МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Національний аерокосмічний університет ім. М.Є. Жуковського

«Харківський авіаційний інститут»

Факультет радіоелектроніки, комп’ютерних систем та інфокомунікацій

Кафедра комп’ютерних систем, мереж і кібербезпеки (503)

Лабораторна робота № *4*

|  |  |
| --- | --- |
|  | *Наслідування та поліморфізм. Інтерфейси.* |
|  |  |
| з дисципліни | *Технології програмування* |
|  | ХАІ**.**503**.**525a**.**03О**.**123-Комп'ютерна інженерія**,** ПЗ №9629619 |

|  |  |  |
| --- | --- | --- |
| Виконав студент гр. | 525а | *Литвиненко А.В.* |
| 11.11.2022 | (№ групи) | (П.І.Б.) |
| (підпис, дата) |  |  |
| Перевірив | канд. техн. наук, доцент | |
|  |  | *Є. В. Бабешенко* |
| (підпис, дата) |  | (П.І.Б.) |

Харків – 2022

**Тема роботи:** Наслідування та поліморфізм. Інтерфейси.

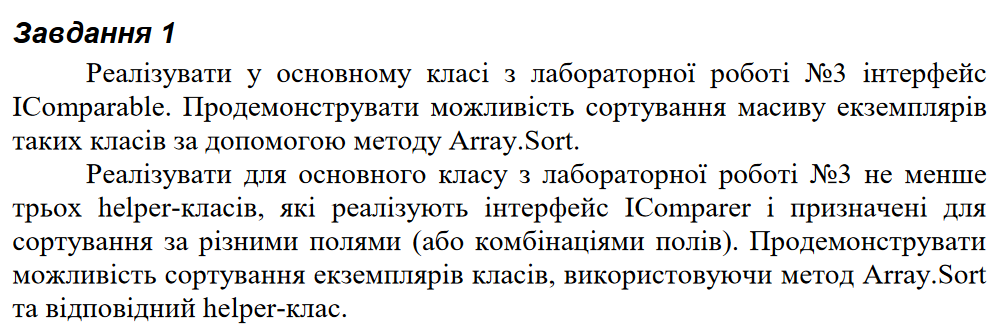
**Мета роботи**: Вивчити парадигми «наслідування» та «поліморфізм». Освоїти можливості мови C#, призначені для реалізації даних парадигм. Вивчити призначення інтерфейсів. Використати на практиці upcast та downcast.

**Варіант 5**

**Задача 1**

**Частина 1**. Постановка завдання

**Умова:**



**Умова з додатка:**



Частина 2. Схема класу

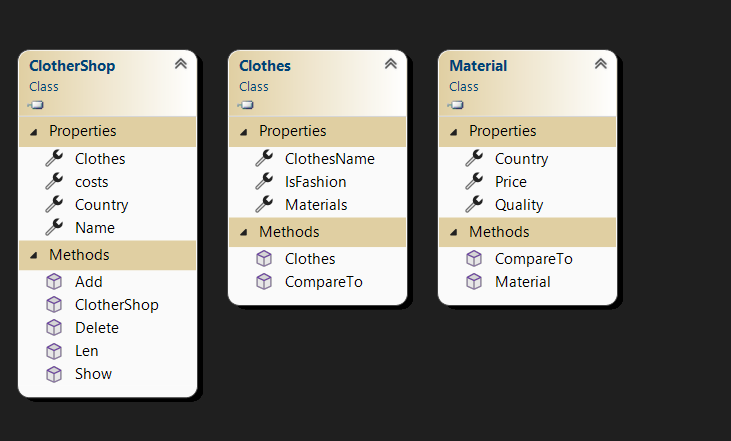


Рисунок 1 - Схема

**Частина 3**. Текст програми

Відповідно до розробленого алгоритму в середовищі Microsoft Visual Studio була написана програма, яка наведена нижче.

Program.cs

using Task02;

namespace Task02\_App

{

internal class Program

{

static void Main(string[] args)

{

Material Cotton = new Material(50, 100, "Ukraine");

Material Silk = new Material(90, 50, "Poland");

Material Leather = new Material(100, 100, "USA");

Material[] forCoat = new Material[3];

forCoat[0] = Cotton;

forCoat[1] = Leather;

forCoat[2] = Silk;

Material[] forJeens = new Material[2];

forJeens[0] = Silk;

forJeens[1] = Cotton;

Clothes Coat = new Clothes(forCoat, "Premium Coat", true);

Clothes Jeens = new Clothes(forJeens, "Cool Jeens", false);

Clothes Pants = new Clothes(forJeens, "Pants from hell", true);

Clothes[] goods = new Clothes[3];

goods[0] = Coat;

goods[1] = Jeens;

goods[2] = Pants;

Console.WriteLine("Unsorted array: ");

foreach (Material item in Jeens.Materials)

{

Console.WriteLine(item.Price);

}

Console.WriteLine();

Array.Sort(Jeens.Materials);

Console.WriteLine("Sorted array: ");

foreach (Material item in Jeens.Materials)

{

Console.WriteLine(item.Price);

}

Console.WriteLine();

Console.WriteLine("Unsorted array: ");

foreach (Clothes item in goods)

{

Console.WriteLine(item.ClothesName);

}

Console.WriteLine();

Array.Sort(goods);

Console.WriteLine("Sorted array: ");

foreach (Clothes item in goods)

{

Console.WriteLine(item.ClothesName);

}

Console.WriteLine();

}

}

}

Class1.cs

using System.Data;

namespace Task02

{

public class Clothes : IComparable<Clothes>

{

public Clothes(Material[] materials, string clothesName, bool isFashion)

{

Materials = materials;

ClothesName = clothesName;

IsFashion = isFashion;

}

public Material[] Materials { get; set; }

public string ClothesName { get; set; }

public bool IsFashion { get; set; }

public int CompareTo(Clothes other)

{

return ClothesName.CompareTo(other.ClothesName);

}

}

public class Material : IComparable<Material>

{

public Material(int price, int quality, string country)

{

Price = price;

Quality = quality;

Country = country;

}

public int Price { get; set; }

public int Quality{ get; set; }

public string Country { get; set; }

public int CompareTo(Material other)

{

return Price.CompareTo(other.Price);

}

}

public class ClotherShop

{

public ClotherShop(Clothes[] clothes, string name, string country, int costs)

{

Clothes = clothes;

Name = name;

Country = country;

this.costs = costs;

}

public int Len()

{

int length = 0;

for (int i = 0; i < Clothes.Length; i++)

{

if (Clothes[i] == null)

{

break;

}

length++;

}

return length;

}

public void Add(Clothes obj)

{

Clothes[Len()] = obj;

}

public void Delete()

{

Clothes[Len()-1] = null;

}

public void Show()

{

// define length

for(int i = 0; i < Clothes.Length; ++i)

{

if (Clothes[i] == null)

{

break;

}

Console.Write(Clothes[i].ClothesName + "\n");

}

Console.WriteLine();

}

public Clothes[] Clothes { get; set; }

public string Name { get; set; }

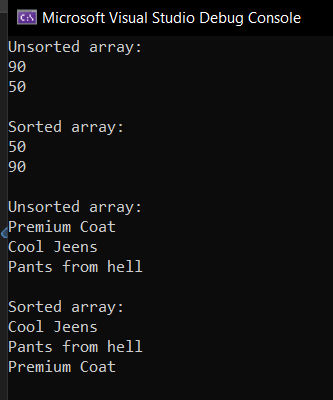
public string Country { get; set; }

public int costs { get; set; }

}

}

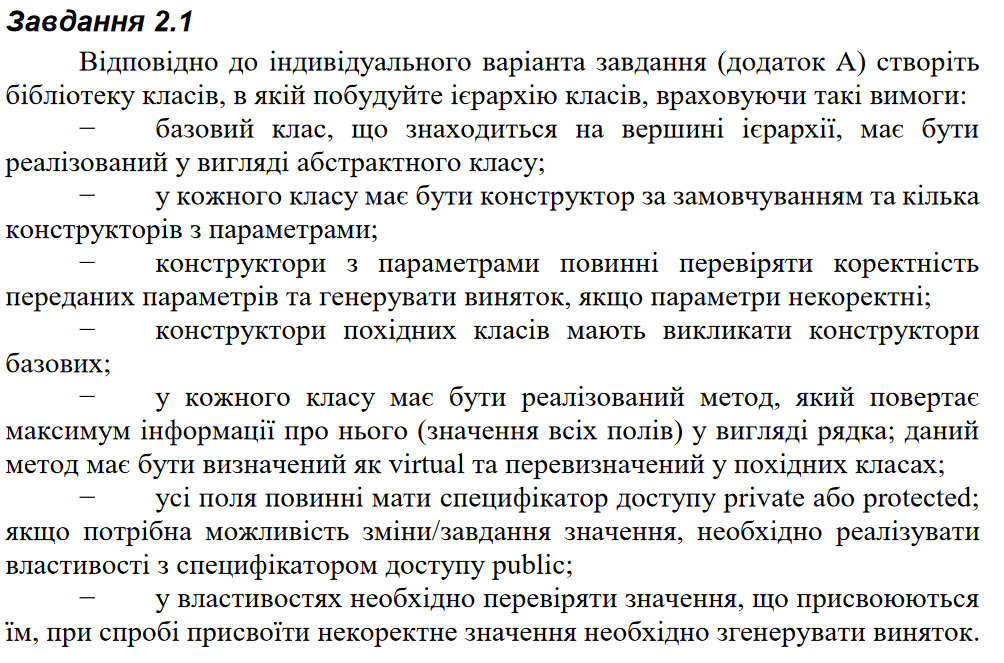
**Скриншот тестування:**

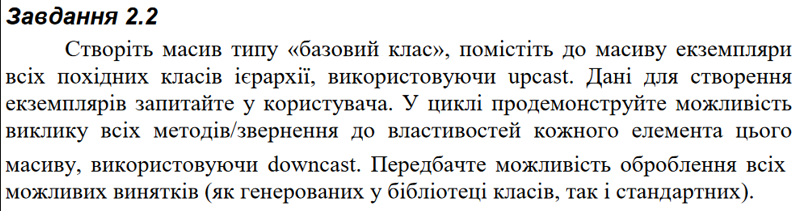


**Задача 2.1/2.2**

**Частина 1**. Постановка завдання

**Умова:**



****

**Частина 2**. Схема класу

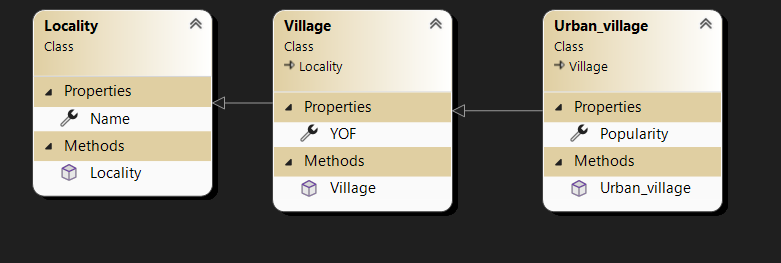


Рисунок 2

**Частина 4**. Текст програми

Program.cs

namespace Task\_2\_1

{

internal class Task\_2\_1

{

static void Main(string[] args)

{

// input data

string name;

int YOF;

int money;

int popularity;

Console.WriteLine("Enter name: ");

name = Console.ReadLine();

Locality Markivka = new Locality(name);

Console.WriteLine("Enter name: ");

name = Console.ReadLine();

Console.WriteLine("Enter year of foundation: ");

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

Village Bilokyrakina = new Village(name, YOF);

Console.WriteLine("Enter name: ");

name = Console.ReadLine();

Console.WriteLine("Enter year of foundation: ");

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

Console.WriteLine("Enter popularity: ");

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

Urban\_village Bilovodsk = new Urban\_village(name, YOF, popularity);

Locality[] cities = { Markivka, Bilokyrakina, Bilovodsk };

Village tmpV;

Urban\_village tmpUV;

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

for(int i = 0; i < 3; ++i)

{

if(i == 0)

{

Console.WriteLine("<LOCATION> Name: " + cities[i].Name);

}

else if (i == 1)

{

tmpV = (Village)cities[i];

Console.WriteLine("<VILLAGE> Name: " + tmpV.Name + "\nYear of foundation: " + tmpV.YOF);

}

else if(i == 2)

{

tmpUV = (Urban\_village)cities[i];

Console.WriteLine("<URBAN VILLAGE> Name: " + tmpUV.Name + "\nYear of foundation: " + tmpUV.YOF + "\nPopularity: " + tmpUV.Popularity);

}

}

}

}

}

Class1.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Task\_2\_1

{

class Locality

{

public Locality(string name)

{

Name = name;

}

public string Name { get; set; }

}

class Village : Locality

{

public Village(string name, int year\_of\_foundation) :base(name){

if (year\_of\_foundation > 0 && year\_of\_foundation < 2023)

{

YOF = year\_of\_foundation;

}

else

{

throw new ArgumentException();

}

}

public int YOF { get; set; }

}

class Urban\_village : Village

{

public Urban\_village(string name, int year\_of\_foundation, int popularity) : base(name, year\_of\_foundation)

{

if(popularity >= -100 && popularity <= 100)

{

Popularity = popularity;

}

else

{

throw new ArgumentException();

}

}

public int Popularity { get; set; }

}

}

**Скриншот тестування:**

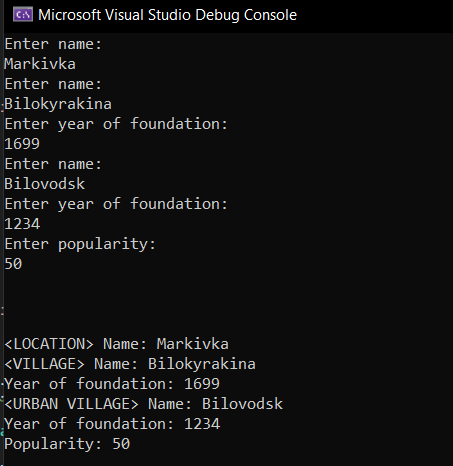


Рисунок 6 – скришот тестування

**Висновки**

Під час цієї лабораторної роботи я вивчив такі парадигми як наслідування та поліморфізм. Освоїв нові можливості мови С# для реалізації даних парадигм, вивчив призначення інтерфейсів та вперше використав у своєму коді на практиці upcast та downcast.