c -> Method of C
get j -> Method of J
get b -> get d -> Method of D
get c -> get f -> Method of F
```

Вывод

Агрегация по значению устанавливает между объектами более жесткую связь, чем агрегация по ссылке. Она используется в случаях, когда создание объекта-части не имеет смысла без создания объекта-целого, то есть объект-часть не существует отдельно от объекта-целого.

Агрегация вложением реализует самый жесткий тип связи. При ее использовании и объекты-части, и их классы определяются внутри объекта-целого.