Министерство науки и высшего образования РФ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Новосибирский государственный технический университет»

Кафедра Автоматизированных систем управления

**Лабораторная работа №1**

по дисциплине «Программирование»

Тема: Язык C#

|  |  |
| --- | --- |
| Выполнил:  Попова Дарья Алексеевна  Группа АВТ-312  Факультет АСУ.  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  подпись  «27» декабря 2024 г. | Проверил:  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (Ф. И. О.)  Балл: \_\_\_\_\_\_\_\_\_\_  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  подпись  «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2024 г. |

Новосибирск 2024

**Оглавление**

[Постановка задачи 3](#_Toc177114782)

[Тестовые данные и ожидаемые результаты: 4](#_Toc177114783)

[Описание разработанных классов в табличном виде 5](#_Toc177114784)

[Тестирование программы 6](#_Toc177114785)

[Скриншоты результатов работы программы и выполнения тестов 7](#_Toc177114786)

[Вывод 9](#_Toc177114787)

[Приложение 10](#_Toc177114788)

**Вариант №12**

# **Постановка задачи**

|  |
| --- |
| Задание:  Взять файлы с исходным кодом программы на C++ для своего варианта и переписать программу на язык C#. Программа должна включать базовый класс "Appliance" и два производных класса: "Printer" и "Fax". Класс Program должен управлять списком устройств, позволяя добавлять, удалять и отображать информацию о каждом устройстве.  Требования:   * Реализовать базовый класс "Appliance" с методом для отображения информации. * Реализовать производные классы "Printer" и "Fax", переопределяющие метод отображения информации. * Реализовать класс "Program" для управления списком устройств. * Программа должна корректно работать с динамическими данными и обеспечивать безопасность работы с памятью. |

# **Тестовые данные и ожидаемые результаты:**

#### **Тестовые данные:**

#### Принтер: "Epson"

#### Принтер: "HP"

#### Факс: "Canon"

#### **Ожидаемые результаты:**

#### При добавлении устройств в Program, они должны корректно отображаться.

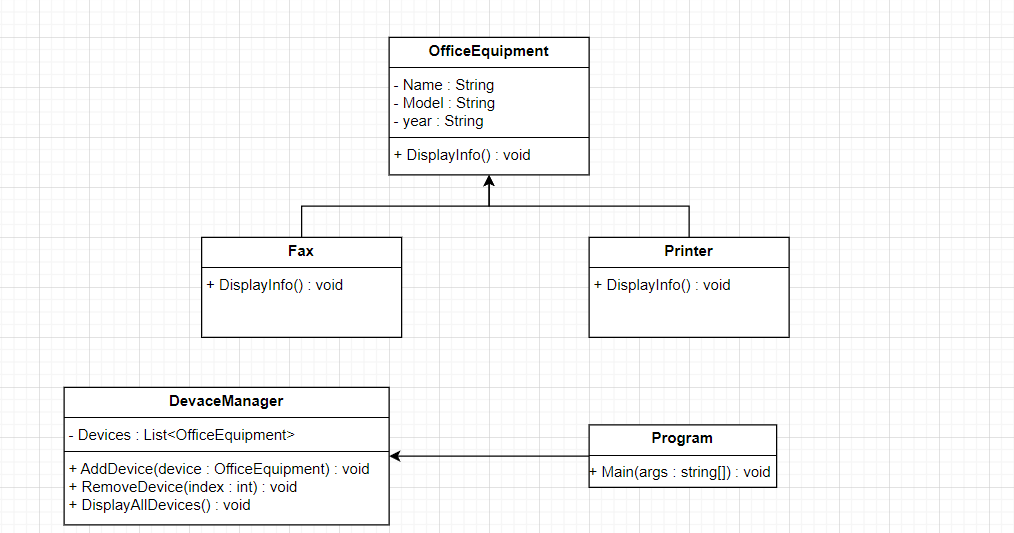
#### При удалении устройства по индексу, оно должно быть удалено из списка.

#### При отображении всех устройств, должна быть выведена информация о каждом устройстве.

# **Описание разработанных классов в табличном виде**

|  |  |  |  |
| --- | --- | --- | --- |
| Класс | Поля класса и их типы | Свойства | Методы класса, тип возвращаемого значения, списка параметров и их типов |
| Appliance | - string model  - int price  - int guarantee | -string Name {get; protected set;} | - OfficeEquipment(string model, int price, int guarantee)  - virtual void DisplayInfo() |
| Printer | -(наследует Name) | -(наследует Name) | - int GetVolume() - int GetSpeed()  - string GetTech() |
| Fax | -(наследует Name) | -(наследует Name) | - Fax(string model, int price, int guarantee, int memory, int speed, string number) - override void DisplayInfo() |
| Program | -List devices  - int choice | -List devices {get;} | - static void Main(string[] args) |

# **Диаграмма классов**



**Тестирование программы**

1. Добавление устройств:
   * Добавлен принтер "Canon".
2. Отображение всех устройств:

Ожидаемый вывод:

Модель: Canon

Цена: 15000

Гарантийный срок: 12

Объем краски: 1

Скорость печати: 1

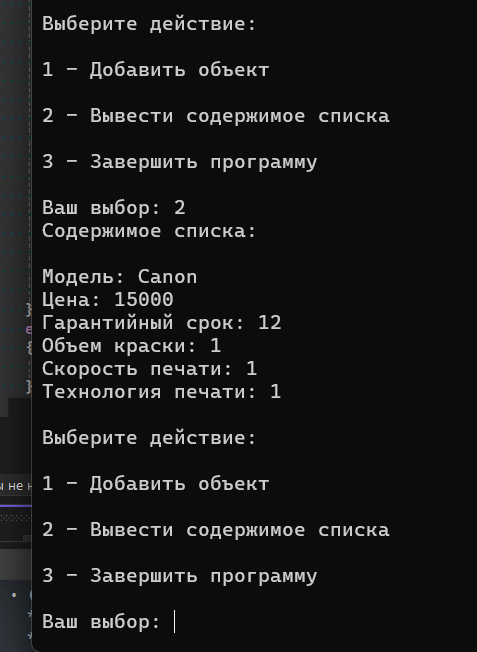
Технология печати: 1

**Результаты тестирования:**

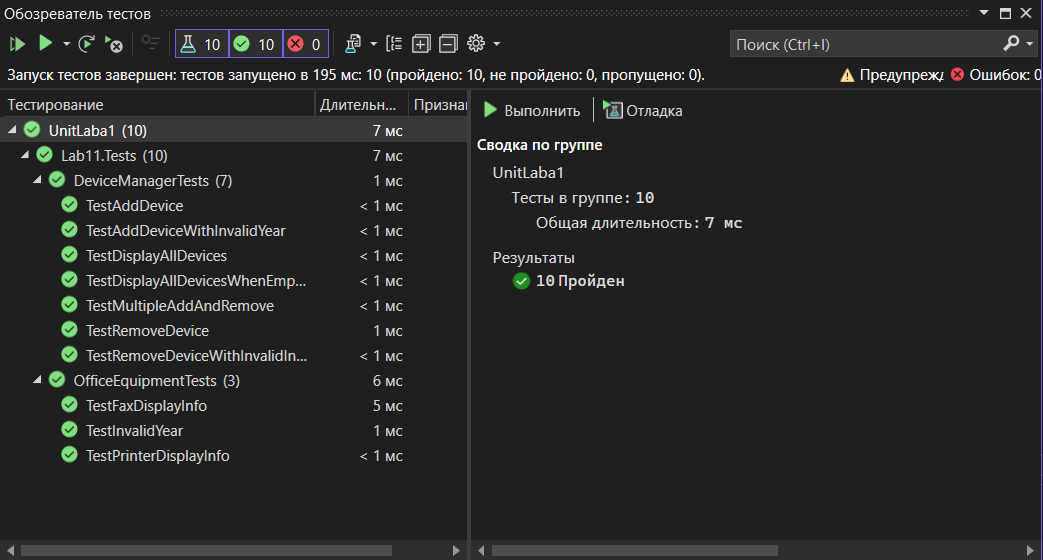
* Программа корректно добавляет устройства в список.
* Программа корректно отображает информацию о всех устройствах.

# **Скриншоты результатов работы программы и выполнения тестов**

# 



Результаты тестов:



# **Вывод**

В результате выполнения лабораторной работы была разработана программа на языке C#, которая моделирует управление офисной техникой.

Программа включает базовый класс " Appliance " и два производных класса: "Printer" и "Fax". Класс "DeviceManager" управляет списком устройств, позволяя добавлять, удалять и отображать информацию о каждом устройстве.

Программа была протестирована на корректность работы с динамическими данными и обеспечивает безопасность работы с памятью благодаря сборщику мусора в C#. Модульные тесты были написаны для всех методов классов, что позволяет убедиться в корректной работе программы.

# **Приложение**

Исходный код программы

Program.cs

using System;

using System.Collections.Generic;

public class Appliance

{

    private string model;

    private int price;

    private int guarantee;

    public Appliance()

    {

        model = null;

        price = 0;

        guarantee = 0;

    }

    public Appliance(string model, int price, int guarantee)

    {

        this.model = model;

        this.price = price;

        this.guarantee = guarantee;

    }

    public virtual void Print()

    {

        if (model != null)

            Console.WriteLine("Модель: " + model);

        Console.WriteLine("Цена: " + price);

        Console.WriteLine("Гарантийный срок: " + guarantee);

    }

    public string GetModel()

    {

        return model;

    }

    public int GetPrice()

    {

        return price;

    }

    public int GetGuarantee()

    {

        return guarantee;

    }

    public void SetModel(string name)

    {

        model = name;

    }

    public void SetPrice(int val)

    {

        if (0 < val && val < 999999)

            price = val;

    }

    public void SetGuarantee(int val)

    {

        if (0 <= val && val <= 10)

            guarantee = val;

    }

}

public class Printer : Appliance

{

    private int volumeOfPaint;

    private int printSpeed;

    private string printTechnology;

    public Printer() : base()

    {

        volumeOfPaint = 0;

        printSpeed = 0;

        printTechnology = null;

    }

    public Printer(string model, int price, int guarantee, int volume, int speed, string tech) : base(model, price, guarantee)

    {

        if (volume > 0 && volume < 100)

            volumeOfPaint = volume;

        if (speed > 0 && speed < 100)

            printSpeed = speed;

        printTechnology = tech;

    }

    public override void Print()

    {

        base.Print();

        Console.WriteLine("Объем краски: " + volumeOfPaint);

        Console.WriteLine("Скорость печати: " + printSpeed);

        if (printTechnology != null)

            Console.WriteLine("Технология печати: " + printTechnology + "\n");

    }

    public int GetVolume()

    {

        return volumeOfPaint;

    }

    public int GetSpeed()

    {

        return printSpeed;

    }

    public string GetTech()

    {

        return printTechnology;

    }

    public void SetVolume(int val)

    {

        if (val > 0 && val < 100)

            volumeOfPaint = val;

    }

    public void SetSpeed(int val)

    {

        if (val > 0 && val < 100)

            printSpeed = val;

    }

}

public class Fax : Appliance

{

    private string phoneNumber;

    private int transmissionSpeed;

    private int memoryCapacity;

    public Fax() : base()

    {

        transmissionSpeed = 0;

        memoryCapacity = 0;

        phoneNumber = null;

    }

    public Fax(string model, int price, int guarantee, int memory, int speed, string number) : base(model, price, guarantee)

    {

        if (memory > 0 && memory < 100)

            memoryCapacity = memory;

        if (speed > 0 && speed < 100)

            transmissionSpeed = speed;

        phoneNumber = number;

    }

    public override void Print()

    {

        base.Print();

        Console.WriteLine("Объем памяти: " + memoryCapacity);

        Console.WriteLine("Скорость передачи: " + transmissionSpeed);

        if (phoneNumber != null)

            Console.WriteLine("Номер: " + phoneNumber + "\n");

    }

    public int GetMemory()

    {

        return memoryCapacity;

    }

    public int GetSpeed()

    {

        return transmissionSpeed;

    }

    public string GetNumber()

    {

        return phoneNumber;

    }

    public void SetMemory(int val)

    {

        if (val > 0 && val < 100)

            memoryCapacity = val;

    }

    public void SetSpeed(int val)

    {

        if (val > 0 && val < 100)

            transmissionSpeed = val;

    }

}

public class Program

{

    public static void Main(string[] args)

    {

        Console.OutputEncoding = System.Text.Encoding.UTF8; // Для корректного отображения кириллицы в консоли

        List<Appliance> appliances = new List<Appliance>();

        int choice;

        while (true)

        {

            Console.WriteLine("Выберите действие:\n");

            Console.WriteLine("1 - Добавить объект\n");

            Console.WriteLine("2 - Вывести содержимое списка\n");

            Console.WriteLine("3 - Завершить программу\n");

            Console.Write("Ваш выбор: ");

            if (int.TryParse(Console.ReadLine(), out choice))

            {

               switch (choice)

                {

                    case 1:

                        Console.WriteLine("Какой тип объекта добавить?\n1 - Прибор\n2 - Принтер\n3 - Факс");

                        Console.Write("Ваш выбор: ");

                        if (int.TryParse(Console.ReadLine(), out int typeChoice))

                        {

                            switch (typeChoice)

                            {

                                case 1:

                                    Console.Write("Введите модель прибора: ");

                                    string model = Console.ReadLine();

                                    Console.Write("Введите цену: ");

                                     if (int.TryParse(Console.ReadLine(), out int price))

                                    {

                                        Console.Write("Введите гарантийный срок: ");

                                        if (int.TryParse(Console.ReadLine(), out int guarantee))

                                        {

                                           appliances.Add(new Appliance(model, price, guarantee));

                                           Console.WriteLine("Объект добавлен!\n");

                                        }else{

                                            Console.WriteLine("Неверный формат ввода\n");

                                            }

                                    }else{

                                            Console.WriteLine("Неверный формат ввода\n");

                                    }

                                    break;

                                case 2:

                                    Console.Write("Введите модель принтера: ");

                                    string printerModel = Console.ReadLine();

                                    Console.Write("Введите цену: ");

                                    if (int.TryParse(Console.ReadLine(), out int printerPrice))

                                    {

                                        Console.Write("Введите гарантийный срок: ");

                                        if(int.TryParse(Console.ReadLine(), out int printerGuarantee))

                                        {

                                            Console.Write("Введите объем краски: ");

                                            if (int.TryParse(Console.ReadLine(), out int volume))

                                            {

                                            Console.Write("Введите скорость печати: ");

                                             if (int.TryParse(Console.ReadLine(), out int speed))

                                            {

                                              Console.Write("Введите технологию печати: ");

                                               string tech = Console.ReadLine();

                                              appliances.Add(new Printer(printerModel, printerPrice, printerGuarantee, volume, speed, tech));

                                             Console.WriteLine("Объект добавлен!\n");

                                            }

                                            else{

                                                Console.WriteLine("Неверный формат ввода\n");

                                            }

                                        }else{

                                                Console.WriteLine("Неверный формат ввода\n");

                                        }

                                        }else{

                                                Console.WriteLine("Неверный формат ввода\n");

                                        }

                                    }else{

                                            Console.WriteLine("Неверный формат ввода\n");

                                    }

                                    break;

                                case 3:

                                    Console.Write("Введите модель факса: ");

                                    string faxModel = Console.ReadLine();

                                     Console.Write("Введите цену: ");

                                     if (int.TryParse(Console.ReadLine(), out int faxPrice))

                                    {

                                        Console.Write("Введите гарантийный срок: ");

                                        if(int.TryParse(Console.ReadLine(), out int faxGuarantee)){

                                         Console.Write("Введите объем памяти: ");

                                            if (int.TryParse(Console.ReadLine(), out int memory))

                                            {

                                                Console.Write("Введите скорость передачи: ");

                                                if (int.TryParse(Console.ReadLine(), out int faxSpeed))

                                                {

                                                 Console.Write("Введите номер телефона: ");

                                                 string number = Console.ReadLine();

                                                 appliances.Add(new Fax(faxModel, faxPrice, faxGuarantee, memory, faxSpeed, number));

                                                Console.WriteLine("Объект добавлен!\n");

                                                }else{

                                                Console.WriteLine("Неверный формат ввода\n");

                                            }

                                         }else{

                                                Console.WriteLine("Неверный формат ввода\n");

                                        }

                                        }else{

                                                Console.WriteLine("Неверный формат ввода\n");

                                        }

                                    }else{

                                            Console.WriteLine("Неверный формат ввода\n");

                                    }

                                    break;

                                default:

                                    Console.WriteLine("Неверный выбор типа объекта.\n");

                                    break;

                            }

                        }else

                        {

                           Console.WriteLine("Неверный формат ввода\n");

                        }

                        break;

                    case 2:

                        if (appliances.Count == 0)

                        {

                            Console.WriteLine("Список пуст.\n");

                        }

                        else

                        {

                            Console.WriteLine("Содержимое списка:\n");

                            foreach (var appliance in appliances)

                            {

                                appliance.Print();

                            }

                        }

                        break;

                    case 3:

                        Console.WriteLine("Программа завершена.");

                        return;

                    default:

                        Console.WriteLine("Неверный выбор действия.\n");

                        break;

                }

                }

            else

            {

               Console.WriteLine("Неверный формат ввода\n");

            }

        }

    }

}

**UnitTest1.cs**

using System;

using System.IO;

using System.Text;

using Microsoft.VisualStudio.TestTools.UnitTesting;

using System.Globalization;

namespace ApplianceTests

{

    [TestClass]

    public class ApplianceTests

    {

        [TestMethod]

        public void Appliance\_CreationAndGetters\_CorrectValues()

        {

            Appliance appliance = new Appliance("TestModel", 100, 2);

            Assert.AreEqual("TestModel", appliance.GetModel());

            Assert.AreEqual(100, appliance.GetPrice());

            Assert.AreEqual(2, appliance.GetGuarantee());

        }

         [TestMethod]

        public void Appliance\_Setters\_CorrectValues()

        {

            Appliance appliance = new Appliance("Initial", 0, 0);

            appliance.SetModel("NewModel");

            appliance.SetPrice(150);

            appliance.SetGuarantee(5);

            Assert.AreEqual("NewModel", appliance.GetModel());

            Assert.AreEqual(150, appliance.GetPrice());

            Assert.AreEqual(5, appliance.GetGuarantee());

            appliance.SetPrice(999999);

            Assert.AreEqual(150, appliance.GetPrice());

             appliance.SetPrice(-1);

            Assert.AreEqual(150, appliance.GetPrice());

             appliance.SetGuarantee(11);

             Assert.AreEqual(5, appliance.GetGuarantee());

            appliance.SetGuarantee(-1);

            Assert.AreEqual(5, appliance.GetGuarantee());

        }

         [TestMethod]

         public void Appliance\_Print\_ValidOutput()

        {

           CultureInfo.CurrentCulture = new CultureInfo("ru-RU");

          Appliance appliance = new Appliance("TestModel", 100, 2);

          var stringWriter = new StringWriter();

          Console.SetOut(stringWriter);

          appliance.Print();

          var output = stringWriter.ToString();

          var expectedOutput = new StringBuilder();

            expectedOutput.AppendLine("Модель: TestModel");

            expectedOutput.AppendLine("Цена: 100");

            expectedOutput.AppendLine("Гарантийный срок: 2");

            Assert.AreEqual(expectedOutput.ToString(), output, ignoreLineEndingDifferences: true);

          // Reset console output

            Console.SetOut(Console.Out);

        }

        [TestMethod]

        public void Printer\_c\_CreationAndGetters\_CorrectValues()

        {

            Printer\_c printer = new Printer\_c("PrinterModel", 200, 1, 50, 20, "Laser");

            Assert.AreEqual("PrinterModel", printer.GetModel());

            Assert.AreEqual(200, printer.GetPrice());

            Assert.AreEqual(1, printer.GetGuarantee());

            Assert.AreEqual(50, printer.GetVolume());

            Assert.AreEqual(20, printer.GetSpeed());

            Assert.AreEqual("Laser", printer.GetTech());

        }

          [TestMethod]

        public void Printer\_c\_Setters\_CorrectValues()

        {

            Printer\_c printer = new Printer\_c("Initial", 0, 0, 0, 0, "Initial");

            printer.SetVolume(60);

            printer.SetSpeed(25);

           Assert.AreEqual(60, printer.GetVolume());

            Assert.AreEqual(25, printer.GetSpeed());

              printer.SetVolume(101);

             Assert.AreEqual(60, printer.GetVolume());

            printer.SetVolume(-1);

             Assert.AreEqual(60, printer.GetVolume());

            printer.SetSpeed(101);

              Assert.AreEqual(25, printer.GetSpeed());

            printer.SetSpeed(-1);

             Assert.AreEqual(25, printer.GetSpeed());

        }

         [TestMethod]

        public void Printer\_c\_Print\_ValidOutput()

        {

            CultureInfo.CurrentCulture = new CultureInfo("ru-RU");

             Printer\_c printer = new Printer\_c("PrinterModel", 200, 1, 50, 20, "Laser");

             var stringWriter = new StringWriter();

            Console.SetOut(stringWriter);

            printer.Print();

            var output = stringWriter.ToString();

          var expectedOutput = new StringBuilder();

            expectedOutput.AppendLine("Модель: PrinterModel");

            expectedOutput.AppendLine("Цена: 200");

            expectedOutput.AppendLine("Гарантийный срок: 1");

            expectedOutput.AppendLine("Объем краски: 50");

            expectedOutput.AppendLine("Скорость печати: 20");

            expectedOutput.AppendLine("Технология печати: Laser");

             expectedOutput.AppendLine("");

            Assert.AreEqual(expectedOutput.ToString(), output, ignoreLineEndingDifferences: true);

            // Reset console output

             Console.SetOut(Console.Out);

        }

        [TestMethod]

        public void Fax\_CreationAndGetters\_CorrectValues()

        {

            Fax fax = new Fax("FaxModel", 150, 3, 30, 10, "123-456-7890");

             Assert.AreEqual("FaxModel", fax.GetModel());

            Assert.AreEqual(150, fax.GetPrice());

            Assert.AreEqual(3, fax.GetGuarantee());

            Assert.AreEqual(30, fax.GetMemory());

            Assert.AreEqual(10, fax.GetSpeed());

            Assert.AreEqual("123-456-7890", fax.GetNumber());

        }

        [TestMethod]

         public void Fax\_Setters\_CorrectValues()

        {

            Fax fax = new Fax("Initial", 0, 0, 0, 0, "Initial");

            fax.SetMemory(60);

            fax.SetSpeed(25);

            Assert.AreEqual(60, fax.GetMemory());

            Assert.AreEqual(25, fax.GetSpeed());

              fax.SetMemory(101);

              Assert.AreEqual(60, fax.GetMemory());

            fax.SetMemory(-1);

            Assert.AreEqual(60, fax.GetMemory());

            fax.SetSpeed(101);

            Assert.AreEqual(25, fax.GetSpeed());

            fax.SetSpeed(-1);

            Assert.AreEqual(25, fax.GetSpeed());

        }

        [TestMethod]

        public void Fax\_Print\_ValidOutput()

        {

              CultureInfo.CurrentCulture = new CultureInfo("ru-RU");

            Fax fax = new Fax("FaxModel", 150, 3, 30, 10, "123-456-7890");

              var stringWriter = new StringWriter();

            Console.SetOut(stringWriter);

            fax.Print();

            var output = stringWriter.ToString();

           var expectedOutput = new StringBuilder();

           expectedOutput.AppendLine("Модель: FaxModel");

           expectedOutput.AppendLine("Цена: 150");

           expectedOutput.AppendLine("Гарантийный срок: 3");

           expectedOutput.AppendLine("Объем памяти: 30");

           expectedOutput.AppendLine("Скорость передачи: 10");

           expectedOutput.AppendLine("Номер: 123-456-7890");

             expectedOutput.AppendLine("");

          Assert.AreEqual(expectedOutput.ToString(), output, ignoreLineEndingDifferences: true);

              // Reset console output

            Console.SetOut(Console.Out);

        }

        [TestMethod]

        public void Main\_Menu\_InvalidInput()

        {

            var input = new StringReader("a\n3");

              var output = new StringWriter();

            Console.SetIn(input);

            Console.SetOut(output);

            Program.Main(new string[0]);

             var consoleOutput = output.ToString();

             Assert.IsTrue(consoleOutput.Contains("Некорректный ввод, попробуйте снова."));

         }

        [TestMethod]

          public void Main\_Menu\_AddAppliance()

        {

              var input = new StringReader("1\n1\nTestModel\n100\n2\n3");

             var output = new StringWriter();

            Console.SetIn(input);

            Console.SetOut(output);

            Program.Main(new string[0]);

            var consoleOutput = output.ToString();

             Assert.IsTrue(consoleOutput.Contains("Модель: TestModel"));

             Assert.IsTrue(consoleOutput.Contains("Цена: 100"));

             Assert.IsTrue(consoleOutput.Contains("Гарантийный срок: 2"));

         }

        [TestMethod]

          public void Main\_Menu\_AddPrinter()

        {

              var input = new StringReader("1\n2\nPrinterModel\n200\n1\n50\n20\nLaser\n3");

            var output = new StringWriter();

            Console.SetIn(input);

            Console.SetOut(output);

           Program.Main(new string[0]);

           var consoleOutput = output.ToString();

             Assert.IsTrue(consoleOutput.Contains("Модель: PrinterModel"));

           Assert.IsTrue(consoleOutput.Contains("Цена: 200"));

            Assert.IsTrue(consoleOutput.Contains("Гарантийный срок: 1"));

            Assert.IsTrue(consoleOutput.Contains("Объем краски: 50"));

          Assert.IsTrue(consoleOutput.Contains("Скорость печати: 20"));

            Assert.IsTrue(consoleOutput.Contains("Технология печати: Laser"));

        }

           [TestMethod]

        public void Main\_Menu\_AddFax()

        {

               var input = new StringReader("1\n3\nFaxModel\n150\n3\n30\n10\n123-456-7890\n3");

           var output = new StringWriter();

            Console.SetIn(input);

            Console.SetOut(output);

          Program.Main(new string[0]);

        var consoleOutput = output.ToString();

            Assert.IsTrue(consoleOutput.Contains("Модель: FaxModel"));

              Assert.IsTrue(consoleOutput.Contains("Цена: 150"));

            Assert.IsTrue(consoleOutput.Contains("Гарантийный срок: 3"));

             Assert.IsTrue(consoleOutput.Contains("Объем памяти: 30"));

            Assert.IsTrue(consoleOutput.Contains("Скорость передачи: 10"));

            Assert.IsTrue(consoleOutput.Contains("Номер: 123-456-7890"));

        }

    }

}