Учреждение образования "Полоцкий Государственный Университет"

Факультет информационных технологий
Кафедра вычислительных систем и сетей

Отчет по Лабораторной работе № 3

по дисциплине: **"Объектно-Ориентированные Технологии Программирования и Стандарты Проектирования"**

ВЫПОЛНИЛ

студент группы 18-IT-2
Сыцевич Д.Н.
вариант № 14

ПРОВЕРИЛ

старший преподаватель
Ярошевич П.В.

Полоцк 2020 г.

1 Задача

На основе предыдущей лабораторной работы необходимо реализовать программу, кото- рая позволяет:

• вводить информацию об объекте в одну строку (для каждого класса должен быть разработан отдельный формат ввода), предполагается что ввод всегда корректный; • добавлять элементы в разработанный класс-список; • сортировать список объектов дочерних классов. Для возможности сравнивать объекты между собой необходимо: создать интерфейс

с названием [НазваниеБазовогоКласса]Compareable, который содержал бы метод сравне- ния compare (метод должен возвращать целочисленное значение, обозначающее необхо- димость перестановки объектов местами, и принимать ссылку на объект базового класса);

объявить базовый класс абстактным принадлежащим этому интерфейсу, и реализовать метод/методы интерфейса в каждом из дочерних классов. Класс-список должен быть реализован на основе массива либо ссылок. Класс-список, содержит следующие методы:

3.2. ИНТЕРФЕЙСЫ В МЕХАНИЗМЕ ОБРАТНОГО ВЫЗОВА 13

• добавление элемента в конец списка; • добавление элемента в произвольное положения в списке; • удаление элемента по индексу; • удаление всех элементов из списка; • получение элемента по индексу; • изменение элемента по индексу; • сортировка списка (сперва по классам, а внутри классов по выбраному полю каждого из дочерних).

2 Вариант № 14

14. деталь, механизм, изделие, узел;

3 Ход выполнения

Написать демонстрационную программу, в которой создаются объекты реализованных классов и помещаются в список. Допускается создание объектов на основе введенных данных. Затем осуществляется вывод элементов массива в консоль.

1.

4 Скриншоты

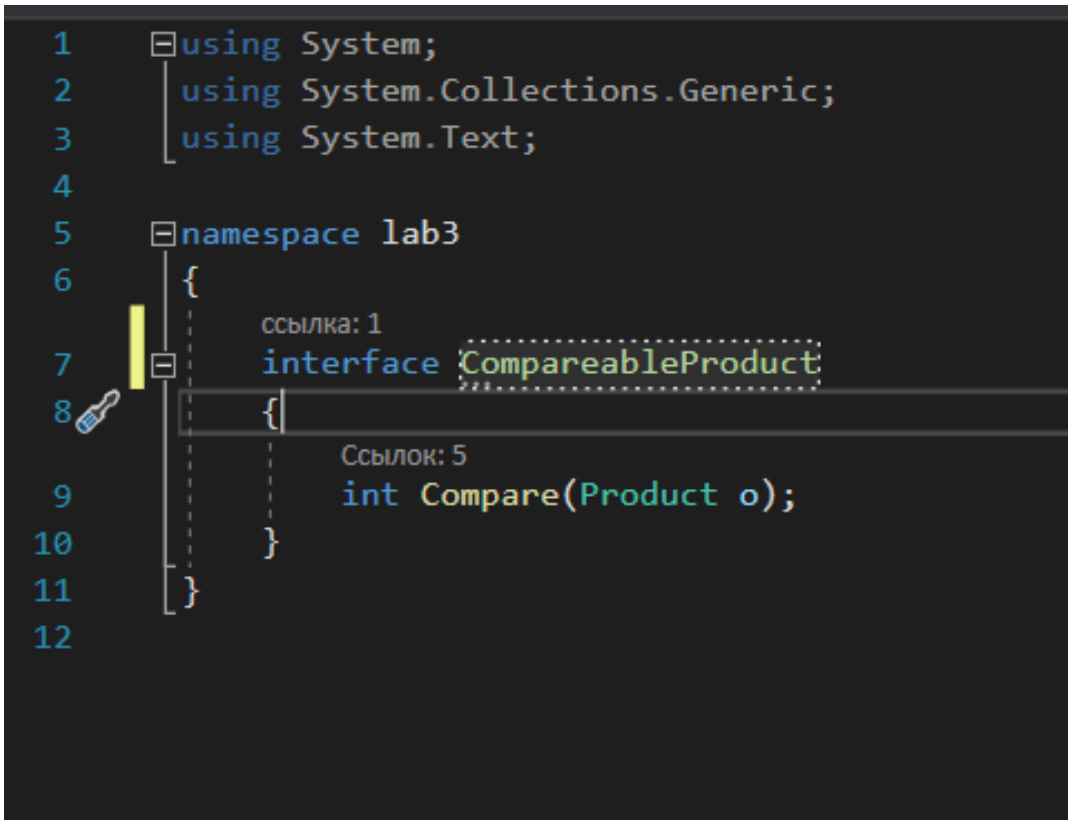
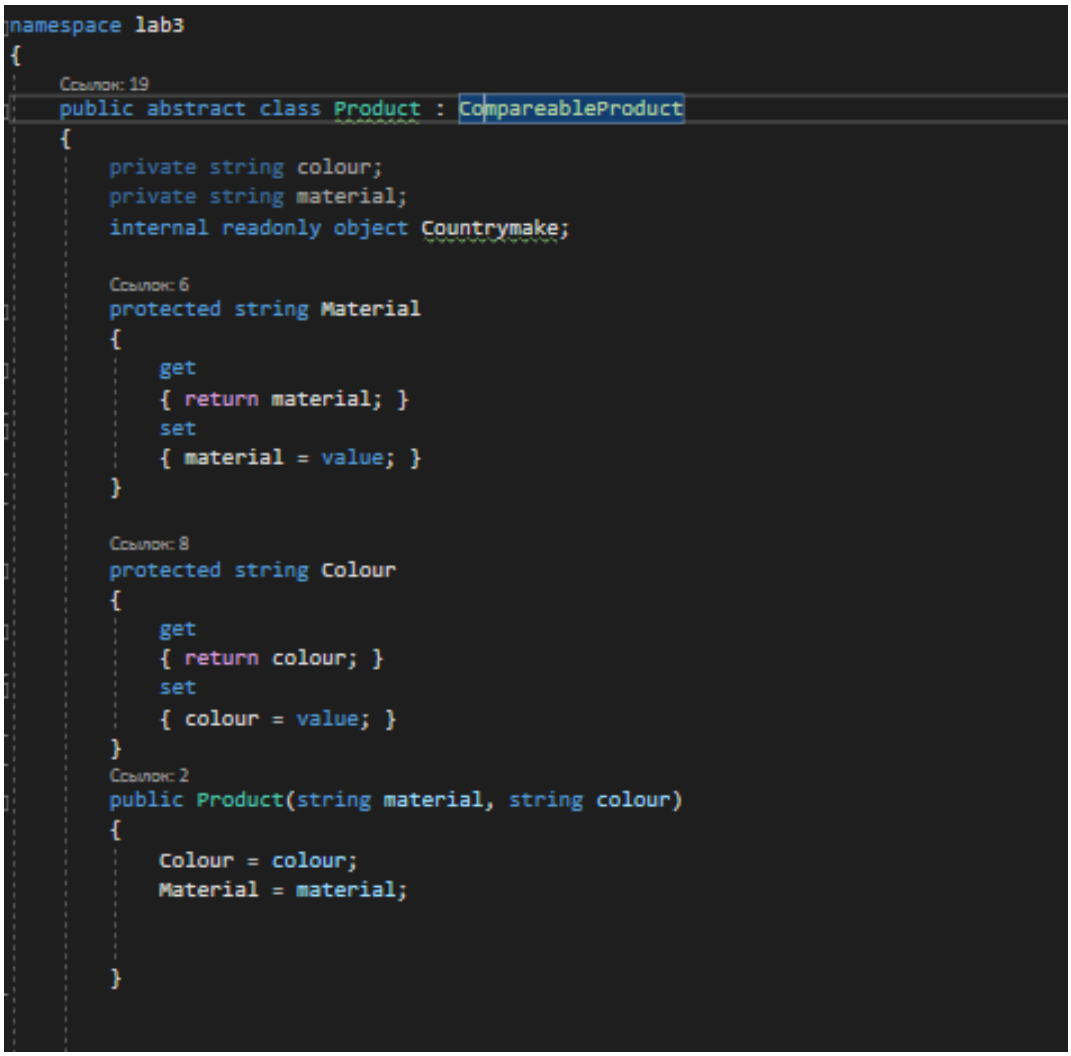


Рис. 1: Интерфейс CompareableProduct



```
namespace lab3
{
    Ссылок: 19
    public abstract class Product : CompareableProduct
    {
        private string colour;
        private string material;
        internal readonly object Countrymake;

        Ссылок: 6
        protected string Material
        {
            get
            { return material; }
            set
            { material = value; }
        }

        Ссылок: 8
        protected string Colour
        {
            get
            { return colour; }
            set
            { colour = value; }
        }
        Ссылок: 2
        public Product(string material, string colour)
        {
            Colour = colour;
            Material = material;


        }
```

Рис. 2: Класс список Product

```
class Program
{


    Ссылок: 0
    static void Main(string[] args)
    {
        Region_list list = new Region_list();
        Mechanizm obj = new Mechanizm("Пластик", "Красный", "Зубчатый", "EcoForm", 101);
        Unit obj2 = new Unit("Древесина", "Желтый", 7, "Средний");
        Detail obj7 = new Detail("Алюминий", "Бирюзовый", 20, "Средний", "В часах", "Албания", 99);
        Unit obj3 = new Unit("Металл", "Оранжевый", 13, "Маленький");
        Mechanizm obj4 = new Mechanizm("Эко-пластик", "Желтый", "Гидравлический", "SichCompany", 98);
        Detail obj5 = new Detail("Металл", "Черный", 5, "Большой", "В технике", "Япония", 1001);

        Detail obj6 = new Detail("Алюминий", "Желтый", 20, "Средний", "В часах", "Хорватия", 1000);



        list.Add(obj);
        list.Add(obj2);
        list.Add(obj3);
        list.Add(obj4);
        list.Add(obj5);
        list.Add(obj6);
        list.Add(obj7);

        Console.WriteLine(list.ToString());
        Console.WriteLine(" ");
        Console.WriteLine("Сортировка:");
        list.Sort();
        Console.WriteLine(list.ToString());
        Console.WriteLine(" ");
        Console.WriteLine("Удаление 4 элемента:");
        list.RemoveAt(3);
        Console.WriteLine(list.ToString());
        Console.WriteLine(" ");
        Console.WriteLine("Отчистить все данные:");
        list.Clear();
        Console.WriteLine(list.ToString());
        Console.WriteLine("ПУсто)");



        Console.ReadLine();
```

Рис. 3: Главный класс программы Program

```
D:\study\OOP\3laba\3laba — копия\3laba\bin\Debug\netcoreapp3.1\3laba.exe
Цвет: Красный Материал: Пластик Тип механизма: Зубчатый Производитель: EcoForm Цена: 101$
Цвет: Желтый Материал: Древесина Размер: Средний Срок службы: 7 лет
Цвет: Оранжевый Материал: Металл Размер: Маленький Срок службы: 13 лет
Цвет: Желтый Материал: Эко-пластик Тип механизма: Гидравлический Производитель: SichCompany Цена: 98$
Цвет: Черный Материал: Металл Размер: Большой Срок службы: 5 лет Где используется: В технике Страна производитель: Япония Количество: 1001 штук
Цвет: Желтый Материал: Алюминий Размер: Средний Срок службы: 20 лет Где используется: В часах Страна производитель: Хорватия Количество: 1000 штук
Цвет: Бирюзовый Материал: Алюминий Размер: Средний Срок службы: 20 лет Где используется: В часах Страна производитель: Албания Количество: 99 штук

Сортировка:
Цвет: Желтый Материал: Эко-пластик Тип механизма: Гидравлический Производитель: SichCompany Цена: 98$
Цвет: Красный Материал: Пластик Тип механизма: Зубчатый Производитель: EcoForm Цена: 101$
Цвет: Желтый Материал: Древесина Размер: Средний Срок службы: 7 лет
Цвет: Оранжевый Материал: Металл Размер: Маленький Срок службы: 13 лет
Цвет: Бирюзовый Материал: Алюминий Размер: Средний Срок службы: 20 лет Где используется: В часах Страна производитель: Албания Количество: 99 штук
Цвет: Желтый Материал: Алюминий Размер: Средний Срок службы: 20 лет Где используется: В часах Страна производитель: Хорватия Количество: 1000 штук
Цвет: Черный Материал: Металл Размер: Большой Срок службы: 5 лет Где используется: В технике Страна производитель: Япония Количество: 1001 штук

Удаление 4 элемента:
Цвет: Желтый Материал: Эко-пластик Тип механизма: Гидравлический Производитель: SichCompany Цена: 98$
Цвет: Красный Материал: Пластик Тип механизма: Зубчатый Производитель: EcoForm Цена: 101$
Цвет: Желтый Материал: Древесина Размер: Средний Срок службы: 7 лет
Цвет: Бирюзовый Материал: Алюминий Размер: Средний Срок службы: 20 лет Где используется: В часах Страна производитель: Албания Количество: 99 штук
Цвет: Желтый Материал: Алюминий Размер: Средний Срок службы: 20 лет Где используется: В часах Страна производитель: Хорватия Количество: 1000 штук
Цвет: Черный Материал: Металл Размер: Большой Срок службы: 5 лет Где используется: В технике Страна производитель: Япония Количество: 1001 штук

Отчистить все данные:

ПУсто)
```

Рис. 4: Результат выполнения программы

5 Source Code

CompareableProduct.cs

```
  using   System;
using System.Collections.Generic;
using System.Text;

namespace lab3
{
    interface CompareableProduct
    {
        int Compare(Product o);
    }
}
```

Product.cs

```
  using   System;
using System.Collections.Generic;
using System.Text;


namespace lab3
{
    public abstract class Product : CompareableProduct
    {
        private string colour;
        private string material;
        internal readonly object Countrymake;

        protected string Material
        {
            get
            { return material; }
            set
            { material = value; }
        }

        protected string Colour
        {
            get
```

```
    { return colour; }
    set
    { colour = value; }
}
public Product(string material, string colour)
{
    Colour = colour;
    Material = material;


}



public Product(Product product)
{
    Colour = product.Colour;
    Material = product.Material;


}
public virtual int Compare(Product o)
{
    return this.Colour.CompareTo(o.Colour);
}


public override bool Equals(Object obj)
{
    if (!(obj is Product)) return false;
    var product = (Product)obj;

    if (Material == product.Material &&

    Colour == product.Colour)
    return true;
    else
    return false;
}
public override string ToString()
```

```
        {
            return $"         : {Colour} " +  $"                    : {M
        }


    }
}
```

class-list.cs

```
  using   System;
using System.Collections;
using System.Collections.Generic;
using System.Text;

namespace lab3
{
    class Region_list
    {
        List<Product> list = new List<Product>();
        public void Add(Product a)
        {
            list.Add(a);
        }

        public void Insert(int Pos, Product a)
        {
            list.Insert(Pos, a);
        }

        public void RemoveAt(int Index)
        {
            list.RemoveAt(Index);
        }

        public void Clear()
        {
            list.Clear();
        }

        public void Sort()
```

```
        {
            for (int i = 0; i < list.Count; i++)
            {
                for (int j = i + 1; j < list.Count; j++)
                {
                    if (this.list[i].Compare(this.list[j]) > 0)
                    {
                        Product o = this.list[i];
                        this.list[i] = this.list[j];
                        this.list[j] = o;
                    }
                }
            }
        }

        public override string ToString()
        {
            string retVal = string.Empty;
            foreach (Product obj in list)
            {
                if (string.IsNullOrEmpty(retVal))
                    retVal += obj.ToString();
                else
                    retVal += string.Format("\n{0}", obj.ToString());
            }
            return retVal;
        }
    }
}
```

Program.cs

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

namespace lab3
{


    class Program
```

```
{


    static void Main(string[] args)
    {
        Region_list list = new Region_list();
        Mechanizm obj = new Mechanizm("                ", "
        Unit obj2 = new Unit("                    ", "
        Detail obj7 = new Detail("                  ", "
        Unit obj3 = new Unit("              ", "
        Mechanizm obj4 = new Mechanizm("       –                ",
        Detail obj5 = new Detail("             ", "              ",

        Detail obj6 = new Detail("                  ", "



        list.Add(obj);
        list.Add(obj2);
        list.Add(obj3);
        list.Add(obj4);
        list.Add(obj5);
        list.Add(obj6);
        list.Add(obj7);

        Console.WriteLine(list.ToString());
        Console.WriteLine(" ");
        Console.WriteLine("                      :");
        list.Sort();
        Console.WriteLine(list.ToString());
        Console.WriteLine(" ");
        Console.WriteLine("                  4                  :"
        list.RemoveAt(3);
        Console.WriteLine(list.ToString());
        Console.WriteLine(" ");
        Console.WriteLine("
        list.Clear();
        Console.WriteLine(list.ToString());
        Console.WriteLine("          )");
```

```
            Console.ReadLine();

        }
    }
}
```

Detail.cs

```
  using   System;
using System.Collections.Generic;
using System.Text;

namespace lab3
{
    public class Detail : Unit
    {

        private string use;
        private string countrymake;
        private int kol;

        public string Use
        {
            get
            { return use; }
            set
            { use = value; }
        }

        public string Countrymake
        {
            get
            { return countrymake; }
            set
            { countrymake = value; }
        }
        public int Kol
        {
            get
```

```
    { return kol; }
    set
    { kol = value; }
}



public Detail(string colour, string material, int year, stri
{
    Use = use;
    Countrymake = countrymake;
    Kol = kol;

}

public Detail(Detail detail) : base(detail)
{
    Use = detail.use;
    Countrymake = detail.countrymake;
    Kol = detail.kol;

}


public override int Compare(Product o)
{
    return this.Countrymake.CompareTo(o.Countrymake);
}

public override bool Equals(Object obj)
{
    if (!(obj is Detail)) return false;
    var detail = (Detail)obj;

    if (Use == detail.use &&
        Countrymake == detail.countrymake &&
        Kol == detail.kol &&
        base.Equals(obj))
        return true;
    else
```

```
                return false;
        }

        public override string ToString()
        {
            return base.ToString() +
                $"                                    : {Use} " + $"
        }
    }
}
```

Input.cs

```
  using   System;
using System.Collections.Generic;
using System.Text;

namespace lab3
{
    class Input
    {
        static Product InputMechanizm()
        {
            Console.WriteLine("                              ,
            string input1 = Console.ReadLine();
            string[] separators = { "," };
            string[] words = input1.Split(separators, StringSplitOpt
            Mechanizm first = new Mechanizm(words[0], words[1],
words[2], words[3], Convert.ToInt32(words[4]));
            return first;
        }
        static Unit InputUnit()
        {
            Console.WriteLine("                              ,
            string input1 = Console.ReadLine();
            string[] separators = { "," };
            string[] words = input1.Split(separators, StringSplitOpt
            Unit first = new Unit(words[0], words[1], Convert.ToInt3
            return first;
        }
        static Detail InputDetail()
```

```
        {
            Console.WriteLine("                                    ,
            string input1 = Console.ReadLine();
            string[] separators = { "," };
            string[] words = input1.Split(separators, StringSplitOpt
            Detail first = new Detail(words[0], words[1], Convert.To
            return first;
        }
    }
}
```

Mechanizm.cs

```
  using   System;
using System.Collections.Generic;
using System.Text;

namespace lab3
{
    public class Mechanizm : Product
    {
        private string type;
        private string proizv;
        private float price;

        public string Type
        {
            get
            { return type; }
            set
            { type = value; }
        }

        public string Proizv
        {
            get
            { return proizv; }
            set
            { proizv = value; }
        }
        protected float Price
```

```
{
    get
    { return price; }
    set
    { price = value; }
}

public Mechanizm(string colour, string material, string type
{
    Type = type;
    Proizv = proizv;
    Price = price;

}

public Mechanizm(Mechanizm mechanizm) : base(mechanizm)
{
    Type = mechanizm.type;
    Proizv = mechanizm.proizv;
    Price = mechanizm.Price;
}
public override int Compare(Product o)
{
  if (o is Unit) return -1;
  if (o is Detail) return 1;
    return (-1) * (o as Mechanizm).Price.CompareTo(Price);
}

public override bool Equals(Object obj)
{
    if (!(obj is Mechanizm)) return false;
    var mechanizm = (Mechanizm)obj;

    if (Type == mechanizm.type &&
    Proizv == mechanizm.proizv &&
    Price == mechanizm.Price &&
        base.Equals(obj))
        return true;
    else
        return false;
```

```
        }

        public override string ToString()
        {

            return base.ToString() +
                $"                              : {Type} " + $"
        }


    }
}
```

Unit.cs

```
  using   System;
using System.Collections.Generic;
using System.Text;

namespace lab3
{
    public class Unit : Product
    {
        private int year;
        private string size;
        public int Year
        {
            get
            { return year; }
            set
            { year = value; }
        }
        public string Size
        {
            get
            { return size; }
            set
            { size = value; }
        }
```

```
    public Unit(string colour, string material, int year, string
    {
        Year = year;
        Size = size;
    }

    public Unit(Unit unit) : base(unit)
    {
        Year = unit.year;
        Size = unit.size;
    }

     public override int Compare(Product o)
    {
        if (o is Detail) return -1;
        if (o is Mechanizm) return 1;
        return (-1) * (o as Unit).Year.CompareTo(Year);
    }
    public override bool Equals(Object obj)
    {
        if (!(obj is Unit)) return false;
        var unit = (Unit)obj;

        if (Year == unit.year &&
            Size == unit.size &&
            base.Equals(obj))
            return true;
        else
            return false;
    }

    public override string ToString()
    {
        return base.ToString() +
            $"            : {Size} " + $"                    :
    }


}
```

```
}
```