

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ  
ФЕДЕРАЦИИ

ФГБОУ ВПО «КАЗАНСКИЙ ГОСУДАРСТВЕННЫЙ АРХИТЕКТУРНО-  
СТРОИТЕЛЬНЫЙ УНИВЕРСИТЕТ» (КГАСУ)

Кафедра информационные системы и технологии в строительстве

Расчетно-графическая работа №1

по дисциплине «Объектно-ориентированное программирование»

**Выполнил:**

студент гр.1ИС02

Егорова Дарья

Андреевна

**Работу проверил:**

Хайруллин Равиль

Сагитович,

профессор, доктор

физико-

математических наук

Казань, 2023

## Проект №1.

### Класс Human

```
using ConsoleApp4;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Xml.Linq;

namespace ConsoleApp4
{
    internal class Human
    {
        private string name; // описание полей
        private string secondname;
        private string lastname;
        private int age;

        public string SecondName // описание свойств
        {
            set
            {
                secondname = value;
            }
        }
        public string LastName
        {
            set
            {
                lastname = value;
            }
        }
        public string Name
        {
            set
            {
                name = value;
            }
        }
        public int Age
        {
            get
            {
                return age;
            }
        }
    }
}
```

```

        set
        {
            age = value;
        }
    }
    public string FullName
    {
        get
        {
            Console.WriteLine("Введите пароль");
            string pw = Console.ReadLine();
            if (pw == "456")
                return lastname + " " + name + " " + secondname;
            else
                throw new Exception("Неправильный пароль!");
        }
    }
    public void Info() // описание методов
    {
        Console.WriteLine(lastname + " " + name + " " + secondname + ", " + age + Option());
    }
    private string Option()
    {
        string s;
        if (age > 10 && age <= 20)
            s = "лет";
        else
        {
            switch (age % 10)
            {
                case 1:
                    s = "год";
                    break;
                case 2:
                case 3:
                case 4:
                    s = "года";
                    break;
                default:
                    s = "лет";
                    break;
            }
        }
        return s;
    }
}

public Human(string lastname, string name, string secondname, int age)
{

```

```

        this.lastname = lastname;
        this.name = name;
        this.secondname = secondname;
        this.age = age;
        counthuman++;
    }
    public Human()
    {
        counthuman++;
    }
    static private int counthuman;
    static public int CountHuman
    {
        set
        {
            counthuman = value;
        }
    }
    static public void GetCountHuman()
    {
        Console.WriteLine("количество Human = " + counthuman);
    }
    static Human()
    {
        Console.WriteLine("Вы используете класс Human");
    }

    public static Human operator !(Human a)
    {
        return new Human(a.lastname, a.name, a.secondname,
            a.age + 1);
    }
    public static Human operator /(Human a, string s)
    {
        return new Human(s, a.name, a.secondname, a.age);
    }
}
}

```

## Класс Office

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ConsoleApp4
{

```

```

internal class Office
{
    //описание полей
    private Human boss;
    private string name;
    private int countworker;
    //описание свойств
    public string Name
    {
        set
        {
            name = value;
        }
    }
    public int CountWorker
    {
        set
        {
            countworker = value;
        }
    }
    public Human Boss
    {
        set
        {
            boss = value;
        }
    }
    //описание метода
    public void Info()
    {
        Console.WriteLine("Подразделение: {0}, работающих {1} человек", name, countworker);
        boss.Info();
    }
    //описание конструкторов
    public Office()
    {
    }
    public Office(string lastname, string name, string secondname, int age, string name1, int
countworker)
    {
        this.name = name1;
        this.countworker = countworker;
        boss = new Human(lastname, name, secondname, age);
    }
}

```

## Класс Office1

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ConsoleApp4
{
    internal class Office1
    {
        // описание полей класса Office1
        private Human1 boss;
        private string name;
        private int countworker;

        // описание вложенного класса Human1
        private class Human1
        {
            private string name;
            private string secondname;
            private string lastname;
            private int age;
            public string FullName
            {
                get
                {
                    return name + " " + secondname + " " + lastname;
                }
            }
            public void Info()
            {
                Console.WriteLine(lastname + " " + name + " "
                    + secondname + "," + age + Option());
            }
            private string Option()
            {
                string s;
                if (age > 10 && age <= 20)
                    s = "лет";
                else
                {
                    switch (age % 10)
                    {
                        case 1:
                            s = "год";
                            break;
                    }
                }
            }
        }
    }
}
```

```

        case 2:
        case 3:
        case 4:
            s = "года";
            break;
        default:
            s = "лет";
            break;
    }
}
return s;
}
public Human1(string lastname, string name,
string secondname, int age)
{
    this.lastname = lastname;
    this.name = name;
    this.secondname = secondname;
    this.age = age;
}
}
//описание метода Info класса Office1
public void Info()
{
    Console.WriteLine("Подразделение: {0}, работающих {1} человек", name, countworker);
    boss.Info();
}
// описание конструктора класса Office1
public Office1(string lastname, string name,
string secondname, int age, string name1,
int countworker)
{
    this.name = name1;
    this.countworker = countworker;
    boss = new Human1(lastname, name, secondname, age);
}
}
}

```

## Main

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

```

```

namespace ConsoleApp4
{
    internal class Program
    {
        static void Main(string[] args)
        {
            Human.CountHuman = 0;
            Human.GetCountHuman();
            Human hmn1 = new Human("Иванов", "Василий", "Петрович", 45);
            Human.GetCountHuman();
            hmn1.Info();
            Human hmn2 = new Human();
            Human.GetCountHuman();
            hmn2.LastName = "Сидоров";
            hmn2.Name = "Александр";
            hmn2.SecondName = "Михайлович";
            hmn2.Age = 42;
            Console.WriteLine("Сотрудник {0}, возраст {1}", hmn2.FullName, hmn2.Age);

            Console.WriteLine("\nКлассы как поля");
            Office off1 = new Office();
            off1.Name = "кафедра математики";
            off1.CountWorker = 12;
            off1.Boss = new Human("Иванов", "Петр", "Николаевич", 46);
            off1.Info();
            Office off2 = new Office("Алексеев", "Семен",
                "Михайлович", 53, "кафедра физики", 15);
            off2.Info();

            Console.WriteLine("\nВложенные классы");
            Office1 off3 = new Office1("Попов", "Иван", "Петрович", 41, "кафедра химии", 9);
            off3.Info();

            Console.WriteLine("\nОписание операций");
            Human hmn4 = new Human("Васильева", "Александра",
                "Павловна", 24);
            hmn4.Info();
            hmn4 = !hmn4;
            hmn4.Info();
            hmn4 = hmn4 / "Петрова";
            hmn4.Info();

            Human hmn5 = new Human("Иванов", "Василий", "Петрович", 45);
            Human hmn6 = new Human("Титов", "Петр", "Николаевич", 46);
            Human hmn7 = new Human("Васильева", "Александра", "Павловна", 24);
            Human hmn8 = new Human("Алексеев", "Федор", "Никифорович", 33);
            Human hmn9 = new Human("Шишкин", "Константин", "Алексеевич", 37);
        }
    }
}

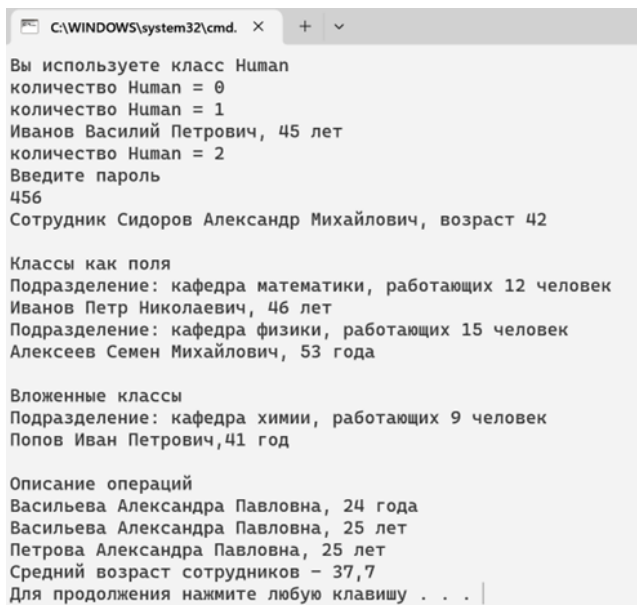
```



```

Human hmn10 = new Human("Николаев", "Алексей", "Иванович", 41);
Human[] a = { hmn5, hmn6, hmn7, hmn8, hmn9, hmn10 };
double s = 0;
for (int k = 0; k < a.Length; k++)
    s += a[k].Age;
s = Math.Round(s / a.Length, 1);
Console.WriteLine("Средний возраст сотрудников - {0}", s);
    }
}
}

```



```

C:\WINDOWS\system32\cmd.  X  +  v
Вы используете класс Human
количество Human = 0
количество Human = 1
Иванов Василий Петрович, 45 лет
количество Human = 2
Введите пароль
456
Сотрудник Сидоров Александр Михайлович, возраст 42

Классы как поля
Подразделение: кафедра математики, работающих 12 человек
Иванов Петр Николаевич, 46 лет
Подразделение: кафедра физики, работающих 15 человек
Алексеев Семен Михайлович, 53 года

Вложенные классы
Подразделение: кафедра химии, работающих 9 человек
Попов Иван Петрович, 41 год

Описание операций
Васильева Александра Павловна, 24 года
Васильева Александра Павловна, 25 лет
Петрова Александра Павловна, 25 лет
Средний возраст сотрудников - 37,7
Для продолжения нажмите любую клавишу . . . |

```

## Проект №2.

### Main

```

internal class Program
{
    static void Main(string[] args)
    {
        Console.WriteLine("используем класс");
        HumanCl h1 = new HumanCl("Иванов", "Василий", "Петрович", 46);
        HumanCl h2 = new HumanCl();
        h2 = h1;
        Console.WriteLine("до изменения");
        h1.Info();
        h2.Info();
        Console.WriteLine("после изменения");
        h1.Name = "Семен";
    }
}

```

```

h1.LastName = "Сидоров";
h1.SecondName = "Алексеевич";
h1.Age = 42;
h1.Info();
h2.Info();

Console.WriteLine("используем структуру");
HumanSt h3 = new HumanSt("Иванов", "Василий", "Петрович", 46);
HumanSt h4 = h3;
Console.WriteLine("до изменения");
h3.Info();
h4.Info();
Console.WriteLine("после изменения");
h3.Name = "Семен";
h3.LastName = "Сидоров";
h3.SecondName = "Алексеевич";
h3.Age = 42;
h3.Info();
h4.Info();

Console.WriteLine("Частичные классы и методы");
Student st = new Student("Иванов", "9ИС01");
st.F();
st.G();

st.G1();

var user = new { name = "Алексеев", group = "1ИС03" };
Console.WriteLine("Имя пользователя {0}, группа {1}",
    user.name, user.group);
user = new { name = "Алексеев", group = "1ИС04" };
Console.WriteLine("Имя пользователя {0}, группа {1}",
    user.name, user.group);
    }
}
}

```

## Класс HumanCl

```

internal class HumanCl
{
    public HumanCl()
    {
    }
    private string name;
    private string secondname;
    private string lastname;
    private int age;
    public string Name

```

```

{
    set
    {
        name = value;
    }
}
public string SecondName
{
    set
    {
        secondname = value;
    }
}
public string LastName
{
    set
    {
        lastname = value;
    }
}
public int Age
{
    set
    {
        age = value;
    }
}
public void Info()
{
    Console.WriteLine(lastname + " " + name + " " + secondname + "," + age + Option());
}
private string Option()
{
    string s;
    if (age > 10 && age <= 20)
        s = " лет";
    else
    {
        switch (age % 10)
        {
            case 1:
                s = " год";
                break;
            case 2:
            case 3:
            case 4:
                s = " года";
                break;
        }
    }
}

```

```

        default:
            s = " Jet";
            break;
        }
    }
    return s;
}
public HumanCl(string lastname, string name, string secondname, int age)
{
    this.lastname = lastname;
    this.name = name;
    this.secondname = secondname;
    this.age = age;
}
}
}

```

## Структура HumanSt

```

internal struct HumanSt
{
    private string name;
    private string secondname;
    private string lastname;
    private int age;
    public string Name
    {
        set
        {
            name = value;
        }
    }
    public string SecondName
    {
        set
        {
            secondname = value;
        }
    }
    public string LastName
    {
        set
        {
            lastname = value;
        }
    }
    public int Age
    {
        set

```

```

        {
            age = value;
        }
    }
    public void Info()
    {
        Console.WriteLine(lastname + " " + name + " " + secondname + "," + age + Option());
    }
    private string Option()
    {
        string s;
        if (age > 10 && age <= 20)
            s = "лет";
        else
        {
            switch (age % 10)
            {
                case 1:
                    s = "год";
                    break;
                case 2:
                case 3:
                case 4:
                    s = "года";
                    break;
                default:
                    s = "лет";
                    break;
            }
        }
        return s;
    }
    //конструктор с параметрами
    public HumanSt(string lastname, string name, string secondname, int age)
    {
        this.lastname = lastname;
        this.name = name;
        this.secondname = secondname;
        this.age = age;
    }
}

```

```

partial class Student
{
    private string name;
    public void G()

```

```

    {
        Console.WriteLine(grup);
    }
    partial void F1();
    public void G1()
    {
        F1();
    }
}
}

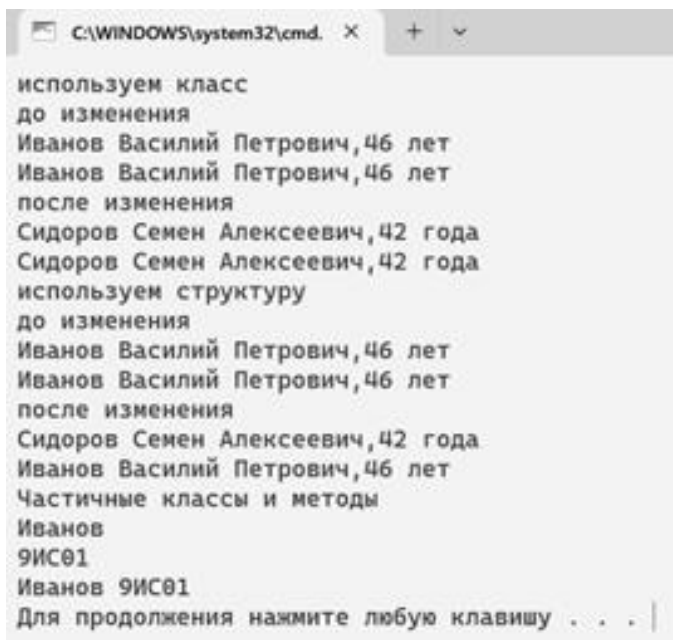
```

## Класс Student

```

partial class Student
{
    private string grup;
    public Student(string name, string grup)
    {
        this.name = name;
        this.grup = grup;
    }
    public void F()
    {
        Console.WriteLine(name);
    }
    partial void F1()
    {
        Console.WriteLine(name + " " + grup);
    }
}
}

```



The screenshot shows a Windows Command Prompt window with the title bar 'C:\WINDOWS\system32\cmd. x'. The output text is as follows:

```

используем класс
до изменения
Иванов Василий Петрович,46 лет
Иванов Василий Петрович,46 лет
после изменения
Сидоров Семен Алексеевич,42 года
Сидоров Семен Алексеевич,42 года
используем структуру
до изменения
Иванов Василий Петрович,46 лет
Иванов Василий Петрович,46 лет
после изменения
Сидоров Семен Алексеевич,42 года
Иванов Василий Петрович,46 лет
Частичные классы и методы
Иванов
9ИС01
Иванов 9ИС01
Для продолжения нажмите любую клавишу . . . |

```

## Проект №3.

### Main

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ConsoleApp6
{
    internal class Program
    {
        static void Main(string[] args)
        {
            Worker wrk1 = new Worker();
            wrk1.LastName = "Сидоров";
            wrk1.Name = "Александр";
            wrk1.SecondName = "Михайлович";
            wrk1.Age = 42;
            wrk1.Tarif = 2000;
            wrk1.Info();
            Console.WriteLine("Сотрудник {0}, зарплата {1}",
            wrk1.FullName, wrk1.ZarPlata(20));
            Worker wrk2 = new Worker("Иванов", "Василий", "Петрович", 45, 1500);
            wrk2.Info();

            Human wrk3 = wrk2;
            wrk3.Info();
            Worker wrk4 = (Worker)wrk3; //операция приведения типа
            wrk4.Info();

            Console.WriteLine("\nИспользование виртуального метода!");
            wrk2.Info1();
            wrk3.Info1();
            wrk4.Info1();

            Console.WriteLine("\nИспользование System Object");
            Object ob1, ob2, ob3;
            ob1 = new System.Object();
            ob2 = ob1;
            Console.WriteLine(ob1.Equals(ob2));
            ob3 = new System.Object();
            Console.WriteLine(ob1.Equals(ob3));
            Console.WriteLine(ob1.GetHashCode());
            Console.WriteLine(ob2.GetHashCode());
            Console.WriteLine(ob3.GetHashCode());
            Console.WriteLine(ob1.GetType());
        }
    }
}
```

```

Console.WriteLine(ob1.ToString());

Console.WriteLine("\nИспользование System String");
Object ob4, ob5, ob6;
ob4 = "Hello";
ob5 = ob4;
Console.WriteLine(ob4.Equals(ob5));
ob6 = "Hello";
Console.WriteLine(ob4.Equals(ob6));
Console.WriteLine(ob4.GetHashCode());
Console.WriteLine(ob5.GetHashCode());
Console.WriteLine(ob6.GetHashCode());
Console.WriteLine(ob4.GetType());
Console.WriteLine(ob4.ToString());

Console.WriteLine("\nИспользование арифметический тип");
Object ob7, ob8, ob9;
ob7 = 3;
ob8 = ob7;
Console.WriteLine(ob7.Equals(ob8));
ob9 = 3;
Console.WriteLine(ob7.Equals(ob9));
Console.WriteLine(ob7.GetHashCode());
Console.WriteLine(ob8.GetHashCode());
Console.WriteLine(ob9.GetHashCode());
Console.WriteLine(ob7.GetType());
Console.WriteLine(ob7.ToString());

Console.WriteLine("\nИспользование Переопределенные методы");
Object ob10, ob11, ob12;
ob10 = new Human("Васильева", "Александра", "Павловна", 24);
ob11 = ob10;
Console.WriteLine(ob10.Equals(ob11));
ob12 = new Human("Васильева", "Александра", "Павловна", 24);
Console.WriteLine(ob10.Equals(ob12));
Console.WriteLine(ob10.GetHashCode());
Console.WriteLine(ob11.GetHashCode());
Console.WriteLine(ob12.GetHashCode());
Console.WriteLine(ob10.GetType());
Console.WriteLine(ob10.ToString());
    }
}
}

```

## Дочерний класс Worker

```

using System;
using System.Collections.Generic;
using System.Linq;

```



```

using System.Text;
using System.Threading.Tasks;
using System.Xml.Linq;

namespace ConsoleApp6
{
    class Worker : Human
    {
        protected int tarif;
        public int Tarif
        {
            set
            {
                tarif = value;
            }
        }
        public int ZarPlata(int countday)
        {
            return tarif * countday;
        }
        public new void Info()
        {
            Console.WriteLine(lastname + " " + name + " " +
                secondname + ", " + age + Option() + ", " + tarif
                + " руб/день");
        }
        // конструктор с параметрами
        public Worker(string lastname, string name,
            string secondname, int age, int tarif) :
            base(lastname, name, secondname, age)
        {
            this.tarif = tarif;
        }
        public Worker()
        {
        }
        override public void Info1()
        {
            Console.WriteLine(lastname + " " + name + " " + secondname + ", " + age + Option() + ", "
                + tarif + " руб/день");
        }
    }
}

```

## Класс Human

```

using System;
using System.Collections.Generic;

```

```

using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ConsoleApp6
{
    internal class Human
    {
        protected string name;
        protected string secondname;
        protected string lastname;
        protected int age;

        public string SecondName
        {
            set
            {
                secondname = value;
            }
        }
        public string LastName
        {
            set
            {
                lastname = value;
            }
        }
        public string Name
        {
            set
            {
                name = value;
            }
        }
        public int Age
        {
            get
            {
                return age;
            }
            set
            {
                age = value;
            }
        }
        public string FullName
        {
            get

```

```

    {
        Console.WriteLine("Введите пароль");
        string pw = Console.ReadLine();
        if (pw == "456")
            return lastname + " " + name + " " + secondname;
        else
            throw new Exception("Неправильный пароль!");
    }
}

public void Info()
{
    Console.WriteLine(lastname + " " + name + " " + secondname + ", " + age + Option());
}

protected string Option()
{
    string s;
    if (age > 10 && age <= 20)
        s = " лет";
    else
    {
        switch (age % 10)
        {
            case 1:
                s = " год";
                break;
            case 2:
            case 3:
            case 4:
                s = " года";
                break;
            default:
                s = " лет";
                break;
        }
    }
    return s;
}

public Human(string lastname, string name, string secondname, int age)
{
    this.lastname = lastname;
    this.name = name;
    this.secondname = secondname;
    this.age = age;
    counthuman++;
}

public Human()
{

```

```

        counthuman++;
    }
    static private int counthuman;
    static public int CountHuman
    {
        set
        {
            counthuman = value;
        }
    }
    static public void GetCountHuman()
    {
        Console.WriteLine("количество Human = " + counthuman);
    }
    static Human()
    {
        Console.WriteLine("Вы используете класс Human");
    }

    public static Human operator !(Human a)
    {
        return new Human(a.lastname, a.name, a.secondname,
            a.age + 1);
    }
    public static Human operator /(Human a, string s)
    {
        return new Human(s, a.name, a.secondname, a.age);
    }
    virtual public void Info1()
    {
        Console.WriteLine lastname + " " + name + " " + secondname + ", " + age + Option();
    }

    public override bool Equals(object ob)
    {
        Human hmn = (Human)ob;
        return name == hmn.name && lastname == hmn.lastname
        &&
        secondname == hmn.secondname && age == hmn.age;
    }
    public override string ToString()
    {
        return name + " " + secondname + " " + lastname +
            ", " + age;
    }
}
}

```

Вы используете класс Human  
Сидоров Александр Михайлович, 42 года, 2000 руб/день  
Введите пароль  
456

Сотрудник Сидоров Александр Михайлович, зарплата 40000  
Иванов Василий Петрович, 45 лет, 1500 руб/день  
Иванов Василий Петрович, 45 лет  
Иванов Василий Петрович, 45 лет, 1500 руб/день

Использование виртуального метода!  
Иванов Василий Петрович, 45 лет, 1500руб/день  
Иванов Василий Петрович, 45 лет, 1500руб/день  
Иванов Василий Петрович, 45 лет, 1500руб/день

Использование System Object  
True  
False  
46104728  
46104728  
12289376  
System.Object  
System.Object

Использование System String  
True  
True  
-694847  
-694847  
-694847  
System.String  
Hello

Использование арифметический тип  
True  
True  
3  
3  
3  
System.Int32  
3

Использование Переопределенные методы  
True  
True  
43495525  
43495525  
55915408  
ConsoleApp6.Human  
Александра Павловна Васильева, 24  
Для продолжения нажмите любую клавишу . . . |

## Проект №4.

### Класс Human

using System;

abstract class Human

```
{
    protected string name;
    protected string secondname;
    protected string lastname;
    protected int age;
    public abstract int ZarPlata();
    public void Info()
    {
        Console.WriteLine("{0} {1} {2}, {3} {4}, зарплата {5} руб", lastname, name, secondname,
age, Option(), ZarPlata());
    }
    private string Option()
    {
        string s;
        if (age > 10 && age <= 20)
            s = " лет";
        else
        {
            switch (age % 10)
            {
                case 1:
                    s = " год";
                    break;
                case 2:
                case 3:
                case 4:
                    s = " года";
                    break;
                default:
                    s = " лет";
                    break;
            }
        }
        return s;
    }
    public Human(string lastname, string name, string secondname, int age)
    {
```

```

        this.lastname = lastname;
        this.name = name;
        this.secondname = secondname;
        this.age = age;
    }
}

```

## Класс Integer

```
using System;
```

```

class Ingener: Human
{
    private int oklad;
    public override int ZarPlata()
    {
        return oklad;
    }
    public Ingener(string lastname, string name, string secondname, int age, int oklad):
base(lastname, name, secondname, age)
    {
        this.oklad = oklad;
    }
}

```

## Класс Worker

```
using System;
```

```

class Worker: Human
{
    private int tarif;
    private int countday;
    public override int ZarPlata()
    {
        return tarif * countday;
    }
    public Worker(string lastname, string name, string secondname, int age, int tarif, int countday):
base(lastname, name, secondname, age)
    {
        this.tarif = tarif;
        this.countday = countday;
    }
}

```

## Класс AbsStavka

```
AbsStavka
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Проект_4
{
    abstract class AbsStavka
    {
        public const int stavka1 = 2500;
        public const int stavka2 = 3000;
        public const int stavka3 = 3500;
    }
}
```

## Класс OkladEnum

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Проект_4
{
    enum OkladEnum
    {
        oklad1 = 25000,
        oklad2 = 30000,
        oklad3 = 35000
    }
}
```

## Класс Integer

```
using System;
using System.Collections.Generic;
using System.Linq;
```



```

using System.Text;
using System.Threading.Tasks;

namespace Пpoект_4
{
    abstract class AbsOklad
    {
        public const int oklad1 = 25000;
        public const int oklad2 = 30000;
        public const int oklad3 = 35000;
    }
}

Oklad
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Пpoект_4
{
    class Oklad
    {
        static private int oklad1 = 25000;
        static private int oklad2 = 30000;
        static private int oklad3 = 35000;
        static public int Oklad1
        {
            get { return oklad1; }
        }
        static public int Oklad2
        {
            get { return oklad2; }
        }
        static public int Oklad3
        {
            get { return oklad3; }
        }
        static public void Koeff(float k)
        {
            oklad1 = (int)(oklad1 * k);
            oklad2 = (int)(oklad2 * k);
            oklad3 = (int)(oklad3 * k);
        }
        private Oklad() { }
    }
}

```

```

    }
}
StavkaEnum
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Пpoект_4
{
    enum StavkaEnum
    {
        stavka1 = 2500,
        stavka2 = 3000,
        stavka3 = 3500
    }
}
Stavka
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Пpoект_4
{
    class Stavka
    {
        static private int stavka1 = 2500;
        static private int stavka2 = 3000;
        static private int stavka3 = 3500;
        static public int Stavka1
        {
            get { return stavka1; }
        }
        static public int Stavka2
        {
            get { return stavka2; }
        }
        static public int Stavka3
        {
            get { return stavka3; }
        }
        static public void Koeff(float k)
    }
}

```

```

    {
        stavka1 = (int)(stavka1 * k);
        stavka2 = (int)(stavka2 * k);
        stavka3 = (int)(stavka3 * k);
    }
    private Stavka() { }
}
}
PROGRAM
namespace Проект_4
{
    class Program
    {
        static void Main(string[] args)
        {
            Console.WriteLine("\n" + "Абстрактные классы");
            Human wrk1 = new Worker("Иванов", "Василий", "Петрович", 45, 1500, 20);
            wrk1.Info();
            Human wrk2 = new Ingener("Сидоров", "Семен", "Алексеевич", 42, 35000);
            wrk2.Info();

            Console.WriteLine("\n" + "Классы без экземпляров");
            Human wrk01 = new Worker("Егоров", "Евгений", "Федорович", 35, Stavka.Stavka3, 25);
            wrk01.Info();
            Human wrk02 = new Ingener("Иванов", "Василий", "Петрович", 46, Oklad.Oklad3);
            wrk02.Info();
            Human wrk03 = new Ingener("Сидоров", "Семен", "Алексеевич", 42, Oklad.Oklad3);
            wrk03.Info();
            Oklad.Koeff(1.3f);
            Stavka.Koeff(1.3f);
            Human wrk011 = new Worker("Егоров", "Евгений", "Федорович", 35, Stavka.Stavka3,
25);
            wrk011.Info();
            Human wrk022 = new Ingener("Иванов", "Василий", "Петрович", 46, Oklad.Oklad2);
            wrk022.Info();
            Human wrk033 = new Ingener("Сидоров", "Семен", "Алексеевич", 42, Oklad.Oklad2);
            wrk033.Info();

            Console.WriteLine("\n" + "Константы");
            Human wrk001 = new Ingener("Иванов", "Василий", "Петрович", 46, AbsOklad.oklad3);
            wrk001.Info();
            Human wrk002 = new Ingener("Сидоров", "Семен", "Алексеевич", 42, AbsOklad.oklad2);
            wrk002.Info();
            Human wrk003 = new Worker("Егоров", "Евгений", "Федорович", 35, AbsStavka.stavka3,
25);

```

```

        wrk003.Info();

        Console.WriteLine("\n" + "Перечисления");
        Human wrk0001 = new Ingener("Иванов", "Василий", "Петрович", 46,
(int)OkladEnum.oklad3);
        wrk0001.Info();
        Human wrk0002 = new Ingener("Сидоров", "Семен", "Алексеевич", 42,
(int)OkladEnum.oklad2);
        wrk0002.Info();
        Human wrk0003 = new Worker("Егоров", "Евгений", "Федорович", 35,
(int)StavkaEnum.stavka3, 25);
        wrk0003.Info();

        Console.WriteLine("\n" + Enum.Format(typeof(OkladEnum), 30000, "G"));
        Console.WriteLine(Enum.Format(typeof(OkladEnum), 30000, "X"));
        Console.WriteLine(Enum.Format(typeof(OkladEnum), 30000, "d") + "\n");

        Console.WriteLine("\n" + Enum.Format(typeof(StavkaEnum), 3000, "G"));
        Console.WriteLine(Enum.Format(typeof(StavkaEnum), 3000, "X"));
        Console.WriteLine(Enum.Format(typeof(StavkaEnum), 3000, "d") + "\n");

        int[] a = (int[])Enum.GetValues(typeof(OkladEnum));
        Console.WriteLine(a[1]);

        int[] b = (int[])Enum.GetValues(typeof(StavkaEnum));
        Console.WriteLine(b[1]);
    }
}
}

```

```

Классы без экземпляров
Калимуллин Тимур Тагирович, 35 лет, зарплата 87500 руб
Петров Иван Васильевич, 46 лет, зарплата 35000 руб
Шишкин Петр Алексеевич, 42 года, зарплата 35000 руб
Калимуллин Тимур Тагирович, 35 лет, зарплата 113725 руб
Петров Иван Васильевич, 46 лет, зарплата 38999 руб
Шишкин Петр Алексеевич, 42 года, зарплата 38999 руб

Константы
Петров Иван Васильевич, 46 лет, зарплата 35000 руб
Шишкин Петр Алексеевич, 42 года, зарплата 30000 руб
Калимуллин Тимур Тагирович, 35 лет, зарплата 87500 руб

Перечисления
Петров Иван Васильевич, 46 лет, зарплата 35000 руб
Шишкин Петр Алексеевич, 42 года, зарплата 30000 руб
Калимуллин Тимур Тагирович, 35 лет, зарплата 87500 руб

oklad2
00007530
30000

stavka2
00000888
3000

30000
3000
Для продолжения нажмите любую клавишу . . .

```

## Проект №5.

### Figure

```

namespace ConsoleApp7
{
    abstract class Figure
    {
        public abstract double Square();
        public abstract double Perimeter();
    }
}

```

### IFigura1

```

public interface IFigura1
{
    double Square();
    double Perimeter();
}

```

```
}  
}
```

## IFigura2

```
public interface IFigura2: ISquare2  
{  
    double Perimeter();  
}  
}
```

## IPerimetr3

```
public interface IPerimetr3  
{  
    double Perimeter();  
}  
}
```

## ISquare2

```
public interface ISquare2  
{  
    double Square();  
}  
}
```

## Rectangle

```
internal class Rectangle: Figure  
{  
    double a, b; //Стороны  
    //Конструктор  
    public Rectangle(double a, double b)  
    {  
        this.a = a;  
        this.b = b;  
    }  
    public override double Square()  
    {  
        return a * b;  
    }  
    public override double Perimeter()  
    {  
        return (a + b) * 2;  
    }  
}  
}
```

## Rectangle1

```

internal class Rectangle1: IFigura1
{
    double a, b; //Стороны
        //Конструктор
    public Rectangle1(double a, double b)
    {
        this.a = a;
        this.b = b;
    }
    public double Square()
    {
        return a * b;
    }
    public double Perimeter()
    {
        return (a + b) * 2;
    }
}

```

## Rectangle2

```

internal class Rectangle2: IFigura2
{
    double a, b; //Стороны
        //Конструктор
    public Rectangle2(double a, double b)
    {
        this.a = a;
        this.b = b;
    }
    public double Square()
    {
        return a * b;
    }
    public double Perimeter()
    {
        return (a + b) * 2;
    }
}

```

## Rectangle3

```

internal class Rectangle3: ISquare2, IPerimetr3
{
    double a, b; //Стороны
        //Конструктор
    public Rectangle3(double a, double b)

```

```

    {
        this.a = a;
        this.b = b;
    }
    public double Square()
    {
        return a * b;
    }
    public double Perimeter()
    {
        return (a + b) * 2;
    }
}

```

## Triangle

```

internal class Triangle: Figure
{
    double a, b, c; //Стороны
    //Конструктор
    public Triangle(double a, double b, double c)
    {
        this.a = a;
        this.b = b;
        this.c = c;
    }
    public override double Square()
    {
        double p = (a + b + c) / 2; //Используем формулу Герона
        return Math.Sqrt(p * (p - a) * (p - b) * (p - c));
    }
    public override double Perimeter()
    {
        return a + b + c;
    }
}

```

## Triangle1

```

internal class Triangle1: IFigure1
{
    double a, b, c; //Стороны
    //Конструктор
    public Triangle1(double a, double b, double c)
    {
        this.a = a;
        this.b = b;
    }
}

```



```

        this.c = c;
    }
    public double Square()
    {
        //Используем формулу Герона
        double p = (a + b + c) / 2;
        return Math.Sqrt(p * (p - a) * (p - b) * (p - c));
    }
    public double Perimeter()
    {
        return a + b + c;
    }
}

```

## Triangle2

```

internal class Triangle2: IFigura2
{
    double a, b, c; //Стороны
    //Конструктор
    public Triangle2(double a, double b, double c)
    {
        this.a = a;
        this.b = b;
        this.c = c;
    }
    public double Square()
    {
        //Используем формулу Герона
        double p = (a + b + c) / 2;
        return Math.Sqrt(p * (p - a) * (p - b) * (p - c));
    }
    public double Perimeter()
    {
        return a + b + c;
    }
}

```

## Triangle3

```

internal class Triangle3 :ISquare2, IPerimetr3
{
    double a, b, c; //Стороны
    //Конструктор
    public Triangle3(double a, double b, double c)
    {
        this.a = a;
        this.b = b;
    }
}

```

```

        this.c = c;
    }
    public double Square()
    {
        //Используем формулу Герона
        double p = (a + b + c) / 2;
        return Math.Sqrt(p * (p - a) * (p - b) * (p - c));
    }
    public double Perimeter()
    {
        return a + b + c;
    }
}
}

```

## Main

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ConsoleApp7
{
    internal class Program
    {
        static void Main(string[] args)
        {
            Figure f1, f2;
            f1 = new Triangle(3, 4, 5);
            f2 = new Rectangle(2, 6);
            Console.WriteLine(f1.Perimeter() + ", " + f1.Square());
            Console.WriteLine(f2.Perimeter() + ", " + f2.Square());

            Triangle1 f3;
            f3 = new Triangle1(6, 8, 10);
            Console.WriteLine(f3.Perimeter() + ", " + f3.Square());

            Triangle2 f4;
            f4 = new Triangle2(5, 12, 13);
            Console.WriteLine(f4.Perimeter() + ", " + f4.Square());

            Triangle3 f5;
            f5 = new Triangle3(8, 15, 17);
            Console.WriteLine(f4.Perimeter() + ", " + f5.Square());

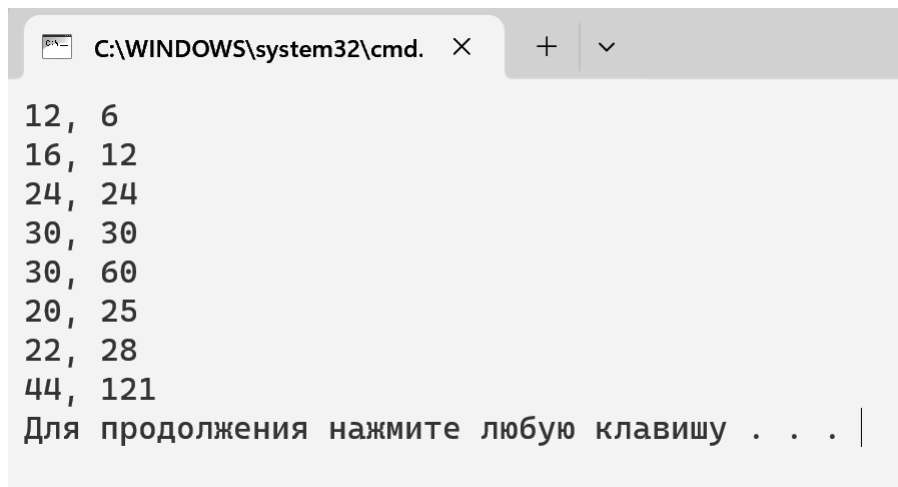
            Rectangle1 f6;
            f6 = new Rectangle1(5, 5);

```

```
Console.WriteLine(f6.Perimeter() + ", " + f6.Square());

Rectangle2 f7;
f7 = new Rectangle2(4, 7);
Console.WriteLine(f7.Perimeter() + ", " + f7.Square());

Rectangle3 f8;
f8 = new Rectangle3(11, 11);
Console.WriteLine(f8.Perimeter() + ", " + f8.Square());
    }
}
}
```



```
C:\WINDOWS\system32\cmd.  ×  +  ∨

12, 6
16, 12
24, 24
30, 30
30, 60
20, 25
22, 28
44, 121
Для продолжения нажмите любую клавишу . . . |
```

Кафедра информационных систем  
и технологий в строительстве

**Расчетно-графическая работа №2**

по дисциплине «Объектно-ориентированное программирование»

**Выполнил:**

студент гр.1ИС02

Егорова Дарья

Андреевна

**Работу проверил:**

Хайруллин Равиль

Сагитович,

профессор, доктор

физико-

математических наук

Казань, 2023

## Проект №6

### Класс Game

```
class Game<P>
{
    private string company;
    private P member;
    public string Company
    {
        set
        {
            company = value;
        }
        get
        {
            return company;
        }
    }
    public P Member
    {
        set
        {
            member = value;
        }
        get
        {
            return member;
        }
    }
    public Game(string company, P member)
    {
        this.company = company;
        this.member = member;
    }
}
```

### Класс Game1

```
class Game1<P> where P : Person1
{
    private string company;
    private P member;
    public string Company
    {
        set
        {
            company = value;
        }
        get
    }
}
```

```

        {
            return company;
        }
    }
    public P Member
    {
        set
        {
            member = value;
        }
        get
        {
            return member;
        }
    }
    public Game1(string company, P member)
    {
        this.company = company;
        this.member = member;
    }
    public void PrName()
    {
        Console.WriteLine("Фамилия участника {0}",
            member.Name);
    }
}

```

## Класс Person1

```

class Person1
{
    private string name;
    private object info;
    public string Name
    {
        set
        {
            name = value;
        }
        get
        {
            return name;
        }
    }
    public object Info
    {
        set
        {
            info = value;
        }
        get
    }
}

```

```

        {
            return info;
        }
    }
    public Person1(string name, object info)
    {
        this.name = name;
        this.info = info;
    }
}

```

## Класс Person2

```

class Person2<T> //обобщенный класс
{
    private string name;
    private T info;
    public string Name
    {
        set
        {
            name = value;
        }
        get
        {
            return name;
        }
    }
    public T Info
    {
        set
        {
            info = value;
        }
        get
        {
            return info;
        }
    }
    public Person2(string name, T info)
    {
        this.name = name;
        this.info = info;
    }
    static private T code;
    static public T Code
    {
        set
        {
            code = value;
        }
    }
}

```

```

    get
    {
        return code;
    }
}

```

## Класс Person3

```

class Person3<T, P>
{
    private string name;
    private T info;
    private P id;
    public string Name
    {
        set
        {
            name = value;
        }
        get
        {
            return name;
        }
    }
    public T Info
    {
        set
        {
            info = value;
        }
        get
        {
            return info;
        }
    }
    public P Id
    {
        set
        {
            id = value;
        }
        get
        {
            return id;
        }
    }
    public Person3(string name, T info, P id)
    {
        this.name = name;
        this.info = info;
    }
}

```



```

        this.id = id;
    }
}

```

## Класс PersonId

```

class PersonId : Person1
{
    private int id;
    public int Id
    {
        set
        {
            id = value;
        }
        get
        {
            return id;
        }
    }
    public PersonId(string name, object info, int id) :
        base(name, info)
    {
        this.id = id;
    }
}

```

## Класс PersonId1

```

class PersonId1<T> : Person2<T>
{
    private int id;
    public int Id
    {
        set
        {
            id = value;
        }
        get
        {
            return id;
        }
    }
    public PersonId1(string name, T info, int id) :
        base(name, info)
    {
        this.id = id;
    }
}

```

```
}
```

## Класс PersonId2

```
class PersonId2 <T> : Person2<T>
{
    private T id;
    public T Id
    {
        set
        {
            id = value;
        }
        get
        {
            return id;
        }
    }
    public PersonId2(string name, T info, T id) :
    base(name, info)
    {
        this.id = id;
    }
}
```

## Класс PersonId3

```
class PersonId3 : Person2<char>
{
    private string id;
    public string Id
    {
        set
        {
            id = value;
        }
        get
        {
            return id;
        }
    }
    public PersonId3(string name, char info, string id) :
    base(name, info)
    {
        this.id = id;
    }
}
```

## Класс PersonId4

```

class PersonId4 <P> : Person2<char>

{
    private P id;
    public P Id
    {
        set
        {
            id = value;
        }
        get
        {
            return id;
        }
    }
    public PersonId4(string name, char info, P id) :
    base(name, info)
    {
        this.id = id;
    }
}

```

## Класс PersonId5

```

class PersonId5<T, P> : Person2<T>

{
    private P id;
    public P Id
    {
        set
        {
            id = value;
        }
        get
        {
            return id;
        }
    }
    public PersonId5(string name, T info, P id) :
    base(name, info)
    {
        this.id = id;
    }
}

```

## Main

```

class Program
{
    static void Main(string[] args)
    {
        Console.WriteLine("Использование типа object");
        Person1 pr1 = new Person1("Иванов", 123);
        Person1 pr2 = new Person1("Петров", "456");
        Person1 pr3 = new Person1("Сидоров", 'f');
        Person1 pr4 = new Person1("Титов", false);
        Person1 pr5 = new Person1("Алексеев", pr1);
        Console.WriteLine("Фамилия {0}, информация {1}", pr1.Name, pr1.Info);
        Console.WriteLine("Фамилия {0}, информация {1}", pr2.Name, pr2.Info);
        Console.WriteLine("Фамилия {0}, информация {1}", pr3.Name, pr3.Info);
        Console.WriteLine("Фамилия {0}, информация {1}", pr4.Name, pr4.Info);
        Console.WriteLine("Фамилия {0}, информация {1}", pr5.Name, pr5.Info);

        pr1.Info = (int)pr1.Info + 1;
        pr2.Info = (string)pr2.Info + 1;
        pr3.Info = (char)pr3.Info + 1;
        pr4.Info = !((bool)pr4.Info);
        pr5.Info = pr5.Info.ToString();

        Console.WriteLine("\n");

        Console.WriteLine("Фамилия {0}, информация {1}", pr1.Name, pr1.Info);
        Console.WriteLine("Фамилия {0}, информация {1}", pr2.Name, pr2.Info);
        Console.WriteLine("Фамилия {0}, информация {1}", pr3.Name, pr3.Info);
        Console.WriteLine("Фамилия {0}, информация {1}", pr4.Name, pr4.Info);
        Console.WriteLine("Фамилия {0}, информация {1}", pr5.Name, pr5.Info);

        Console.WriteLine("\n");

        Console.WriteLine("Использование обобщенного класса");
        Person2<int> pr6 = new Person2<int>("Иванов", 123);
        Person2<string> pr7 = new Person2<string>("Петров", "456");
        Person2<char> pr8 = new Person2<char>("Сидоров", 'f');
        Person2<bool> pr9 = new Person2<bool>("Титов", false);
        Person2<object> pr10 = new Person2<object>("Алексеев", pr1);
        Console.WriteLine("Фамилия {0}, информация {1}", pr6.Name, pr6.Info);
        Console.WriteLine("Фамилия {0}, информация {1}", pr7.Name, pr7.Info);
        Console.WriteLine("Фамилия {0}, информация {1}", pr8.Name, pr8.Info);
        Console.WriteLine("Фамилия {0}, информация {1}", pr9.Name, pr9.Info);
        Console.WriteLine("Фамилия {0}, информация {1}", pr10.Name, pr10.Info);

        Console.WriteLine("\n");

        pr6.Info = pr6.Info + 1;
        pr7.Info = pr7.Info + 1;
        pr8.Info = (char)(pr8.Info + 1);
        pr9.Info = !pr9.Info;
        pr10.Info = pr10.Info.ToString();
        Console.WriteLine("Фамилия {0}, информация {1}", pr6.Name, pr6.Info);
    }
}

```

```

Console.WriteLine("Фамилия {0}, информация {1}", pr7.Name, pr7.Info);
Console.WriteLine("Фамилия {0}, информация {1}", pr8.Name, pr8.Info);
Console.WriteLine("Фамилия {0}, информация {1}", pr9.Name, pr9.Info);
Console.WriteLine("Фамилия {0}, информация {1}", pr10.Name, pr10.Info);

Console.WriteLine("\n");

Console.WriteLine("Использование нескольких параметров");
Person3<int, int> pr11 = new Person3<int, int>("Иванов", 123, 101);
Person3<string, int> pr12 = new Person3<string, int>("Петров", "456", 202);
Person3<char, string> pr13 = new Person3<char, string>("Сидоров", 'f', "a12");

Console.WriteLine("Фамилия {0}, инфо {1}, id {2}", pr11.Name, pr11.Info, pr11.Id);
Console.WriteLine("Фамилия {0}, инфо {1}, id {2}", pr12.Name, pr12.Info, pr12.Id);
Console.WriteLine("Фамилия {0}, инфо {1}, id {2}", pr13.Name, pr13.Info, pr13.Id);

Console.WriteLine("\n");

Console.WriteLine("Обобщенный класс как обобщенный тип");
Game<Person2<int>> gm1 = new Game<Person2<int>>("КГАСУ", new
Person2<int>("Иванов", 123));
    Console.WriteLine("Организация участника {0}, фамилия {1}, информация {2}",
gm1.Company, gm1.Member.Name, gm1.Member.Info);

Console.WriteLine("\n");

Console.WriteLine("Статические члены");
Person2<int>.Code = 765;
Person2<string>.Code = "Hello!";
Person2<char>.Code = 'w';
Person2<bool>.Code = true;
Console.WriteLine("Статическое поле имеет значение:\n для int {0} \n для string {1} \n
для char {2} \n для bool {3}", Person2<int>.Code, Person2<string>.Code, Person2<char>.Code,
Person2<bool>.Code);

Console.WriteLine("\n");

Console.WriteLine("Обобщенный метод");
int a = 5;
int b = 7;
Sw<int>(ref a, ref b);
Console.WriteLine("Для чисел a = {0}, b = {1}", a, b);
string u = "Tom";
string v = "Bob";
Sw<string>(ref u, ref v);
Console.WriteLine("Для строк u = {0}, v = {1}", u, v);
char s = 'x';
char t = 'y';
Sw<char>(ref s, ref t);
Console.WriteLine("Для символов s = {0}, t = {1}", s, t);
bool p = true;
bool q = false;

```

```

Sw<bool>(ref p, ref q);
Console.WriteLine("Для логического типа p = {0}, q = {1}",
    p, q);

Console.WriteLine("\n");

Console.WriteLine("Использование первого класса");
PersonId1<string> prid1 =
    new PersonId1<string>("Иванов", "123", 321);
Console.WriteLine("До преобразования info = {0}",
    prid1.Info);
prid1.Info = prid1.Info + 1;
Console.WriteLine("После преобразования info = {0}",
    prid1.Info);

Console.WriteLine("\n Использование второго класса");
PersonId2<string> prid2 = new PersonId2<string>("Иванов", "123", "321");
Console.WriteLine("До преобразования info = {0}, id = {1}", prid2.Info, prid2.Id);
prid2.Info = prid2.Info + 1;
prid2.Id = prid2.Id + 1;
Console.WriteLine("После преобразования info = {0}, id = {1}", prid2.Info, prid2.Id);

Console.WriteLine("\n Использование третьего класса");
PersonId3 prid3 = new PersonId3("Иванов", 'g', "321");
Console.WriteLine("До преобразования info = {0}, id = {1}", prid3.Info, prid3.Id);
prid3.Info = (char)(prid3.Info + 1);
prid3.Id = prid3.Id + 1;
Console.WriteLine("После преобразования info = {0}, id = {1}", prid3.Info, prid3.Id);

Console.WriteLine("\n Использование четвертого класса");
PersonId4<int> prid4 = new PersonId4<int>("Иванов", 'g', 321);
Console.WriteLine("До преобразования info = {0}, id = {1}", prid4.Info, prid4.Id);
prid4.Info = (char)(prid4.Info + 1);
prid4.Id = prid4.Id + 1;
Console.WriteLine("После преобразования info = {0}, id = {1}", prid4.Info, prid4.Id);

Console.WriteLine("\n Использование пятого класса");
PersonId5<int, string> prid5 = new PersonId5<int, string>("Иванов", 123, "321");
Console.WriteLine("До преобразования info = {0}, id = {1}", prid5.Info, prid5.Id);
prid5.Info = prid5.Info + 1;
prid5.Id = prid5.Id + 1;
Console.WriteLine("После преобразования info = {0}, id = {1}", prid5.Info, prid5.Id);

Console.WriteLine("\n ");

Console.WriteLine("Использование ограничений на классы");
Game1 < Person1 > gm2 = new Game1<Person1>("КГАСУ", new Person1("Иванов",
123));
gm2.PrName();

Game1<PersonId> gm3 = new Game1<PersonId>("КГАСУ", new PersonId("Петров",
"345", 567));

```

```

gm3.PrName();

Console.WriteLine("\n ");

Console.WriteLine("Использование ограничений на методы");
Person1 pr14 = new Person1("Николаев", 423);
PersonId pr15 = new PersonId("Михайлов", "543", 678);
G(pr14);
G(pr15);

Console.WriteLine("\n ");
string a1 = "хорошо", b1 = "плохо";
SwCl<string>(ref a1, ref b1);
Console.WriteLine("Первое = {0}, второе = {1}", a1, b1);

Console.WriteLine("\n ");
int a2 = 2, b2 = 5;
SwSt<int>(ref a2, ref b2);
Console.WriteLine("Первое = {0}, второе = {1}", a2, b2);
}
static void Sw<T>(ref T x, ref T y)
{
    T z = x;
    x = y;
    y = z;
}
static void G<T>(T x) where T : Person1
{
    Console.WriteLine("Фамилия {0}, информация {1}", x.Name, x.Info);
}
static void SwCl<T>(ref T x, ref T y) where T : class
{
    T z = x;
    x = y;
    y = z;
}
static void SwSt<T>(ref T x, ref T y) where T : struct
{
    T z = x;
    x = y;
    y = z;
} }

```

C:\Windows\system32\cmd.exe

Использование типа object

фамилия Иванов, информация 123  
фамилия Петров, информация 456  
фамилия Сидоров, информация f  
фамилия Титов, информация False  
фамилия Алексеев, информация \_6проект.Person1

фамилия Иванов, информация 124  
фамилия Петров, информация 4561  
фамилия Сидоров, информация 103  
фамилия Титов, информация True  
фамилия Алексеев, информация \_6проект.Person1

Использование обобщенного класса

фамилия Иванов, информация 123  
фамилия Петров, информация 456  
фамилия Сидоров, информация f  
фамилия Титов, информация False  
фамилия Алексеев, информация \_6проект.Person1

фамилия Иванов, информация 124  
фамилия Петров, информация 4561  
фамилия Сидоров, информация g  
фамилия Титов, информация True  
фамилия Алексеев, информация \_6проект.Person1

Использование нескольких параметров

фамилия Иванов, инфо 123, id 101  
фамилия Петров, инфо 456, id 202  
фамилия Сидоров, инфо f, id a12



Обобщенный класс как обобщенный тип  
Организация участника КГАСУ, фамилия Иванов, информация 123

Статические члены  
Статическое поле имеет значение:  
для int 765  
для string Hello!  
для char w  
для bool True

Обобщенный метод  
Для чисел a = 7, b = 5  
Для строк u = Bob, v = Tom  
Для символов s = y, t = x  
Для логического типа p = False, q = True

Использование первого класса  
До преобразования info = 123  
После преобразования info = 1231

Использование второго класса  
До преобразования info = 123, id = 321  
После преобразования info = 1231, id = 3211

Использование третьего класса  
До преобразования info = g, id = 321  
После преобразования info = h, id = 3211

Использование четвертого класса  
До преобразования info = g, id = 321  
После преобразования info = h, id = 322

Использование пятого класса  
До преобразования info = 123, id = 321  
После преобразования info = 124, id = 3211

Использование ограничений на классы  
фамилия участника Иванов  
фамилия участника Петров

Использование ограничений на методы  
фамилия Николаев, информация 423  
фамилия Михайлов, информация 543

Первое = плохо, второе = хорошо

Первое = 5, второе = 2  
Для продолжения нажмите любую клавишу . . .

## Проект №7

### Main

```
static void Main()
{
    Student st1 = new Student("Иванов", 203, 5, 4, 3);
    Console.WriteLine(st1[1]);
    st1[1] = 5;
    Console.WriteLine(st1.Fiz);

    Console.WriteLine("Оценка по математике {0}", st1["Математика"]);
    st1["Математика"] = 4;
    Console.WriteLine("Оценка по математике {0}", st1[0]);

    Student st2 = new Student("Иванов", 203, 5, 4, 3);
    Student st3 = new Student("Петров", 204, 3, 4, 4);
    Student st4 = new Student("Сидоров", 201, 4, 5, 4);
    Student st5 = new Student("Алексеев", 202, 5, 5, 4);
    Student st6 = new Student("Титов", 201, 4, 5, 5);
    Student[] stud1 = new Student[] { st2, st3, st4, st5, st6 };
    double s_mat = 0;
    for (int k = 0; k < stud1.Length; k++)
        s_mat += stud1[k]["Математика"];
    s_mat /= stud1.Length;
    Console.WriteLine("Средний балл по математике равен {0}", Math.Round(s_mat, 2));

    Students studs = new Students(stud1);
    for (int k = 0; k < stud1.Length; k++)
        Console.WriteLine("Студент {0}, группа {1}", studs[k].Name, studs[k].Grup);
    Console.WriteLine("Студент {0}, средний балл {1}", studs["Петров"].Name,
        studs["Петров"].Sr_ball());

    Matrix1 mat = new Matrix1();
    mat[1, 2] = 10;
    Console.WriteLine("Первый = {0}, последний = {1}", mat[0, 0], mat[1, 2]);
}
```

### Класс Matrix1

```
class Matrix1
{
    private int[,] m = new int[,] { { 1, 2, 3 }, { 4, 5, 6 } };
    public int this[int i, int j]
    {
        set { m[i, j] = value; }
        get { return m[i, j]; }
    }
}
```

### Класс Student

```
class Student
{

```

```

public string Name { get; set; }
public int Grup { set; get; }
public int Mat { set; get; }
public int Fiz { set; get; }
public int Him { set; get; }
public Student(string name, int grup, int mat, int fiz, int him)
{
    Name = name;
    Grup = grup;
    Mat = mat;
    Fiz = fiz;
    Him = him;
}
public double Sr_ball() =>
Math.Round((Mat + Fiz + Him) / 3.0, 2);
public int this[int k]
{
    set
    {
        switch (k)
        {
            case 0:
                Mat = value;
                break;
            case 1:
                Fiz = value;
                break;
            case 2:
                Him = value;
                break;
            default:
                throw
                new Exception("Некорректный номер!");
        }
    }
    get
    {
        switch (k)
        {
            case 0:
                return Mat;
            case 1:
                return Fiz;
            case 2:
                return Him;
            default:
                throw
                new Exception("Некорректный номер!");
        }
    }
}
public int this[string s]

```

```

{
    set
    {
        switch (s)
        {
            case "Математика":
                Mat = value;
                break;
            case "Физика":
                Fiz = value;
                break;
            case "Химия":
                Him = value;
                break;
            default:
                throw new Exception("Некорректный индекс!");
        }
    }
    get
    {
        switch (s)
        {
            case "Математика":
                return Mat;
            case "Физика":
                return Fiz;
            case "Химия":
                return Him;
            default:
                throw new Exception("Некорректный индекс!");
        }
    }
}

```

## Класс Students

```

class Students
{
    private Student[] stud;
    public Students(Student[] stud)
    {
        this.stud = stud;
    }
    public Student this[int k]
    {
        set { stud[k] = value; }
        get { return stud[k]; }
    }
    public Student this[string s]

```

```

{
    get
    {
        foreach (Student st in stud)
        {
            if (st.Name == s)
                return st;
        }
        throw new Exception("Нет такого студента!");
    }
}

```

C:\Windows\system32\cmd.exe

```

4
5
Оценка по математике 5
Оценка по математике 4
Средний балл по математике равен 4,2
Студент Иванов, группа 203
Студент Петров, группа 204
Студент Сидоров, группа 201
Студент Алексеев, группа 202
Студент Титов, группа 201
Студент Петров, средний балл 3,67
Первый = 1, последний = 10
Для продолжения нажмите любую клавишу . . .

```

## Проект №8

### Main

```
static void Main()
{
    Console.WriteLine("Список");
    Student st1 = new Student("Иванов", 203, 5, 4, 3);
    Student st2 = new Student("Петров", 204, 3, 4, 4);
    Student st3 = new Student("Сидоров", 201, 4, 5, 4);
    Student st4 = new Student("Алексеев", 202, 5, 5, 4);
    Student st5 = new Student("Титов", 201, 4, 5, 5);
    List<Student> list1 = new List<Student> {st1, st2, st3};
    Console.WriteLine("Студент {0}, средний балл {1} \n\nAdd and insert", list1[2].Name,
list1[2].Sr_ball());

    list1.Add(st4);
    list1.Insert(2, st5);
    foreach (var x in list1)
        Console.WriteLine("Студент {0} средний балл {1}", x.Name, x.Sr_ball());

    Console.WriteLine("\nReverse");
    list1.Reverse();
    for (int k = 0; k < list1.Count; k++)
        Console.WriteLine("Студент {0} средний балл {1}", list1[k].Name, list1[k].Sr_ball());

    Console.WriteLine("\nRemove and RemoveAt");
    if (list1.Remove(st2))
        Console.WriteLine("Элемент успешно удален");
    else
        Console.WriteLine("Элемент не удален");
    if (list1.Remove(st2))
        Console.WriteLine("Элемент успешно удален");
    else
        Console.WriteLine("Элемент не удален");
    list1.RemoveAt(2);
    for (int k = 0; k < list1.Count; k++)
        Console.WriteLine("Студент {0} средний балл {1}", list1[k].Name, list1[k].Sr_ball());

    Console.WriteLine("\nContains");
    Console.WriteLine("st1 содержится {0}", list1.Contains(st1));
    Console.WriteLine("st2 содержится {0}", list1.Contains(st2));

    Console.WriteLine("\nIndexOf and LastIndexOf");
    list1.Add(st3);
    list1.Add(st2);
    list1.Add(st3);
    list1.Add(st5);
    Console.WriteLine("Первое вхождение {0} под номером {1}", st3.Name,
list1.IndexOf(st3));
```

```

    Console.WriteLine("Последнее вхождение {0} под номером {1}", st3.Name,
list1.LastIndexOf(st3));
    for (int k = 0; k < list1.Count; k++)
        Console.WriteLine("Студент {0}", list1[k].Name);

    Console.WriteLine("\nClear");
    list1.Clear();
    Console.WriteLine("Список содержит {0} элементов", list1.Count);

    Console.WriteLine("\nОчередь");
    Student[] studs = {st1, st2, st3, st4};
    Queue<Student> queue1 = new Queue<Student>(studs);
    Console.WriteLine("В очереди {0} элементов\n", queue1.Count);
    foreach (var x in queue1)
        Console.WriteLine("Студент {0} ", x.Name);

    Console.WriteLine();
    queue1.Enqueue(st5);
    Console.WriteLine("В очереди {0} элементов\n", queue1.Count);
    foreach (var x in queue1)
        Console.WriteLine("Студент {0}", x.Name);

    Console.WriteLine();
    Student stQueue = queue1.Peek();
    Console.WriteLine("Получен студент {0} без удаления из очереди", stQueue.Name);
    Console.WriteLine();

    Console.WriteLine("st1 содержится {0}", queue1.Contains(st1));
    Console.WriteLine();

    Console.WriteLine("В очереди {0} элементов", queue1.Count);
    Console.WriteLine();

    while (queue1.Count > 0)
    {
        var x = queue1.Dequeue();
        Console.WriteLine("Студент {0} средний балл {1}",
            x.Name, x.Sr_ball());
    }
    Console.WriteLine("\nВ очереди {0} элементов", queue1.Count);

    Console.WriteLine("\nСтек\n");

    Stack<Student> stack1 = new Stack<Student>(studs);
    Console.WriteLine("В стеке {0} элементов", stack1.Count);
    Console.WriteLine();

    foreach (var x in stack1)
        Console.WriteLine("Студент {0}", x.Name);
    Console.WriteLine();

    stack1.Push(st5);

```

```

Console.WriteLine("В стеке {0} элементов", stack1.Count);
Console.WriteLine();

foreach (var x in stack1)
    Console.WriteLine("Студент {0}", x.Name);
Console.WriteLine();

Student stStack = stack1.Peek();
Console.WriteLine("Получен студент {0} без удаления из стека", stStack.Name);
Console.WriteLine();

Console.WriteLine("st5 содержится {0}", stack1.Contains(st5));
Console.WriteLine();

Console.WriteLine("В стеке {0} элементов", stack1.Count);
Console.WriteLine();

while (stack1.Count > 0)
{
    var x = stack1.Pop();
    Console.WriteLine("Студент {0} средний балл {1}",
        x.Name, x.Sr_ball());
}
Console.WriteLine();
Console.WriteLine("В стеке {0} элементов", stack1.Count);

Console.WriteLine("\nСловарь\n");
Dictionary<string, Student> dictionary1 = new Dictionary<string, Student>
{
    ["КГАСУ"] = st1,
    ["КФУ"] = st2,
    ["КГТУ"] = st3
};
Console.WriteLine("В словаре {0} элементов", dictionary1.Count);
foreach (var x in dictionary1)
{
    Console.WriteLine("Ключ {0} студент {1}", x.Key, x.Value.Name);
}

dictionary1.Add("НГПУ", st4);
dictionary1.Add("КЮИ", st5);

Console.WriteLine("\nAdd");
Console.WriteLine("В словаре {0} элементов", dictionary1.Count);
foreach (var x in dictionary1)
{
    Console.WriteLine("Ключ {0} студент {1}", x.Key, x.Value.Name);
}

Console.WriteLine();
Console.WriteLine("Студент {0} средний балл {1}", dictionary1["КФУ"].Name,
    dictionary1["КФУ"].Sr_ball());

```



```

Console.WriteLine("Update");
dictionary1["КФУ"].Name = "Николаев";
Console.WriteLine("Студент {0} средний балл {1}", dictionary1["КФУ"].Name,
dictionary1["КФУ"].Sr_ball());

Console.WriteLine("\nRemove");
if (dictionary1.Remove("КФУ"))
    Console.WriteLine("Элемент удален!");
else
    Console.WriteLine("Элемент не удален!");
Console.WriteLine("В словаре {0} элементов", dictionary1.Count);

Console.WriteLine("\nContainsKey and ContainsValue");
Console.WriteLine("Элемент содержится в словаре {0}",
dictionary1.ContainsKey("МГУ"));
Console.WriteLine("Элемент содержится в словаре {0}", dictionary1.ContainsValue(st5));
}

```

## Класс Student

```

class Student
{
public string Name { get; set; }
public int Grup { set; get; }
public int Mat { set; get; }
public int Fiz { set; get; }
public int Him { set; get; }
public Student(string name, int grup, int mat, int fiz, int him)
{
    Name = name;
    Grup = grup;
    Mat = mat;
    Fiz = fiz;
    Him = him;
}
public double Sr_ball() => Math.Round((Mat + Fiz + Him) / 3.0, 2);
public int this[int k]
{
    set
    {
        switch (k)
        {
            case 0:
                Mat = value;
                break;
            case 1:
                Fiz = value;
                break;

```

```


        case 2:
            Him = value;
            break;
        default:
            throw
            new Exception("Некорректный номер!");
    }
}
get
{
    switch (k)
    {
        case 0:
            return Mat;
        case 1:
            return Fiz;
        case 2:
            return Him;
        default:
            throw
            new Exception("Некорректный номер!");
    }
}
}
public int this[string s]
{
    set
    {
        switch (s)
        {
            case "Математика":
                Mat = value;
                break;
            case "Физика":
                Fiz = value;
                break;
            case "Химия":
                Him = value;
                break;
            default:
                throw new Exception("Некорректный индекс!");
        }
    }
}
get
{
    switch (s)
    {
        case "Математика":
            return Mat;
        case "Физика":
            return Fiz;
        case "Химия":

```

```

        return Him;
    default:
        throw new Exception("Некорректный индекс!"); }}}

```

 C:\Windows\system32\cmd.exe

```

Список
Студент Сидоров, средний балл 4,33

Add and Insert
Студент Иванов средний балл 4
Студент Петров средний балл 3,67
Студент Титов средний балл 4,67
Студент Сидоров средний балл 4,33
Студент Алексеев средний балл 4,67

Reverse
Студент Алексеев средний балл 4,67
Студент Сидоров средний балл 4,33
Студент Титов средний балл 4,67
Студент Петров средний балл 3,67
Студент Иванов средний балл 4

Remove and RemoveAt
Элемент успешно удален
Элемент не удален
Студент Алексеев средний балл 4,67
Студент Сидоров средний балл 4,33
Студент Иванов средний балл 4

Contains
st1 содержится True
st2 содержится False

IndexOf and LastIndexOf
Первое вхождение Сидоров под номером 1
Последнее вхождение Сидоров под номером 5
Студент Алексеев
Студент Сидоров
Студент Иванов
Студент Сидоров
Студент Петров
Студент Сидоров
Студент Титов

Clear
Список содержит 0 элементов

Очередь
В очереди 4 элементов

Студент Иванов
Студент Петров
Студент Сидоров
Студент Алексеев

В очереди 5 элементов

Студент Иванов
Студент Петров
Студент Сидоров
Студент Алексеев
Студент Титов

Получен студент Иванов без удаления из очереди

st1 содержится True

В очереди 5 элементов

Студент Иванов средний балл 4
Студент Петров средний балл 3,67
Студент Сидоров средний балл 4,33
Студент Алексеев средний балл 4,67
Студент Титов средний балл 4,67

В очереди 0 элементов

```

```

Стек

В стеке 4 элементов

Студент Алексеев
Студент Сидоров
Студент Петров
Студент Иванов

В стеке 5 элементов

Студент Титов
Студент Алексеев
Студент Сидоров
Студент Петров
Студент Иванов

Получен студент Титов без удаления из стека

st5 содержится True

В стеке 5 элементов

Студент Титов средний балл 4,67
Студент Алексеев средний балл 4,67
Студент Сидоров средний балл 4,33
Студент Петров средний балл 3,67
Студент Иванов средний балл 4

В стеке 0 элементов

Словарь

В словаре 3 элементов
Ключ КГАСУ студент Иванов
Ключ КФУ студент Петров
Ключ КГТУ студент Сидоров

Add
В словаре 5 элементов
Ключ КГАСУ студент Иванов
Ключ КФУ студент Петров
Ключ КГТУ студент Сидоров
Ключ НГПУ студент Алексеев
Ключ КЮИ студент Титов

Студент Петров средний балл 3,67
Update
Студент Николаев средний балл 3,67

Remove
Элемент удален!
В словаре 4 элементов

ContainsKey and ContainsValue
Элемент содержится в словаре False
Элемент содержится в словаре True
Для продолжения нажмите любую клавишу . . . █

```

## Проект №9

```
class Program
{
    delegate void Del();
    delegate int Fun(int x, int y);
    delegate T Pr<T, P, K>(P x, K y);
    delegate int St(ref int x);
    static int t1(ref int x)
    {
        x += 2;
        return x;
    }
    static int t2(ref int x)
    {
        x += 3;
        return x;
    }
    static int t3(ref int x)
    {
        x += 4;
        return x;
    }
    static void f1() => Console.WriteLine("Привет!");
    static int g1(int a, int b) => a + b;
    static int g2(int a, int b) => a * b;
    static void Operator(int a, int b, Fun f) => Console.WriteLine("Первое число {0}, второе  
число {1}, результат {2}", a, b, f(a, b));
    static Del dd(int k)
    {
        switch (k)
        {
            case 1: return f1;
            case 2: return Test.f2;
            default: return new Test().f3;
        }
    }
    static Func<int, int, int> a2(int x)
    {
        if (x > 0)
            return g1;
        else
            return g2;
    }
    static Del dd2(int k)
    {
        switch (k)
        {
            case 1:
                return delegate
                { Console.WriteLine("Благодарю Вас!"); };
            case 2:
```

```

        return () =>
        Console.WriteLine("Всего хорошего!");
        default: return f1;
    }
}
static void Main(string[] args)
{
    Console.WriteLine("Понятие делегата");
    Del del1 = f1;
    del1();
    del1 = Test.f2;
    del1();
    del1 = new Test().f3;
    del1();

    Fun fun1 = g1;
    Console.WriteLine("\n" + fun1(3, 4));
    fun1 = g2;
    Console.WriteLine(fun1(3, 4));
    Pr<int, int, int> pr1 = g1;
    Console.WriteLine(pr1(3, 4));

    Del del2 = f1;
    Console.WriteLine("\nДобавление методов");
    del2 += Test.f2;
    Test st = new Test();
    del2 += st.f3;
    del2 += f1;
    del2 += new Test().f3;
    del2();

    Console.WriteLine("\nУдаление методов");
    del2 -= st.f3;
    del2 -= Test.f2;
    del2 -= Test.f2;
    del2 -= new Test().f3;
    del2();

    Del del3 = f1;
    Console.WriteLine("\nДобавление методов 2");
    del3 += Test.f2;
    del3 += f1;
    del3 += new Test().f3;
    del3 += f1;
    Test st1 = new Test();
    del3 += st.f3;
    del3 += f1;
    del3();

    Console.WriteLine("\nУдаление методов 2");
    del3 -= f1;
    del3 -= f1;

```

```

del3();

Console.WriteLine("\nПроверка на null");
Del del4 = f1;
del4 += f1;
if (del4 == null)
    Console.WriteLine("null");
else
{
    del4();
    Console.WriteLine();
}
del4 -= f1;
if (del4 == null)
    Console.WriteLine("null");
else
{
    del4();
    Console.WriteLine();
}
del4 -= f1;
if (del4 == null)
    Console.WriteLine("Значение null");
else
{
    del4();
    Console.WriteLine();
}

Console.WriteLine("\nФункции");
St st2 = t1;
st2 += t2;
st2 += t3;
int a = 5;
int b = 5;
Console.WriteLine("a = {0}, st1(5) = {1}, a = {2}", a, st2(ref a), a);
Console.WriteLine("b = {0}, t3(5) = {1}", b, t3(ref b));

St st3 = t1;
St st4 = t2;
St st5 = t3;
St st6 = st3 + st4 + st5;
int a1 = 5;
Console.WriteLine("a = {0}, st4(5) = {1}, a = {2}", a1, st6(ref a1), a1);

Console.WriteLine("\nИспользование делегатов при описании методов");
Operator(4, 5, g1);
Operator(4, 5, g2);
dd(1());
dd(2());
dd(6());
Console.WriteLine("Первое {0}, второе {1}", a2(3)(3, 4), a2(-2)(3, 4));

```

```

Console.WriteLine("\nАнонимные методы");
Fun fun2 = delegate (int x, int y)
{
    return x - y;
};
Console.WriteLine(fun2(3, 4));
Del del5 = f1;
del5 += delegate
{
    Console.WriteLine("Что нового?");
    Console.WriteLine("Будьте здоровы!");
};
del5();
Operator(5, 3, delegate (int x, int y) { return x - y; });

Fun fun3 = (x, y) => x - y;
Console.WriteLine("\nЛямбда-выражения\n" + fun3(3, 4));

Del del6 = f1;
del6 += () =>
{
    Console.WriteLine("Что нового?");
    Console.WriteLine("Будьте здоровы!");
};
del6();
Operator(5, 3, (int x, int y) => x - y);
dd2(3)();
dd2(2)();
dd2(1)();

Predicate<int> isPositive = (int x) => x > 0;
Console.WriteLine("\nВстроенные делегаты\n" + isPositive(20));
Console.WriteLine(isPositive(-20));
}
}
class Test
{
    public static void f2() => Console.WriteLine("Как дела?");
    public void f3() => Console.WriteLine("До свидания!");
}

```



cmd C:\Windows\system32\cmd.exe

Понятие делегата

Привет!

Как дела?

До свидания!

7

12

7

Добавление методов

Привет!

Как дела?

До свидания!

Привет!

До свидания!

Удаление методов

Привет!

Привет!

До свидания!

Добавление методов 2

Привет!

Как дела?

Привет!

До свидания!

Привет!

До свидания!

Привет!

Удаление методов 2

Привет!

Как дела?

Привет!

До свидания!

До свидания!

Проверка на null

Привет!

Привет!

Привет!

Значение null

Функции

a = 5, st1(5) = 14, a = 14

b = 5, t3(5) = 9

a = 5, st4(5) = 14, a = 14

Использование делегатов при описании методов

Первое число 4, второе число 5, результат 9

Первое число 4, второе число 5, результат 20

Привет!

Как дела?

До свидания!

Первое 7, второе 12

Анонимные методы

-1

Привет!

Что нового?

Будьте здоровы!

Первое число 5, второе число 3, результат 2

Лямбда-выражения

-1

Привет!

Что нового?

Будьте здоровы!

Первое число 5, второе число 3, результат 2

Привет!

Всего хорошего!

Благодарю Вас!

Встроенные делегаты

True

False

Для продолжения нажмите любую клавишу . . .

## Проект №10

### Program

```
class Program
{
    static void PrFile1(string s)
    {
        StreamWriter sw = new StreamWriter(@"C:\Users\dasha\Desktop\11\1файл.txt", true,
Encoding.Default);
        sw.WriteLine(s);
        sw.Close();
    }
    static void PrFile2(string s)
    {
        StreamWriter sw = new StreamWriter(@"C:\Users\dasha\Desktop\11\2файл.txt", true,
Encoding.Default);
        sw.WriteLine(s);
        sw.Close();
    }
    static void PrCon(string s) => Console.WriteLine(s);
    static void Main(string[] args)
    {
        Account1 acc1 = new Account1("Сидоров", 48000, "000");
        Account1 acc2 = new Account1("Фиников", 6500, "999");
        Action<string> a1 = PrFile1;
        a1 += PrCon;
        acc1.Registr(a1);
        acc1.Act();
        Action<string> a2 = PrFile2;
        a2 += PrCon;
        acc2.Registr(a2);
        acc2.Act();
        Account2 acc3 = new Account2("Пряников", 36000, "888");
        Account2 acc4 = new Account2("Шилов", 4400, "666");
        acc3.Sob += a1;
        acc3.Act();
        acc4.Sob += a2;
        acc4.Act();
    }
}
```

### Account1

```
public class Account1
{
    private string name;
    private Action<string> sob;
    private int sum;
    private string password;
    public Account1(string name, int sum, string password)
    {
        this.name = name;
```

```

        this.sum = sum;
        this.password = password;
    }
    private void Add(int s)
    {
        sum += s;
        string str = string.Format("Пополнено!: остаток {0}", sum);
        sob(str);
    }
    private void Sub(int s)
    {
        if (s > sum)
        {
            string str = string.Format("Средств недостаточно!: остаток {0}", sum);
            sob(str);
        }
        else
        {
            sum -= s;
            string str = string.Format("Снято!: остаток {0}", sum);
            sob(str);
        }
    }
    public void Act()
    {
        Console.WriteLine("Клиент {0}, введите пароль!", name);
        string pw = Console.ReadLine();
        if (pw == password)
        {
            string code = "0";
            do
            {
                Console.WriteLine("Введите код операции:\n0 - завершить,\n1 - пополнить,\n2 -
снять.");
                code = Console.ReadLine();
                if (code == "1" || code == "2")
                {
                    Console.WriteLine("Введите сумму:");
                    int s = Int32.Parse(Console.ReadLine());
                    if (code == "1") Add(s);
                    else Sub(s);
                }
            }
            while (code != "0");
        }
        else Console.WriteLine("Неправильный пароль!");
    }
    public void Registr(Action<string> x) => sob = x;
}

```

```


class Account2
{
    public event Action<string> Sob;
    private string name;
    private int sum;
    private string password;
    public Account2(string name, int sum, string password)
    {
        this.name = name;
        this.sum = sum;
        this.password = password;
    }
    private void Add(int s)
    {
        sum += s;
        string str = string.Format("Пополнено!: остаток {0}", sum);
        Sob(str);
    }
    private void Sub(int s)
    {
        if (s > sum)
        {
            string str = string.Format("Средств недостаточно!: остаток {0}", sum);
            Sob(str);
        }
        else
        {
            sum -= s;
            string str = string.Format("Снято!: остаток {0}", sum);
            Sob(str);
        }
    }
    public void Act()
    {
        Console.WriteLine("Клиент {0}, введите пароль!", name);
        string pw = Console.ReadLine();
        if (pw == password)
        {
            string code = "0";
            do
            {
                Console.WriteLine("Введите код операции:\n0 - завершить,\n1 - пополнить,\n2 -
снять.");
                code = Console.ReadLine();
                if (code == "1" || code == "2")
                {
                    Console.WriteLine("Введите сумму:");
                    int s = Int32.Parse(Console.ReadLine());
                    if (code == "1") Add(s);
                    else Sub(s);
                }
            }
        }
    }
}

```

```

        while (code != "0");
    }
    else Console.WriteLine("Неправильный пароль!");
}
}
}

```

 1файл.txt – Блокнот

Файл Правка Формат Вид Справка

Пополнено!: остаток 48444


Средств недостаточно!: остаток 48444

Снято!: остаток 48101

Пополнено!: остаток 40033

Средств недостаточно!: остаток 40033

Снято!: остаток 39478

 2файл.txt – Блокнот

Файл Правка Формат Вид Справка

Пополнено!: остаток 8500



Средств недостаточно!: остаток 8500

Снято!: остаток 3500

Пополнено!: остаток 8964

Средств недостаточно!: остаток 8964

Снято!: остаток 8619

 C:\Windows\system32\cmd.exe Клиент Сидоров, введите пароль! 000 Введите код операции: 0 - завершить, 1 - пополнить, 2 - снять. 1 Введите сумму: 444 Пополнено!: остаток 48444 Введите код операции: 0 - завершить, 1 - пополнить, 2 - снять. 2 Введите сумму: 4444444 Средств недостаточно!: остаток 48444 Введите код операции: 0 - завершить, 1 - пополнить, 2 - снять. 2 Введите сумму: 343 Снято!: остаток 48101 Введите код операции: 0 - завершить, 1 - пополнить, 2 - снять. 0 Клиент Фиников, введите пароль! 999 Введите код операции: 0 - завершить, 1 - пополнить, 2 - снять. 1 Введите сумму: 2000 Пополнено!: остаток 8500 Введите код операции: 0 - завершить, 1 - пополнить, 2 - снять. 2 Введите сумму: 60000 Средств недостаточно!: остаток 8500 Введите код операции: 0 - завершить, 1 - пополнить, 2 - снять. 2 Введите сумму: 5000 Снято!: остаток 3500 Введите код операции: 0 - завершить, 1 - пополнить, 2 - снять. 0	 Выбрать C:\Windows\system32\cmd Клиент Пряников, введите пароль! 888 Введите код операции: 0 - завершить, 1 - пополнить, 2 - снять. 1 Введите сумму: 4033 Пополнено!: остаток 40033 Введите код операции: 0 - завершить, 1 - пополнить, 2 - снять. 2 Введите сумму: 34343434 Средств недостаточно!: остаток 40033 Введите код операции: 0 - завершить, 1 - пополнить, 2 - снять. 2 Введите сумму: 555 Снято!: остаток 39478 Введите код операции: 0 - завершить, 1 - пополнить, 2 - снять. 0 Клиент Шилов, введите пароль! 666 Введите код операции: 0 - завершить, 1 - пополнить, 2 - снять. 1 Введите сумму: 4564 Пополнено!: остаток 8964 Введите код операции: 0 - завершить, 1 - пополнить, 2 - снять. 2 Введите сумму: 3300000 Средств недостаточно!: остаток 8964 Введите код операции: 0 - завершить, 1 - пополнить, 2 - снять. 2 Введите сумму: 345 Снято!: остаток 8619 Введите код операции: 0 - завершить, 1 - пополнить, 2 - снять. 0
--	--

## Проект №11

### Program

```
internal class Program
{
    static void PrFile1(string s)
    {
        StreamWriter sw = new StreamWriter(@"C:\Users\dasha\Desktop\11\1.txt", true,
Encoding.Default);
        sw.WriteLine(s);
        sw.Close();
    }
    static void PrFile2(string s)
    {
        StreamWriter sw = new StreamWriter(@"C:\Users\dasha\Desktop\11\2.txt", true,
Encoding.Default);
        sw.WriteLine(s);
        sw.Close();
    }
    static void PrCon(string s) => Console.WriteLine(s);
    static void Main(string[] args)
    {
        Account3 acc1 = new Account3("Pavlov", 20, "123");
        acc1.Sob += PrCon;
        acc1.Sob += PrFile1;
        acc1.Act();
        acc1.Sob -= PrFile1;
        acc1.Sob += PrFile2;
        acc1.Act();
    }
}
```

### Account

```
internal class Account3
{
    private string name;
    private int sum;
    private string password;
    private Action<string> sob;
    public Account3(string bame, int sum, string password)
    {
        this.name = name;
        this.sum = sum;
        this.password = password;
    }
    private void Add(int s)
    {
        sum += s;
        string str = string.Format("Пополнено {0} руб. Остаток: {1}", s, sum);
        sob(str);
    }
}
```



```

private void Sub(int s)
{
    if (s > sum)
    {
        string str = string.Format("Недостаточно средств для снятия {0} руб! Остаток: {1}",
s, sum);
        sob(str);
    }
    else
    {
        sum -= s;
        string str = string.Format("Снято {0} руб! Остаток: {1}", s, sum);
        sob(str);
    }
}
}
public event Action<string> Sob
{
    add
    {
        Console.WriteLine("\nКлиент {0} введите пароль для добавления метода", name);
        string pw = Console.ReadLine();
        if (pw == password) {
            sob += value;
            Console.WriteLine("Добавляем метод {0}", value.Method.Name);
        }
        else
        {
            Console.WriteLine("Неправильный пароль!");
        }
    }
    remove
    {
        Console.WriteLine("Клиент {0} введите пароль для удаления метода", name);
        string pw = Console.ReadLine();
        if (pw == password)
        {
            sob -= value;
            Console.WriteLine("Удаляем метод {0}", value.Method.Name);
        }
        else
        {
            Console.WriteLine("Неправильный пароль!");
        }
    }
}
}
public void Act()
{
    Console.WriteLine("\nКлиент {0}, введите пароль для выполнения операции", name);
    string pw = Console.ReadLine();
    if (pw == password)
    {
        string code = "0";

```

```

do {
    Console.WriteLine("Введите код операции: \n0-завершение \n1-пополнение \n2-
снятие");
    code = Console.ReadLine();
    if (code == "1" || code == "2")
    {
        Console.WriteLine("Введите сумму");
        int s = Int32.Parse(Console.ReadLine());
        if (code == "1")
            Add(s);
        else
            Sub(s);
    }
} while (code != "0");
}
else {
    Console.WriteLine("Неправильный пароль!");
}
}
}

```

```
Клиент  введите пароль для добавления метода
666
Добавляем метод PrCon

Клиент  введите пароль для добавления метода
666
Добавляем метод PrFile1

Клиент , введите пароль для выполнения операции
666
Введите код операции:
0-завершение
1-пополнение
2-снятие
1
Введите сумму
3434
Пополнено 3434 руб. Остаток: 3454
Введите код операции:
0-завершение
1-пополнение
2-снятие
2
Введите сумму
333333
Недостаточно средств для снятия 333333 руб! Остаток: 3454
Введите код операции:
0-завершение
1-пополнение
2-снятие
2
Введите сумму
3000
Снято 3000 руб! Остаток: 454
Введите код операции:
0-завершение
1-пополнение
2-снятие
0
Клиент  введите пароль для удаления метода
666
Удаляем метод PrFile1

Клиент  введите пароль для добавления метода
666
Добавляем метод PrFile2
```

Клиент , введите пароль для выполнения операции  
666  
Введите код операции:  
0-завершение  
1-пополнение  
2-снятие  
1  
Введите сумму  
4564  
Пополнено 4564 руб. Остаток: 5018  
Введите код операции:  
0-завершение  
1-пополнение  
2-снятие  
2  
Введите сумму  
500000  
Недостаточно средств для снятия 500000 руб! Остаток: 5018  
Введите код операции:  
0-завершение  
1-пополнение  
2-снятие  
2  
Введите сумму  
5000  
Снято 5000 руб! Остаток: 18  
Введите код операции:  
0-завершение  
1-пополнение  
2-снятие  
0  
Для продолжения нажмите любую клавишу . . .



1.txt – Блокнот

Файл Правка Формат Вид Справка

Пополнено 3434 руб. Остаток: 3454

Недостаточно средств для снятия 333333 руб! Остаток: 3454

Снято 3000 руб! Остаток: 454



Стр 1, стлб 1

100%

Windows (CRLF)

AI



2.txt – Блокнот

Файл Правка Формат Вид Справка

Пополнено 4564 руб. Остаток: 5018

Недостаточно средств для снятия 500000 руб! Остаток: 5018

Снято 5000 руб! Остаток: 18

## Проект №12

### Program

```
class Program
{
    static void PrCon(object a, EventArgs e)
    {
        string name = ((Account4)a).Name;
        int oldSum = ((Account4Args)e).OldSum;
        int newSum = ((Account4Args)e).NewSum;
        string s = String.Format("Клиент {0}: было {1}, стало {2}", name, oldSum, newSum);
        Console.WriteLine(s);
    }
    static void PrFile1(object a, EventArgs e)
    {
        string name = ((Account4)a).Name;
        int oldSum = ((Account4Args)e).OldSum;
        int newSum = ((Account4Args)e).NewSum;
        StreamWriter sw = new StreamWriter(@"C:\Users\dasha\Desktop\1\1.txt", true,
Encoding.Default);
        string s = String.Format("Клиент {0}: было {1}, стало {2}", name, oldSum, newSum);
        sw.WriteLine(s);
        sw.Close();
    }
    static void PrFile2(object a, EventArgs e)
    {
        string name = ((Account4)a).Name;
        int oldSum = ((Account4Args)e).OldSum;
        int newSum = ((Account4Args)e).NewSum;
        StreamWriter sw = new StreamWriter(@"C:\Users\dasha\Desktop\1\2.txt", true,
Encoding.Default);
        string s = String.Format("Клиент {0}: было {1}, стало {2}", name, oldSum, newSum);
        sw.WriteLine(s);
        sw.Close();
    }
    static void Main()
    {
        Account4 acc1 = new Account4("Егорова", 80000, "1111");
        acc1.Sob += PrCon;
        acc1.Sob += PrFile1;
        acc1.Act();
        acc1.Sob -= PrFile1;
        acc1.Sob += PrFile2;
        acc1.Act();
    }
}
```

### Account4

```
class Account4
{
    private int sum;
```

```

public string Name {get;}
private string password;
private EventHandler sob;
public Account4(string name, int sum, string password)
{
    this.sum = sum;
    Name = name;
    this.password = password;
}
public event EventHandler Sob
{
    add
    {
        Console.WriteLine("Клиент {0}, введите пароль для добавления метода!", Name);
        string pw = Console.ReadLine();
        if (pw == password)
        {
            sob += value;
            Console.WriteLine("Добавлен метод " + value.Method.Name);
        }
        else
        {
            Console.WriteLine("Неверный пароль!");
        }
    }
    remove
    {
        Console.WriteLine("Клиент {0}, введите пароль для удаления метода!", Name);
        string pw = Console.ReadLine();
        if (pw == password)
        {
            sob -= value;
            Console.WriteLine("Удален метод " + value.Method.Name);
        }
        else
        {
            Console.WriteLine("Неверный пароль!");
        }
    }
}
public void Add(int s)
{
    int old = sum;
    sum += s;
    sob(this, new Account4Args(old, sum));
}
public void Sub(int s)
{
    if (s > sum)
    {
        sob(this, new Account4Args(sum, sum));
    }
}

```

```

else
{
    int old = sum;
    sum -= s;
    sob(this, new Account4Args(old, sum));
}
}
public void Act()
{
    Console.WriteLine("Клиент {0}, введите пароль для выполнения операции!", Name);
    string pw = Console.ReadLine();
    if (pw == password)
    {
        string code = "0";
        do
        {
            Console.WriteLine("Введите код операции:\n0 - завершить,\n1 - пополнить,\n2 -
снять.");
            code = Console.ReadLine();
            if (code == "1" || code == "2")
            {
                Console.WriteLine("Введите сумму:");
                int s = Int32.Parse(Console.ReadLine());
                if (code == "1") Add(s);
                else Sub(s);
            }
        }
        while (code != "0");
    }
    else Console.WriteLine("Неправильный пароль!");
}
}
}

```

## Account4Args

```

class Account4Args: EventArgs
{
    public int OldSum {get;}
    public int NewSum {get;}
    public Account4Args(int oldSum, int newSum)
    {
        OldSum = oldSum;
        NewSum = newSum;
    }
}

```

```
Консоль отладки Microsoft Visual Studio
Клиент Егорова, введите пароль для добавления метода!
1111
Добавлен метод PrCon
Клиент Егорова, введите пароль для добавления метода!
1111
Добавлен метод PrFile1
Клиент Егорова, введите пароль для выполнения операции!
1111
Введите код операции:
0 - завершить,
1 - пополнить,
2 - снять.
1
Введите сумму:
1000
Клиент Егорова: было 80000, стало 81000
Введите код операции:
0 - завершить,
1 - пополнить,
2 - снять.
2
Введите сумму:
90000
Клиент Егорова: было 81000, стало 81000
Введите код операции:
0 - завершить,
1 - пополнить,
2 - снять.
2
Введите сумму:
80000
Клиент Егорова: было 81000, стало 1000
Введите код операции:
0 - завершить,
1 - пополнить,
2 - снять.
0
```



```

Клиент Егорова, введите пароль для удаления метода!
1111
Удален метод PrFile1
Клиент Егорова, введите пароль для добавления метода!
1111
Добавлен метод PrFile2
Клиент Егорова, введите пароль для выполнения операции!
1111
Введите код операции:
0 - завершить,
1 - пополнить,
2 - снять.
1
Введите сумму:
2000
Клиент Егорова: было 1000, стало 3000
Введите код операции:
0 - завершить,
1 - пополнить,
2 - снять.
2
Введите сумму:
5000
Клиент Егорова: было 3000, стало 3000
Введите код операции:
0 - завершить,
1 - пополнить,
2 - снять.
2
Введите сумму:
2900
Клиент Егорова: было 3000, стало 100
Введите код операции:
0 - завершить,
1 - пополнить,
2 - снять.
0

```



1.txt – Блокнот

Файл Правка Формат Вид Справка

```

Клиент Егорова: было 80000, стало 81000
Клиент Егорова: было 81000, стало 81000
Клиент Егорова: было 81000, стало 1000

```

<



2.txt – Блокнот

Файл Правка Формат Вид Справка

```

Клиент Егорова: было 1000, стало 3000
Клиент Егорова: было 3000, стало 3000
Клиент Егорова: было 3000, стало 100

```

<