Московский авиационный институт

(национальный исследовательский университет)

**Институт информационных технологий и прикладной математики**

Кафедра вычислительной математики и программирования

**Лабораторная работа №1 по курсу ОOП:**

**Агрегация по ссылке на С#**

Работу выполнила:

М8О-209Б-19 Офицерова Т.И. \_\_\_\_\_\_ \_\_\_\_\_\_\_\_

Группа ФИО Подпись Вариант

Руководитель: \_\_\_\_\_\_\_\_/Кузнецова С.В./

Подпись:

Дата:19 сентября 2020

**Вариант графа**



**Текст программы**

using Microsoft.VisualBasic;

using System;

namespace lab1

{

class A

{

private B b = null;

private C c = null;

private J j = null;

public A(B b, C c, J j)//констуктор А с 3мя аргументами

{

this.b = b;

this.c = c;

this.j = j;

this.b.id = 100;

}

public void mA() //тест метод А

{

Console.WriteLine("method of A");

}

public B bA //свойство для связи с объектом класса B

{

set

{

Console.WriteLine("set b");

b = value;

}

get

{

Console.Write("get b ->");

return b;

}

}

public C cA //свойство для связи с объектом класса C

{

set

{

Console.WriteLine("set c");

c = value;

}

get

{

Console.Write("get c ->");

return c;

}

}

public J jA //свойство для связи с объектом класса J

{

set

{

Console.WriteLine("set j");

j = value;

}

get

{

Console.Write("get j ->");

return j;

}

}

}

class B

{

private D d = null;

public B(D d) //конструктор B с 1 аргументом

{

this.d = d;

}

public void mB()//тест метод B

{

Console.WriteLine("method of B");

}

public int id;

public D dA //свойство для связи с объектом класса D

{

set

{

Console.WriteLine("set d");

d = value;

}

get

{

Console.Write("get d ->");

return d;

}

}

}

class C

{

private F f = null;

public C(F f) //конструктор С с 1 аргументом

{

this.f = f;

}

public void mC()//тест метод C

{

Console.WriteLine("method of C");

}

public F fA //свойство для связи с объектом класса А

{

set

{

Console.WriteLine("set f");

f = value;

}

get

{

Console.Write("get f ->");

return f;

}

}

}

class D

{

public D() { }//пустой конструктор D

public void mD()//тест метод D

{

Console.WriteLine(" method of D");

}

}

class J

{

public J() { }//пустой конструктор J

public void mJ()//тест метод J

{

Console.WriteLine(" method of J");

}

}

class F

{

public F() { }//пустой конструктор F

public void mF()//тест метод F

{

Console.WriteLine(" method of F");

}

}

class Program

{

static void Main(string[] args)

{

//создание объектов без подчиненных

D d = new D();

F f = new F();

J j = new J();

//создание объекта класса B

B b = new B(d);

b.dA = d;

b.id = 10;

//Создание объекта класса C

C c = new C(f);

// Создание объекта класса A

A a = new A(b, c, j);

//тест передачи b в a и изменения id

Console.WriteLine("b.id = {0}", b.id);

Console.WriteLine("a.b.id = {0}", a.bA.id);

//вызов метода класса A

a.mA();

//обход вершин

a.bA.mB();

a.cA.mC();

a.jA.mJ();

a.bA.dA.mD();

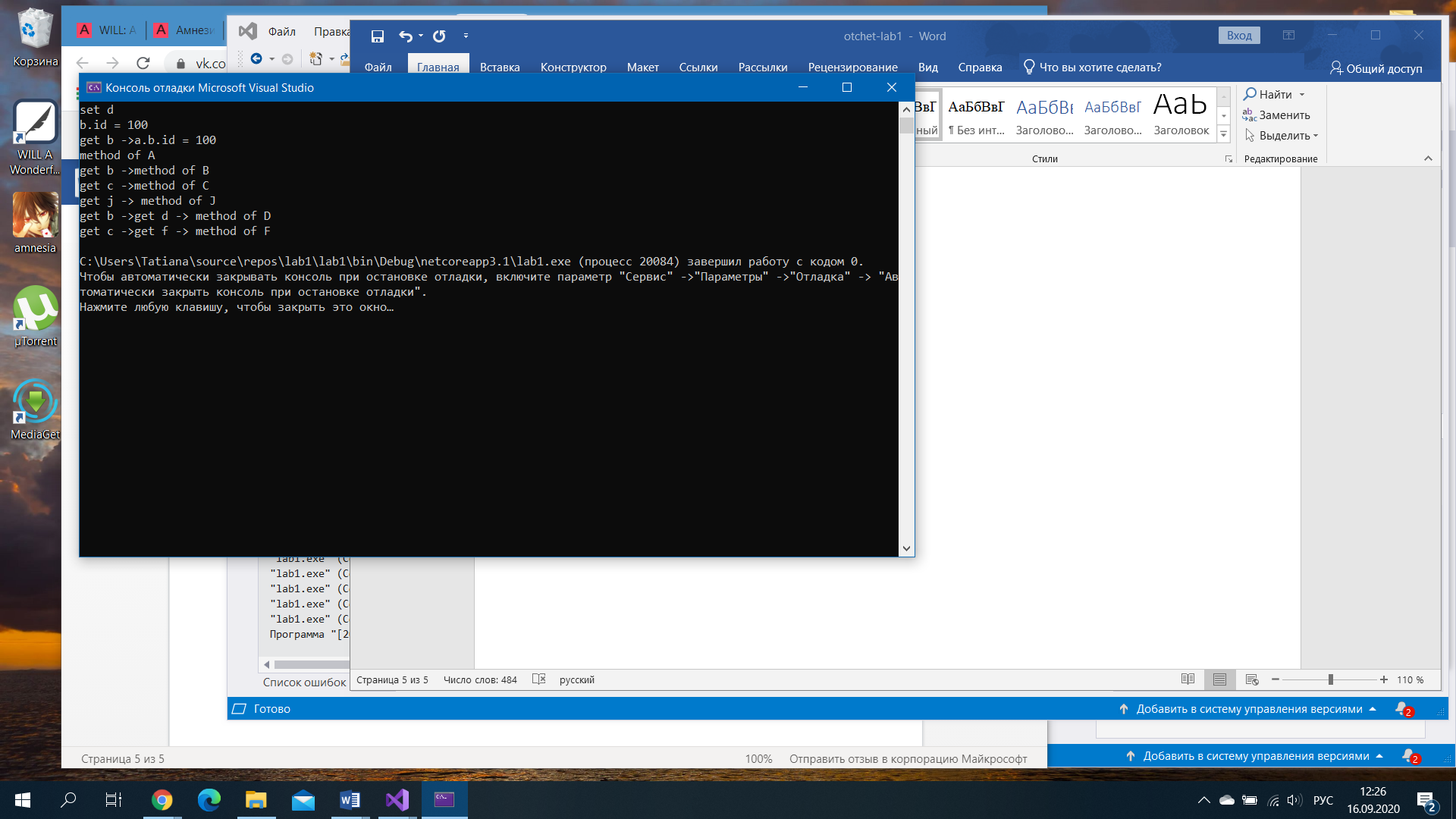
a.cA.fA.mF();

}

}

}

**Результат работы**



**Пример**

Сумка и тетради в сумке, ручка и стержень, лук и тетива, гитара и струны

**Вывод**

Агрегация по ссылке устанавливает самую слабую связь между объектами среди всех типов агрегации. При агрегации по ссылке объекты не являются единым целым в том смысле, что подчиненный объект можно создать или уничтожить независимо от того объекта, с которым он связан.

Это удобно в случае, если программа разделена на несколько несвязанных или слабо связанных логически частей, каждая из которых выполняет свою функцию, но в то же время все части должны взаимодействовать друг с другом.