**Лабораторная работа №2. Агрегация по значению и вложением.**

**Агрегация по значению.**

**A**

**Public**

**Private j b c**

**Protected**

**C**

**B**

**J**

**Public**

**Public**

**Public**

**Private**

**Private e f**

**Private k**

**Protected**

**Protected**

**Protected**

**F**

**E**

**K**

**Public**

**Public**

**Public**

**Private**

**Private**

**Private**

**Protected**

**Protected**

**Protected**

Диаграмма Классов: Агрегация по значению.

A()

J() B() C()

K() E() F()

a

e

f

k

j

b

c

**Текст программы:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace lab2

{

class Program

{

class A

{

public A() { }

~A() { }

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

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

public B bA { get { Console.Write("get b ->"); return b; } }

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

private B b = new B();

private C c = new C();

private J j = new J();

}

class B

{

public B() { }

~B() { }

public void mB() { Console.WriteLine("method of B"); }

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

private K k = new K();

}

class C

{

public C() {}

~C() { }

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

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

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

private E e = new E();

private F f = new F();

public int c1 { get; set; }

}

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 J

{

public J() { }

~J() { }

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

}

class K

{

public int k;

public K() { this.k = 15; }

~K() { }

public void mK() { Console.Write("method of K");Console.WriteLine(" k = "+k); }

}

static void Main(string[] args)

{

A a = new A();

//вкладываем в a

a.mA();

a.bA.mB();

a.cA.mC();

a.jA.mJ();

//вкладываем в a -> b - >k

a.bA.kA.mK();

//вкладываем в a -> c - > e & a -> c ->f

a.cA.eA.mE();

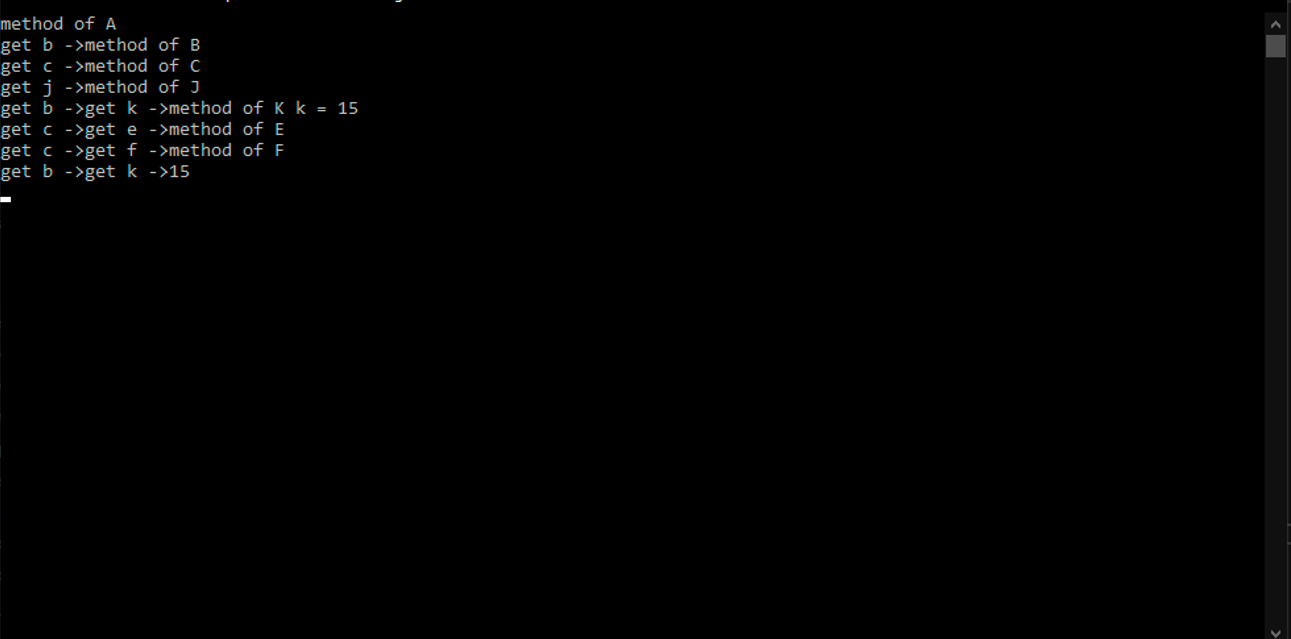
a.cA.fA.mF();

Console.WriteLine(a.bA.kA.k);

Console.ReadKey();

}

}

}

**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, c, 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.

**Агрегация вложением.**

A

F

E

C

Protected

Private e f

Public

K

Protected

Private k

B

Public

Protected

**Private b j c**

**Protected**

Private

Public

Protected

Private

Public

Protected

Private

Public

Public

Protected

Private

Public

**J**

****

****

****

**Текст программы:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace lab2\_AgregationVal

{

class Program

{

class A

{

public A() { }

~A() { }

public class B

{

public B() { }

~B() { }

public class K

{

public int k;

public K() { this.k = 15; }

~K() { }

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

}

public void mB() { Console.WriteLine(" method of B"); }

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

private K k = new K();

}

public class C

{

public C() { }

~C() { }

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

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

private E e = new E();

private F f = new F();

}

public class J

{

public J() { }

~J() { }

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

}

public class K

{

public K() { }

~K() { }

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

}

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

public B bA { get { Console.Write("get b ->"); return b; } }

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

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

private B b = new B();

private C c = new C();

private J j = new J();

}

static void Main(string[] args)

{

A a = new A();

a.mA();

//агрегируем в A: B, C, J

a.bA.mB();

a.cA.mC();

a.jA.mJ();

//агрегируем в B: K

a.bA.kA.mK();

//агрегируем в C: E, F

a.cA.eA.mE();

a.cA.fA.mF();

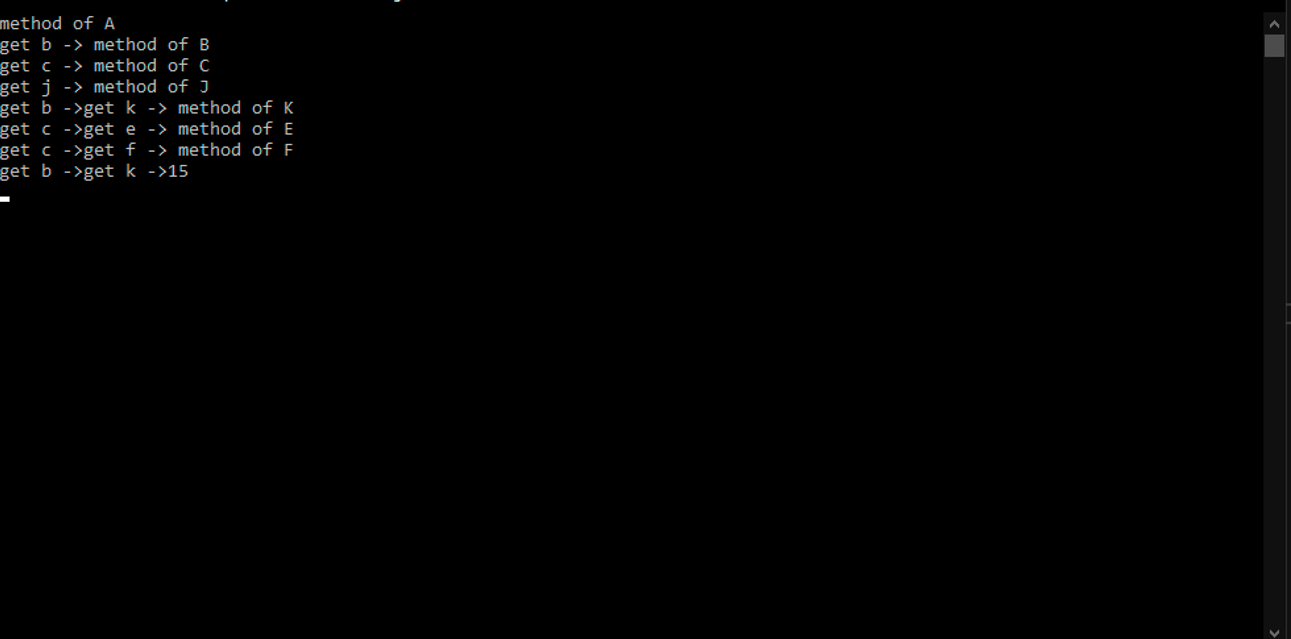
Console.WriteLine(a.bA.kA.k);

Console.ReadKey();

}

}

}



**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.