**Лабораторная работа №6. Ассоциация 1:1**

Protectededd

Private

Public

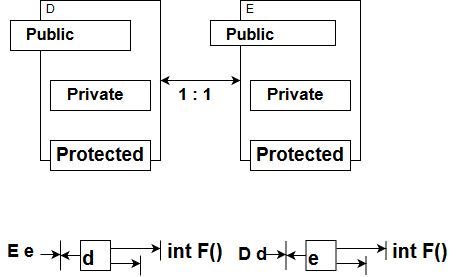
E

D

Protectededd

Public

Private

**** 1:1

**Текст программы:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace lab6\_1k1

{

class Program

{

//1 : 1

//ссылка используется

//в D на E. и наоборот

class D

{

public D() { Console.WriteLine("Constructor D"); }

~D() { }

public E e { set; get; }

public int F() { return 0; }

}

class E

{

public E()

{

Console.WriteLine("Constructor E");

this.k = 1;

}

~E() { }

public D d { set; get; }

public int k { set; get; }

public int F() { return 1; }

}

static void Main(string[] args)

{

//создаются объекты

D d = new D();

E e = new E();

//пока что эти объекты классов не связаны. их надо связать с помощью ассоциации

Console.WriteLine("d.F() = {0}", d.F());

Console.WriteLine("d.F() = {0}", e.F());

Console.WriteLine();

//ассоциация

//объект e обращается к атрибуту доступа класса D и подставляет ему объект d

e.d = d;

//аналогично для объекта d

d.e = e;

Console.WriteLine("d.F() = {0}", d.F());

Console.WriteLine("e.F() = {0}", e.F());

Console.WriteLine();

//обращение к функциям. нельзя так делать, пока не проведена ассоциация.

Console.WriteLine("e.d.F = {0}", e.d.F());

Console.WriteLine("d.e.F = {0}", d.e.F());

Console.WriteLine();

Console.WriteLine("d.e.k = {0}", d.e.k);

e.d.e.k = 2;

Console.WriteLine();

Console.WriteLine("e.d.e.k = {0}", e.d.e.k);

e.d.e.d.e.k = 3;

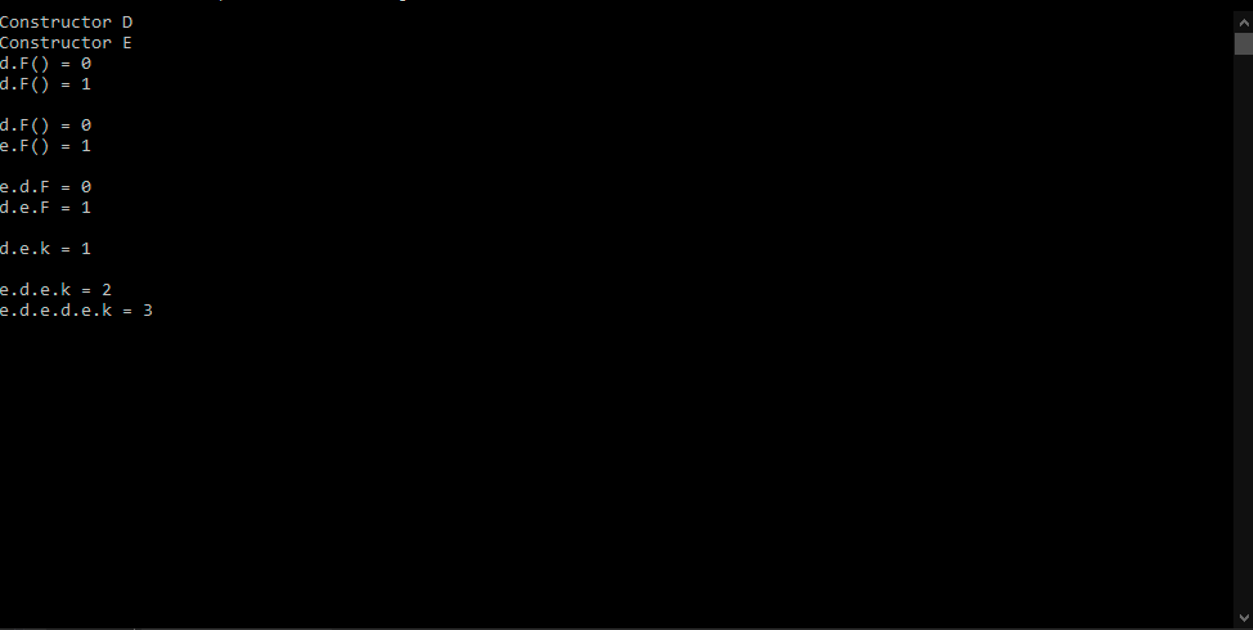
Console.WriteLine("e.d.e.d.e.k = {0}", e.d.e.d.e.k);

Console.ReadKey();

}

}

}



Вывод: Ассоциация - двусторонняя зависимость. Бинарная связь объектов. При ассоциации взаимодействующие объекты, между собой условно находятся на одном уровне. Ни один из объектов не имеет превосходства над другим, и из любого можно получить другой, связанный с ним объект.

**Ассоциация 1:N**

Protectededd

Private

Public

N

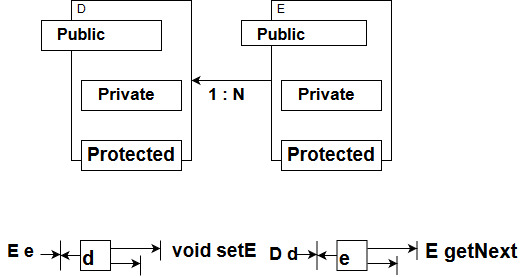
Protectededd

Private

Public

D

1:N

****

**Текст программы:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace lab6\_1kN

{

class Program

{

class D

{

//конструктор номер 1

public D()

{

this.e = new E[N];

Console.WriteLine("Constructor1 D");

}

//конструктор номер 2, если N вводится пользователем, например

public D(int N)

{

this.N = N;

this.e = new E[N];//массив скрытых ссылок на объекты класса E

Console.WriteLine("Constructor2 D");

}

~D() { Console.WriteLine("Ddestructor D"); }

public void setE(E e) // создание ассоциации

{

//ассоциация

if (size < N)

{

Console.WriteLine("Proverka");

this.e[size] = e;

size++;

}

}

//метод, позволяющий просматривать последовательно объекты класса E, связанные с объектом класса D

public E getNext(int index)

{

if (index < size)

{

return e[index];//возвращаем ссылку

}

return null;

}

private E[] e = null;

private int size = 0;

//количество объектов класса E

private int N = 7;

}

class E

{

//конструктор 1

public E() { Console.WriteLine("Constructor1 E"); }

//конструктор 2

public E(D d)

{

d.setE(this);

Console.WriteLine("Constructor2 E");

}

~E() { Console.WriteLine("Destructor E"); }

public int f() { return v; }

private int v = 11;

public D d { set; get; }

}

static void Main(string[] args)

{

D d1 = new D(); //создание объекта класса D с помощью конструктора номер 1

D d2 = new D(8);//создание объекта класса D с помощью конструктора номер 2

E e1 = new E();

d1.setE(e1);// установка для N-ого ассоциации

E e3 = new E();

d1.setE(e3);// установка для N-ого ассоциации

E e4 = new E();

d1.setE(e4);

e1.d = d1;// обычная установка ассоциации( объект е1 обращается к атрибуту доступа класса D и присваивает ему d1 )

Console.WriteLine("e1.d = {0}", e1.d);

d1.getNext(1).d = d1; // здесь мы присвоили ассоциацию как d1.e[1].d1 и что то подобного типа (под вопросом?)

Console.WriteLine("e1.d.getNext(0) = {0}", e1.d.getNext(0));

Console.WriteLine(" d1.getNext(0).f() = {0}", d1.getNext(0).f());

Console.WriteLine(" d1.getNext(1) = {0}", d1.getNext(1));

Console.WriteLine(" d1.getNext(2) = {0}", d1.getNext(2));

Console.WriteLine(" d1.getNext(1).d = {0}", d1.getNext(1).d);

E e2 = new E(d1); // (?)

e2.d = d1;

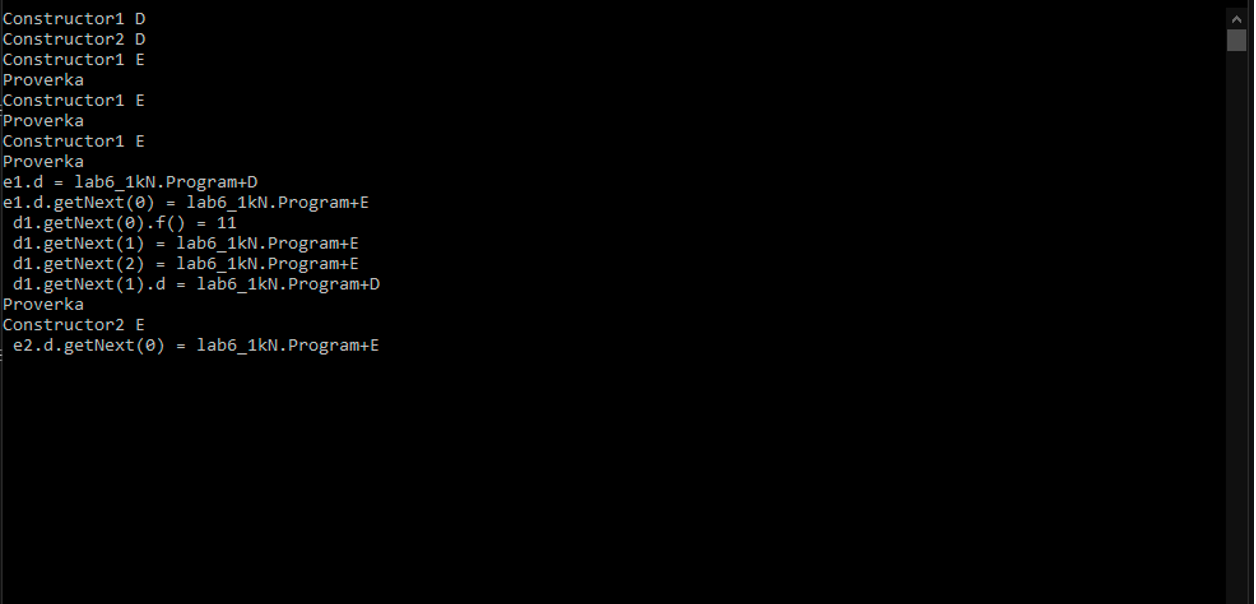
Console.WriteLine(" e2.d.getNext(0) = {0}", e2.d.getNext(0));

Console.ReadKey();

}

}

}



**Вывод:** В ассоциации 1 : N каждый из N объектов указывает на 1, общий для всех, объект, который в свою очередь имеет N указателей на другие объекты. При этом, из одного объекта группы, с помощью общего можно перейти к другим объектам этой группы. Это N-нарная отношение.