**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ**

Федеральное государственное бюджетное образовательное учреждение высшего образования

**«Сибирский государственный университет науки и технологий  
имени академика М.Ф. Решетнева»**

Институт информатики и телекоммуникаций

Кафедра информатики и вычислительной техники

**ЛАБОРАТОРНАЯ РАБОТА №4**

по дисциплине «Объектно-ориентированное программирование»

Вариант 21

Выполнил студент группы БПИ20-02

Очной формы обучения Сухачев Р. А.

Руководитель: канд. техн наук, доцент Вдовенко В.В.

Дата сдачи: «\_\_\_\_\_»\_\_\_\_\_\_\_\_ 2022 г.

Дата защиты: «\_\_\_\_»\_\_\_\_\_\_\_\_2022 г.

Красноярск 2022 г.

# ЦЕЛЬ РАБОТЫ

Получить практические навыки по работе с классами-интерфейсами библиотеки **FCL** платформы **.NET**.

# порядок выполнения работы

1. Изучить теоретические сведения и примеры, представленные в гл. 9 «Интерфейсы» учебного пособия В. В. Вдовенко «Разработка приложения на языке C#» (см. с. 190–204).

2. Ответить на контрольные вопросы.

3. Выполнить практическое задание.

4. Разработать **UML**-диаграмму классов для созданного приложения.

5. Составить отчет в электронном виде, который должен содержать титульный лист, цель лабораторной работы, задание, ответы на контрольные вопросы, листинг программы и результаты ее работы.

# постановка задачи

1. Изучить средства для реализации концепции множественного наследования в языке С#;
2. Изучить стандартные интерфейсы **IComparable** и **IComparer** библиотеки **FCL**;
3. Выполнить практическое задание по разработке приложения на языке С#;

Реализовать для классов, созданных при выполнении лабораторной работы 2, механизм интерфейсов. Использовать стандартные интерфейсы **IComparable** и **IComparer** для выполнения сортировки объектов по различным полям.

1. Разработать **UML**-диаграмму классов.

# ХОД РАБОТЫ

1. Изучил средства для реализации концепции множественного наследования в языке С#;
2. Изучил стандартные интерфейсы **IComparable** и **IComparer** библиотеки **FCL**;
3. Выполнил практическое задание по разработке приложения на языке С#.

Листинг программы:

using System;

namespace Lab4 // Реализовать для классов, созданных при выполнении лабораторной работы 2, механизм интерфейсов.

// Использовать стандартные интерфейсы IComparable и IComparer для выполнения сортировки объектов по различным полям.

{

interface IProgramm

{

void print();

void scan();

void Square();

void Volume();

int CountWindow { get; set; }

string Name { get; set; }

}

class Room : IProgramm, IComparable

{

string name;

double length;

double width;

double high;

int countWindow;

public Room()

{

Name = "Kitchen";

Length = 1.0;

Width = 1.0;

High = 1.0;

CountWindow = 1;

}

public Room(string name, double length, double width, double high, int countWindow)

{

this.Name = name;

this.Length = length;

this.Width = width;

this.High = high;

this.CountWindow = countWindow;

}

public void print()

{

Console.WriteLine($"\nИмя комнаты: {Name} Длина комнаты: {Length}, ширина: {Width}, высота: {High}, количество окон: {CountWindow}.\n");

}

public void scan()

{

Console.Write("\nВведите имя комнаты: ");

this.Name = Console.ReadLine()!;

Console.Write("\nВведите длину команты: ");

this.Length = Double.Parse(Console.ReadLine()!);

Console.Write("Введите ширину команты: ");

this.Width = Double.Parse(Console.ReadLine()!);

Console.Write("Введите высоту комнаты: ");

this.High = Double.Parse(Console.ReadLine()!);

Console.Write("Введите количество окон в комнате: ");

this.CountWindow = Int32.Parse(Console.ReadLine()!);

}

public void Square()

{

Console.WriteLine($"Площадь комнаты: {Length \* Width}.\n");

}

public void Volume()

{

Console.WriteLine($"Объем комнаты: {Length \* Width \* High}.\n");

}

public string Name

{

set

{

// if (value != "")

// Console.WriteLine("Значение не может быть пустым!\n");

// else

name = value;

}

get { return name; }

}

protected double Length

{

set

{

if (value < 1)

Console.WriteLine("Значение не может быть отрицательным!\n");

else

length = value;

}

get { return length; }

}

protected double Width

{

set

{

if (value < 1)

Console.WriteLine("Значение не может быть отрицательным!\n");

else

width = value;

}

get { return width; }

}

protected double High

{

set

{

if (value < 1)

Console.WriteLine("Значение не может быть отрицательным!\n");

else

high = value;

}

get { return high; }

}

public int CountWindow

{

set

{

if (value < 1)

Console.WriteLine("Значение не может быть отрицательным!\n");

else

countWindow = value;

}

get { return countWindow; }

}

int IComparable.CompareTo(object o)

{

IProgramm temp = (IProgramm)o;

if (this.CountWindow > temp.CountWindow)

return 1;

if (this.CountWindow < temp.CountWindow)

return -1;

else

return 0;

}

private class SortByNameHelper : System.Collections.Generic.IComparer<Room>

{

public SortByNameHelper() { }

int System.Collections.Generic.IComparer<Room>.Compare(Room o1, Room o2)

{

Room t1 = o1;

Room t2 = o2;

return String.Compare(t1.Name, t2.Name);

}

}

public static System.Collections.Generic.IComparer<Room> SortByName

{ get { return new SortByNameHelper(); } }

static void Main(string[] args)

{

Console.Clear();

Console.Write("\nЦель данной программы заполнение данных о комнатах с последующим их выводом на экран,");

Console.Write("а так же вычисление площади и объёма комнат.\n");

int option = 0;

while (option != 10)

{

Console.Write("\n1. Старт;\n10. ВЫХОД.\nВвод: ");

option = Convert.ToInt32(Console.ReadLine()!);

switch (option)

{

case 1:

{

Console.Write("\nВведите количество комнат: ");

int RoomsCount = Int32.Parse(Console.ReadLine()!);

var AI = new Room[RoomsCount];

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

{

AI[i] = new Room();

AI[i].scan();

}

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

{

Console.WriteLine("\nКомната №" + (i + 1));

AI[i].print();

AI[i].Square();

AI[i].Volume();

}

Console.WriteLine("\nПосле сортировки: ");

Array.Sort(AI);

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

{

Console.WriteLine("\nКомната №" + (i + 1));

AI[i].print();

}

Console.WriteLine("\nПосле сортировки по окнам: ");

Array.Sort(AI, SortByName);

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

{

Console.WriteLine("\nКомната №" + (i + 1));

AI[i].print();

}

}

break;

case 10:

{

//exit

break;

}

default:

{

Console.WriteLine("\nВведено неверное число.\nВведите число 1 или 10.");

break;

}

}

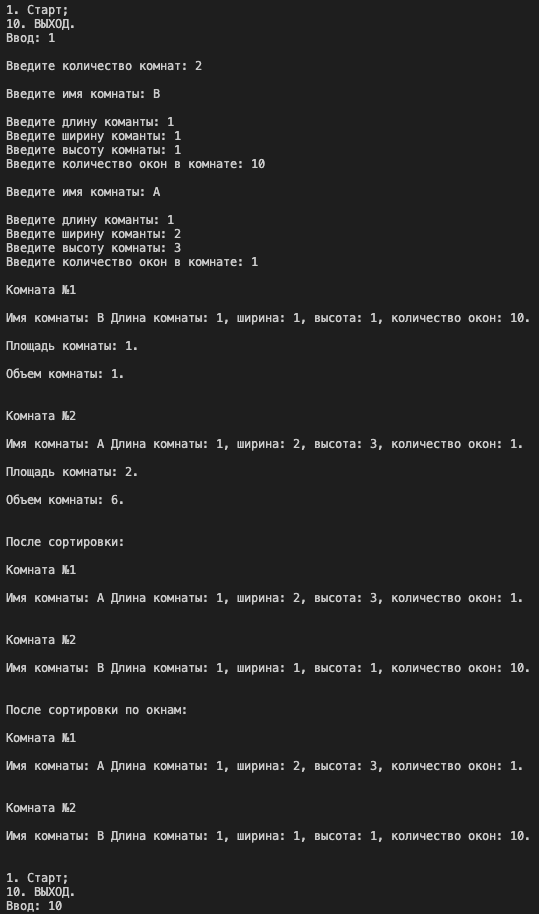
}

}

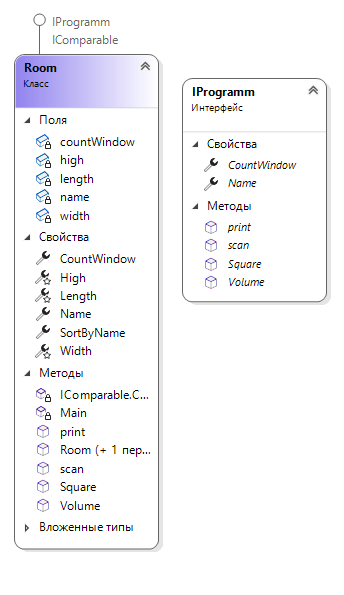
}

}

Работа программы:



1. Разработал **UML**-диаграмму классов:



**ОТВЕТЫ НА КОНТРОЛЬНЫЕ ВОПРОСЫ**

1. Дайте определение интерфейса.

Интерфейсом называется чисто абстрактный класс, содержащий только описания без реализации.

1. Назовите отличия синтаксиса интерфейса от синтаксиса абстрактного класса.

Интерфейс описывает только поведение. У него нет состояния. А у абстрактного класса состояние есть: он описывает и то, и другое. Абстрактный класс связывает между собой и объединяет классы, имеющие очень близкую связь. В то же время, один и тот же интерфейс могут реализовать классы, у которых вообще нет ничего общего. Классы могут реализовывать сколько угодно интерфейсов, но наследоваться можно только от одного класса.

1. Какое ключевое слово языка C# используется для описания интерфейса?

Слово − interface.

1. Какие объекты языка C# могут быть членами интерфейсов?

Методы, свойства, индексаторы, события, статические поля и константы.

1. С помощью какого количества классов может быть реализован интерфейс?

С помощью любого количества классов.

1. Может ли класс реализовывать множественные интерфейсы?

Да, может.

1. Необходима ли реализация методов интерфейса в классе, включающем этот интерфейс?

Да, необходима.

1. Какой модификатор доступа соответствует интерфейсу?

Public.

1. Допустимо ли явное указание модификатора доступа для интерфейса?

Да, допустимо с версии C# 8.0.

1. Приведите синтаксис интерфейса в общем виде. Проиллюстрируйте его фрагментом программы на языке C#.

interface IName

{

тип имя();

}

interface IProgramm

{

void print();

void scan();

void Square();

void Volume();

int CountWindow { get; set; }

string Name { get; set; }

}

1. Возможно ли создание ссылочной переменной интерфейсного типа?

Да, возможно.

1. Поддерживается ли в языке C# наследование интерфейсов?

Да, поддерживается.

1. Насколько синтаксис наследования интерфейсов отличается от синтаксиса наследования классов?

Синтаксис не отличается.

# ВЫВОДЫ

Были получены практические навыки по работе с классами-интерфейсами библиотеки **FCL** платформы **.NET**.