МІНІСТЕРСТВО ОСВІТИ ТА НАУКИ УКРАЇНИ

Національний технічний університет України

Факультет інформатики та обчислювальної техніки

«Київський політехнічний інститут імені Ігоря Сікорського»

Кафедра інформаційних систем та технологій

Звіт

з лабораторної роботи № 2

з дисципліни

«Програмування – 2»

Варіант № 20

|  |  |  |
| --- | --- | --- |
| Перевірив: |  | Виконав: П’ятак Олександр |
|  |  | cтудент гр. ІС-13 , ФІОТ |
|  |  | 1 курс |

Київ 2021

**Лабораторна робота № 2**

**Тема**: Поліморфізм. Наслідування. Перегрузка методів.

**Обладнання:** персональний комп’ютер.

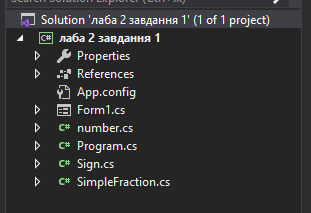
ЗАВДАННЯ

**№1** Створити додаток, який задовольняє вимогам, наведеним в завданні. Наслідування застосовувати тільки в тих завданнях, в яких воно **логічно обґрунтоване.** Аргументувати належність класу кожного створюваного методу і коректно перевизначити для кожного класу методи *Equals*, *GetHashCode*, *ToString*. При виклику будь-якого методу класу, виводити на екран текстове повідомлення.

Створити об'єкт класу Простий дріб, використовуючи клас Число, Знак. Методи: висновок на екран, додавання, віднімання, множення, ділення.

Лістинг коду:

using System;

namespace Task1

{

internal class Program

{

static void Main()

{

SimpleFraction simpleFraction1 = new SimpleFraction(new Number(3), new Number(4));

simpleFraction1 = simpleFraction1.Sum(new SimpleFraction(new Number(4), new Number(5)));

Console.Write(simpleFraction1.Numerator.Value + "/" + simpleFraction1.Denominator.Value + "\n");

SimpleFraction simpleFraction2 = new SimpleFraction(new Number(6), new Number(7));

simpleFraction2 = simpleFraction2.Difference(new SimpleFraction(new Number(3), new Number(7)));

Console.Write(simpleFraction2.Numerator.Value + "/" + simpleFraction2.Denominator.Value + "\n");

SimpleFraction simpleFraction3 = new SimpleFraction(new Number(3), new Number(9));

simpleFraction3 = simpleFraction3.Product(new SimpleFraction(new Number(4), new Number(8)));

Console.Write(simpleFraction3.Numerator.Value + "/" + simpleFraction3.Denominator.Value + "\n");

SimpleFraction simpleFraction4 = new SimpleFraction(new Number(2), new Number(3));

simpleFraction4 = simpleFraction4.Division(new SimpleFraction(new Number(5), new Number(4)));

Console.Write(simpleFraction4.Numerator.Value + "/" + simpleFraction4.Denominator.Value + "\n");

Console.WriteLine("\nPress \"enter\" to close the window...");

Console.ReadKey();

}

}

}

Клас SimpleFraction:

using System;

namespace Task1

{

class SimpleFraction

{

public Number Numerator { get; set; }

public Number Denominator { get; set; }

public Sign Operand { get; set; }

public SimpleFraction(Number value1, Number value2)

{

this.Numerator = value1;

this.Denominator = value2;

}

public SimpleFraction Sum(SimpleFraction simpleFraction2)

{

Operand = new Sign('+');

Console.Write($"{Numerator.Value}/{Denominator.Value} {Operand.Operation} {simpleFraction2.Numerator.Value}/{simpleFraction2.Denominator.Value} = ");

if (Denominator.Value == simpleFraction2.Denominator.Value)

return new SimpleFraction(new Number(Numerator.Value + simpleFraction2.Numerator.Value),

new Number(Denominator.Value));

else

return new SimpleFraction(new Number(Numerator.Value \* simpleFraction2.Numerator.Value

+ Denominator.Value \* simpleFraction2.Denominator.Value), new Number(Denominator.Value \* simpleFraction2.Denominator.Value));

}

public SimpleFraction Difference(SimpleFraction simpleFraction2)

{

Operand = new Sign('-');

Console.Write($"\n{Numerator.Value}/{Denominator.Value} {Operand.Operation} {simpleFraction2.Numerator.Value}/{simpleFraction2.Denominator.Value} = ");

if (Denominator.Value == simpleFraction2.Denominator.Value)

return new SimpleFraction(new Number(Numerator.Value - simpleFraction2.Numerator.Value),

new Number(Denominator.Value));

else

return new SimpleFraction(new Number(Numerator.Value \* simpleFraction2.Numerator.Value

- Denominator.Value \* simpleFraction2.Denominator.Value), new Number(Denominator.Value \* simpleFraction2.Denominator.Value));

}

public SimpleFraction Product(SimpleFraction simpleFraction2)

{

Operand = new Sign('\*');

Console.Write($"\n{Numerator.Value}/{Denominator.Value} {Operand.Operation} {simpleFraction2.Numerator.Value}/{simpleFraction2.Denominator.Value} = ");

return new SimpleFraction(new Number(Numerator.Value \* simpleFraction2.Numerator.Value),

new Number(Denominator.Value \* simpleFraction2.Denominator.Value));

}

public SimpleFraction Division(SimpleFraction simpleFraction2)

{

Operand = new Sign('/');

Console.Write($"\n{Numerator.Value}/{Denominator.Value} {Operand.Operation} {simpleFraction2.Numerator.Value}/{simpleFraction2.Denominator.Value} = ");

return new SimpleFraction(new Number(Numerator.Value \* simpleFraction2.Denominator.Value),

new Number(Denominator.Value \* simpleFraction2.Numerator.Value));

}

}

}

Клас Number:

using System;

namespace Task1

{

class Number

{

public int Value { get; set; }

public Number(int value)

{

this.Value = value;

}

}

}

Клас Sign:

using System;

namespace Task1

{

class Sign

{

public char Operation { get; set; }

public Sign(char value)

{

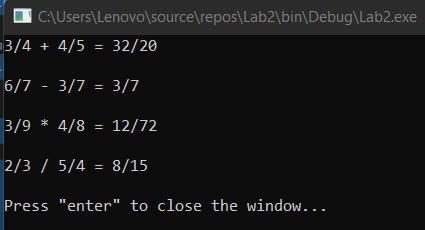
Operation = value;

}

}

}

Програма:



**№2** Створити програму, яка задовольняє наступним вимогам:

• Використовувати можливості ООП: класи, наслідування, поліморфізм, інкапсуляція.

• Кожен клас повинен мати змістовну назву та інформативний склад.

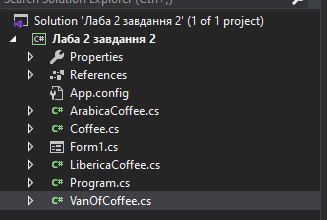
• Наслідування має застосовуватися тільки тоді, коли це має сенс.

• Класи повинні бути грамотно розкладені по пакетах.

• Консольне меню повинно бути мінімальним.

• Для зберігання параметрів ініціалізації можна використовувати файли.

Фургон кави. Завантажити фургон певного обсягу вантажем на певну суму з різних сортів кави, що знаходяться, до того ж, в різних фізичних станах (зерно, мелену, розчинну в банках і пакетиках). Враховувати обсяг кави разом з упаковкою. Провести сортування товарів на основі співвідношення ціни і ваги. Знайти в фургоні товар, відповідний заданому діапазону параметрів якості.



Лістинг коду:

using System;

namespace Task2

{

internal class Program

{

static void Main()

{

Coffee arabicaCoffee1 = new ArabicaCoffee(30, Coffee.PhysicalConditions[2], 0.8);

Coffee arabicaCoffee2 = new ArabicaCoffee(38, Coffee.PhysicalConditions[0], 0.5);

Coffee robustaCoffee1 = new RobustaCoffee(45, Coffee.PhysicalConditions[1], 1.5);

Coffee robustaCoffee2 = new RobustaCoffee(50, Coffee.PhysicalConditions[0], 1.75);

Coffee libericaCoffee1 = new LibericaCoffee(35, Coffee.PhysicalConditions[3], 1);

Coffee libericaCoffee2 = new LibericaCoffee(42, Coffee.PhysicalConditions[2], 0.6);

VanOfCoffee van1 = new VanOfCoffee(arabicaCoffee1, arabicaCoffee2, libericaCoffee1, libericaCoffee2, robustaCoffee1, robustaCoffee2);

Console.WriteLine("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Вагон кави\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_");

Console.WriteLine("\nСортування кави за цiною..");

van1.SortByPrice();

van1.ShowContent();

Console.WriteLine("\nСортування кави за вагою..");

van1.SordByWeight();

van1.ShowContent();

Console.WriteLine("\nСортування кави за сортом..");

van1.SortByGrade();

van1.ShowContent();

Console.WriteLine("\nЗнайдено каву за заданими параметрами(вага - 0.75-1кг, цiна - 26-30$):");

van1.FindAppropriateCoffee(26, 30, 0.75, 1);

Console.WriteLine("\nНатиснiть будь-яку клавiшу для закриття консолi...");

Console.ReadKey();

}

}

}

Клас Coffee:

using System;

namespace Task2

{

abstract class Coffee

{

public abstract void ShowInfoAboutGood();

public double PricePerKilo { get; set; }

public double PricePerPackage { get; set; }

public double FullPrice

{

get { return PricePerKilo \* Weight + PricePerPackage; }

}

public double Weight { get; set; }

static public string[] PhysicalConditions = { "мелена", "в зернах", "в пакетиках", "в банках"};

public string PhysicalCondition { get; set; }

}

}

Клас ArabicaCoffee:

using System;

namespace Task2

{

class ArabicaCoffee : Coffee

{

public ArabicaCoffee(double pricePerKilo, string physicalCondition, double weight)

{

this.PhysicalCondition = physicalCondition;

this.PricePerKilo = pricePerKilo;

this.Weight = weight;

if (physicalCondition == PhysicalConditions[0])

PricePerPackage = 1;

else if (physicalCondition == PhysicalConditions[1])

PricePerPackage = 1.5;

else if (physicalCondition == PhysicalConditions[2])

PricePerPackage = 2.5;

else if (physicalCondition == PhysicalConditions[3])

PricePerPackage = 4;

}

public override void ShowInfoAboutGood()

{

Console.WriteLine($"Кава арабiка {PhysicalCondition}, цiна за кiло - {PricePerKilo}$, цiна з упаковкою - {FullPrice}$, вага - {Weight}кг");

}

}

}

Клас RobustaCoffee:

using System;

namespace Task2

{

class RobustaCoffee : Coffee

{

public RobustaCoffee(double pricePerKilo, string physicalCondition, double weight)

{

this.PhysicalCondition = physicalCondition;

this.PricePerKilo = pricePerKilo;

this.Weight = weight;

if (physicalCondition == PhysicalConditions[0])

PricePerPackage = 1;

else if (physicalCondition == PhysicalConditions[1])

PricePerPackage = 1.5;

else if (physicalCondition == PhysicalConditions[2])

PricePerPackage = 2.5;

else if (physicalCondition == PhysicalConditions[3])

PricePerPackage = 4;

}

public override void ShowInfoAboutGood()

{

Console.WriteLine($"Кава робуста {PhysicalCondition}, цiна за кiло - {PricePerKilo}$, цiна з упаковкою - {FullPrice}$, вага - {Weight}кг");

}

}

}

Клас LibericaCoffee:

using System;

namespace Task2

{

class LibericaCoffee : Coffee

{

public LibericaCoffee(double pricePerKilo, string physicalCondition, double weight)

{

this.PhysicalCondition = physicalCondition;

this.PricePerKilo = pricePerKilo;

this.Weight = weight;

if (physicalCondition == PhysicalConditions[0])

PricePerPackage = 1;

else if (physicalCondition == PhysicalConditions[1])

PricePerPackage = 1.5;

else if (physicalCondition == PhysicalConditions[2])

PricePerPackage = 2.5;

else if (physicalCondition == PhysicalConditions[3])

PricePerPackage = 4;

}

public override void ShowInfoAboutGood()

{

Console.WriteLine($"Кава лiберiка {PhysicalCondition}, цiна за кiло - {PricePerKilo}$, цiна з упаковкою - {FullPrice}$, вага - {Weight}кг");

}

}

}

Класс VanOfCoffee:

using System;

namespace Task2

{

class VanOfCoffee

{

Coffee[] coffee;

public VanOfCoffee(params Coffee[] coffee)

{

this.coffee = coffee;

}

public void SortByGrade()

{

Coffee[] newCoffee = new Coffee[coffee.Length];

int index = 0;

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

{

if (coffee[i] is LibericaCoffee)

{

newCoffee[index] = coffee[i];

index++;

}

}

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

{

if (coffee[i] is ArabicaCoffee)

{

newCoffee[index] = coffee[i];

index++;

}

}

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

{

if (coffee[i] is RobustaCoffee)

{

newCoffee[index] = coffee[i];

index++;

}

}

coffee = newCoffee;

}

public void SordByWeight()

{

for (int i = 0; i < coffee.Length - 1; i++)

{

for (int j = 0; j < coffee.Length - 1; j++)

{

if (coffee[j].Weight > coffee[j + 1].Weight)

{

Coffee temp = coffee[j];

coffee[j] = coffee[j + 1];

coffee[j + 1] = temp;

}

}

}

}

public void SortByPrice()

{

for (int i = 0; i < coffee.Length - 1; i++)

{

for (int j = 0; j < coffee.Length - 1; j++)

{

if ((coffee[j].PricePerKilo + coffee[j].PricePerPackage) > (coffee[j + 1].PricePerKilo + coffee[j].PricePerPackage))

{

Coffee temp = coffee[j];

coffee[j] = coffee[j + 1];

coffee[j + 1] = temp;

}

}

}

}

public void ShowContent()

{

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

{

coffee[i].ShowInfoAboutGood();

}

}

public void FindAppropriateCoffee(double StartPrice, double FinalPrice, double StartWeight, double FinalWeight)

{

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

{

if (coffee[i].PricePerKilo >= StartPrice && coffee[i].PricePerKilo <= FinalPrice &&

coffee[i].Weight >= StartWeight && coffee[i].Weight <= FinalPrice)

{

coffee[i].ShowInfoAboutGood();

}

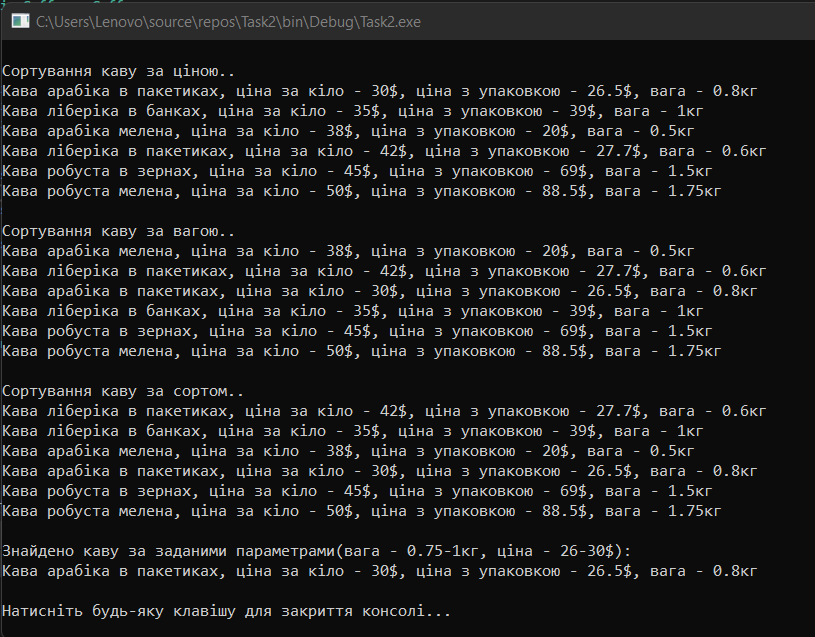
}

}

}

}

Програма:



**Висновок:** У ході виконання даної лабораторної роботи я ознайомився з основними парадигмами ООП, а саме наслідування, поліморфізм та інкапсуляція.