**9 Интерфейсы**

Задание 1. Интерфейсы Ix, Iy, Iz, содержат объявления методов с одной и той же сигнатурой следующим образом Эти интерфейсы наследуются в классе TestClass, содержащий член w типа параметр и реализуются так, как задано в варианте. В каждом методе задать вывод результата.

Рассмотреть случай

 неявной реализации интерфейсов

 явной реализации интерфейса Iz

В программе должна выполняться:

 неявная неоднозначная реализация методов интерфейсов Iy и Iz,

 вызов функций с явным приведением к типу интерфейса,

 вызов метода для объекта посредством интерфейсной ссылки.

static void Main(string[] args)

{

TestClass ts = new TestClass();

Console.WriteLine($"IxF0: {ts.IxF0(10):f2}, IxF1: {ts.IxF1():f2}");

Console.WriteLine($"F0: {ts.F0(10)}, F1: {ts.F1()}");

Console.WriteLine($"F0: {((Iz)ts).F0(10)}, F1: {((Iz)ts).F1()}");

}

interface Ix

{

double IxF0(double key);

double IxF1();

}

interface Iy

{

double F0(double key);

double F1();

}

interface Iz

{

double F0(double key);

double F1();

}

class TestClass:Ix,Iy,Iz

{

public double val;

public double IxF0(double key)

{

val = key;

return Math.Log(val);

}

public double IxF1()

{

return Math.Log(val);

}

public double F0(double key)

{

val = key;

return 2 / val;

}

public double F1()

{

return 2 /val;

}

double Iz.F0(double key)

{

val = key;

return Math.Pow(val, 3); ;

}

double Iz.F1()

{

return Math.Pow(val, 3);

}

}

Таблица 1.1 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
|  | IxF0: 2,30, IxF1: 2,30  F0: 0,2, F1: 0,2  F0: 1000, F1: 1000 |

Анализ результатов:

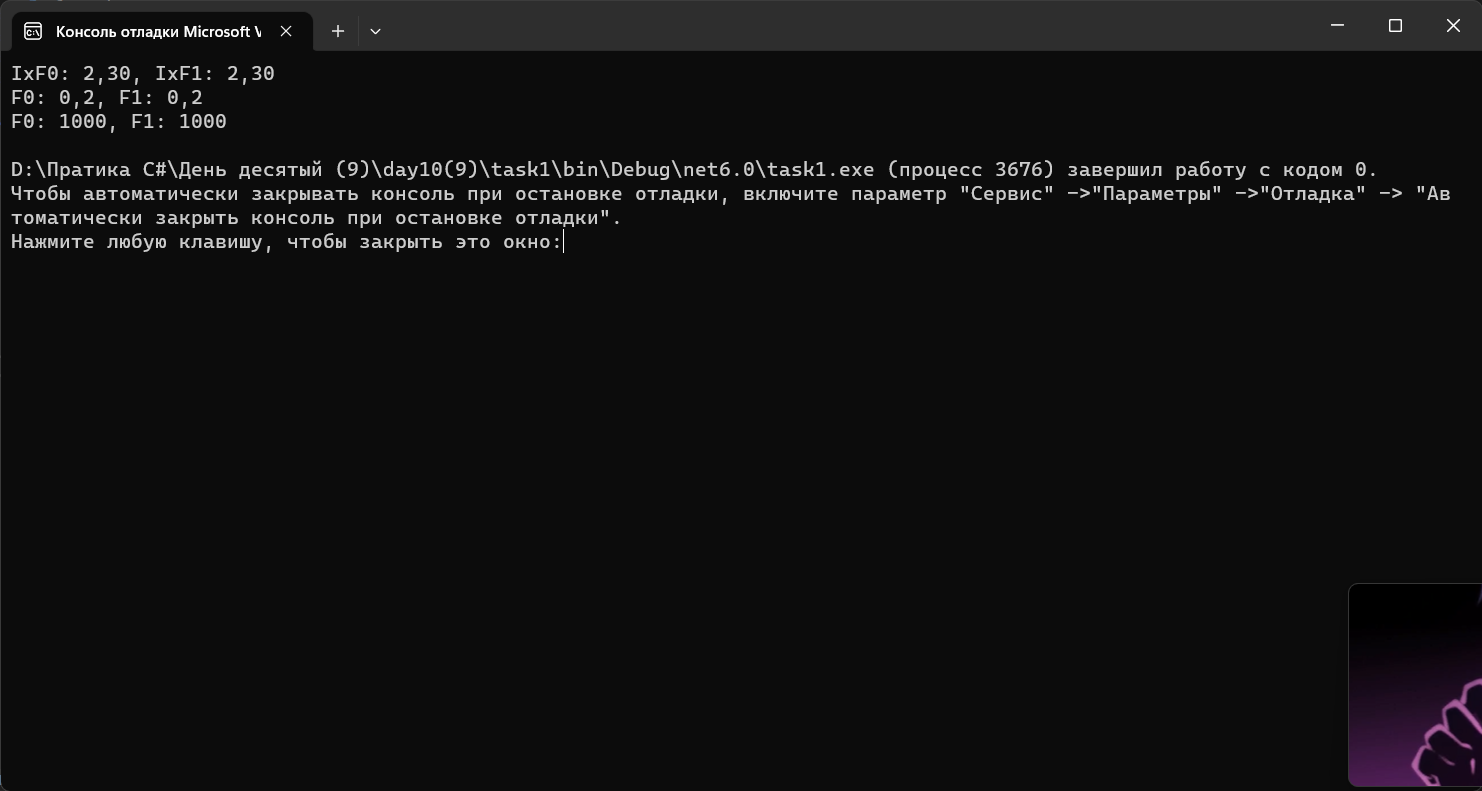


Рисунок 1.1 – Результат работы программы

Задание 2. Выполнить задания, используя для хранения экземпляров разработанных классов массивы. Во всех классах реализовать интерфейс IComparable и перегрузить операции отношения для реализации значимой семантики сравнения объектов по какому-либо полю на усмотрение студента.

Описать класс «товар», содержащий следующие закрытые поля:

 название товара;

 название магазина, в котором продается товар;

 стоимость товара в рублях.

Предусмотреть свойства для получения состояния объекта. Описать класс «склад», содержащий закрытый массив товаров. Обеспечить следующие возможности:

 вывод информации о товаре по номеру с помощью индекса;

 сортировку товаров по названию магазина, по наименованию и по

цене;

Написать программу, демонстрирующую все разработанные элементы

классов.

namespace task2

{

internal class Program

{

static void Main(string[] args)

{

Warehouse warehouse = new Warehouse();

Product product1 = new Product("Книга", "Книжный магазин", 500);

Product product2 = new Product("Мыло", "Супермаркет", 50);

Product product3 = new Product("Футболка", "Магазин Одежды", 1000);

Product product4 = new Product("Кросовки", "Обувной магазин", 200);

warehouse.Add(product1);

warehouse.Add(product2);

warehouse.Add(product3);

warehouse.Add(product4);

Console.WriteLine("Выберите сортировку: \n 1 - по названию Товара \n 2 - по названию магазина \n 3 - по цене ");

int number = int.Parse(Console.ReadLine());

switch (number)

{

case 1:

warehouse.SortByName(); // сортировка товаров по названию Товара

break;

case 2:

warehouse.SortByStore(); // сортировка товаров по названию магазина

break;

case 3:

warehouse.SortByPrice(); // сортировка товаров по цене

break;

}

foreach (Product product in warehouse.GetAll())

{

Console.WriteLine(product);

}

Console.WriteLine("-----------------------------------------------");

Console.WriteLine("Введите индекс элемента который хотите вывести: ");

int index = int.Parse(Console.ReadLine());

warehouse.Display(index - 1); // вывод информации о товаре по номеру с помощью индекса

}

}

}

internal class Product: IComparable<Product>

{

private string \_name;

private string \_store;

private int \_price;

public string Name { get { return \_name; } set { \_name = value; } }

public string Store { get { return \_store; } set { \_store = value; } }

public int Price { get { return \_price; } set { \_price = value; } }

public Product(string name, string store, int price )

{

\_name = name;

\_store = store;

\_price = price;

}

public override string ToString()

{

return String.Format($"Товар: {Name} ({Store}, {Price:C})");

}

public int CompareTo(Product other)

{

return Name.CompareTo(other.Name);

}

public static bool operator <(Product product1, Product product2)

{

return product1.Price < product2.Price;

}

public static bool operator >(Product product1, Product product2)

{

return product1.Price > product2.Price;

}

}

class Warehouse

{

private List<Product> products;

public Warehouse()

{

products = new List<Product>();

}

public void Add(Product product)

{

products.Add(product);

}

public void Remove(Product product)

{

products.Remove(product);

}

public void Display(int index)

{

Console.WriteLine(products[index]);

}

public void SortByName()

{

products.Sort((p1, p2) => p1.Name.CompareTo(p2.Name));

}

public void SortByStore()

{

products.Sort((p1, p2) => p1.Store.CompareTo(p2.Store));

}

public void SortByPrice()

{

products.Sort((p1, p2) => p1.Price.CompareTo(p2.Price));

}

public Product[] GetAll()

{

return products.ToArray();

}

}

Таблица 1.2 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
|  | Книга (Книжный магазин, 500,00 Br)  Кросовки (Обувной магазин, 200,00 Br)  Мыло (Супермаркет, 50,00 Br)  Футболка (Магазин Одежды, 1 000,00 Br) |

Анализ результатов:

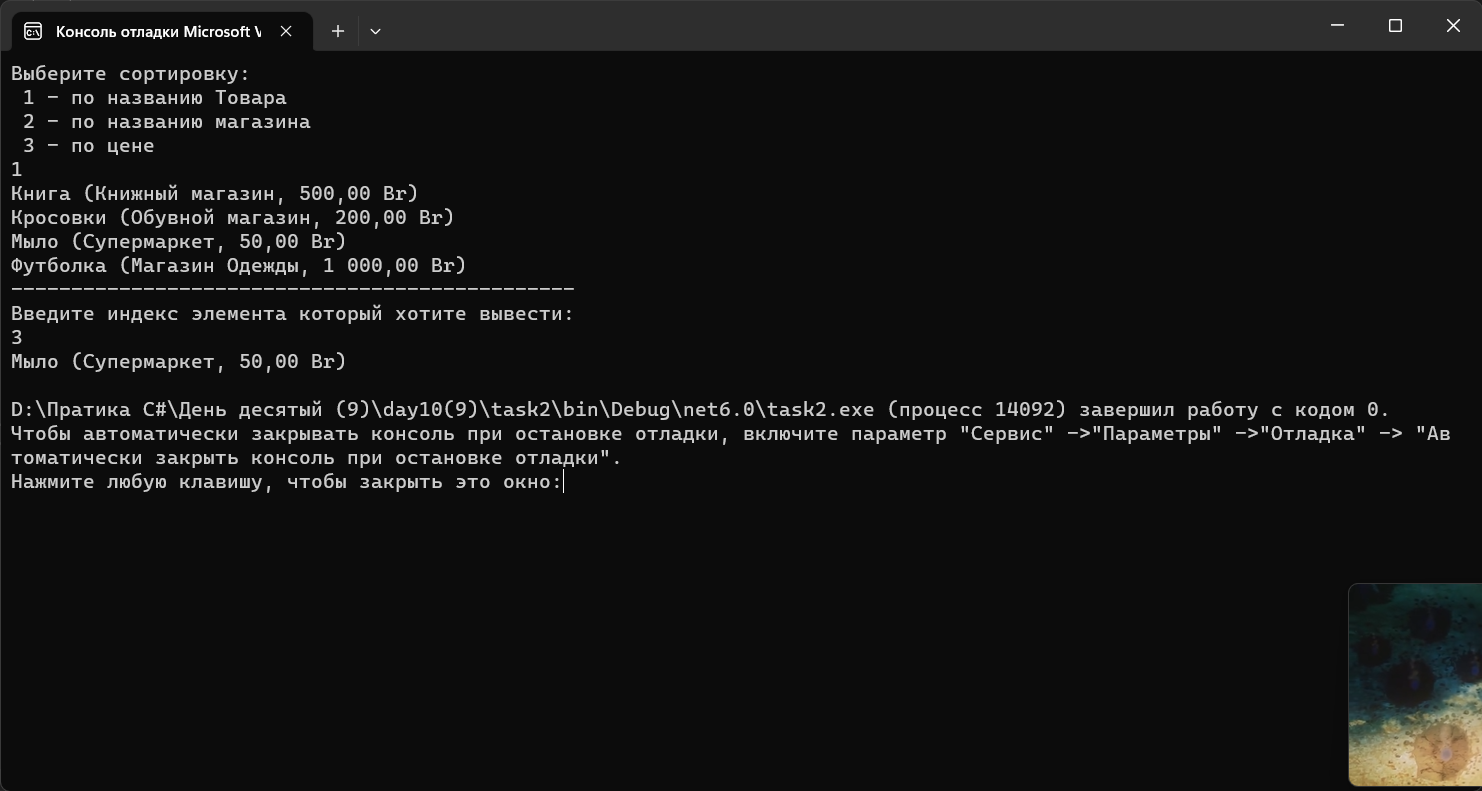
**

Рисунок 1.2 – Результат работы программы

работы программы