МИНИСТЕРСТВО ЭНЕРГЕТИКИ, ПРОМЫШЛЕННОСТИИ СВЯЗИ СТАВРОПОЛЬСКОГО КРАЯ

Государственное бюджетное профессиональное образовательное учреждение «Ставропольский региональный колледж вычислительной техники и электроники» (ГБПОУ СРКВТ и Э)

ОТЧЕТ О ПРАКТИЧЕСКИХ РАБОТАХ

МДК 01.01 Разработка программных модулей

ПР.09.02.07.ПЗ

Проверил Преподаватель: Баранова Т. М. подпись		Выполнила Студентка 2 курса Маркизова Е.А. Группы 27АБД подпись

Ставрополь 2023

29.03.2023 г.

Индивидуальное задание. Создание матрицы.

1. Цель и содержание

Цель лабораторной работы: изучить типы и принципы работы с массивами.

Задачи лабораторной работы:

– научиться работать с простыми массивами;

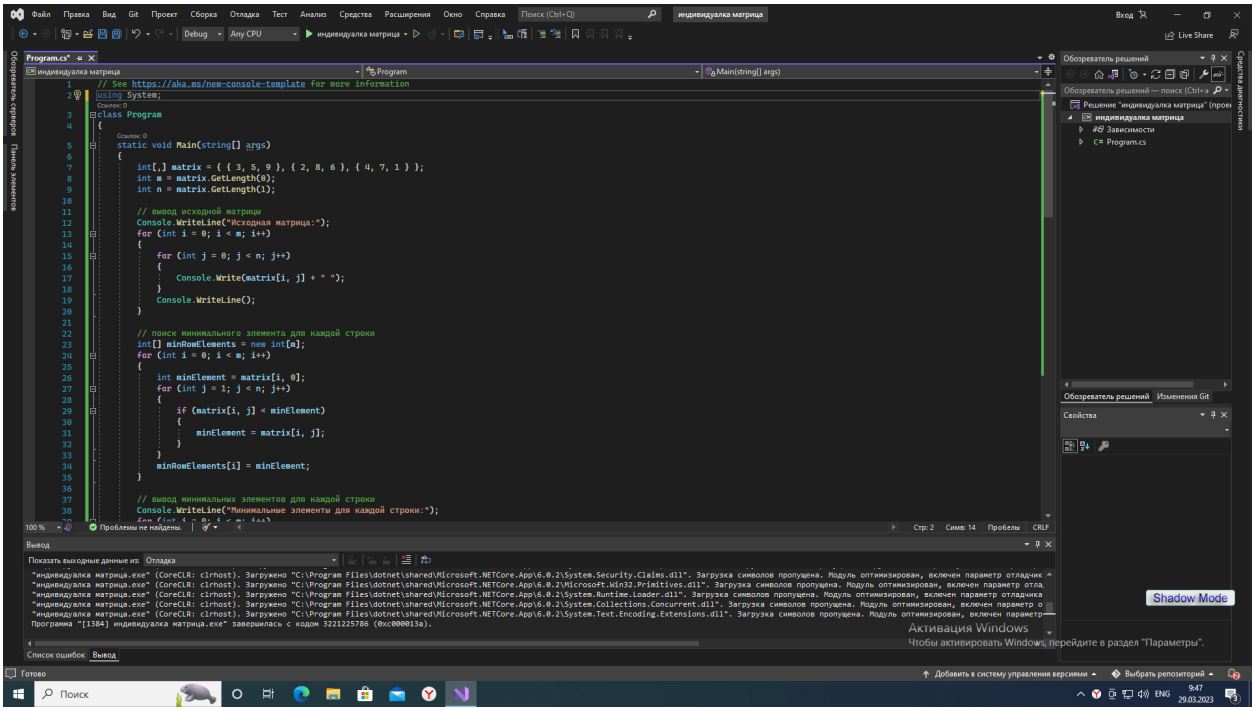
– научиться работать с многомерными массивами.

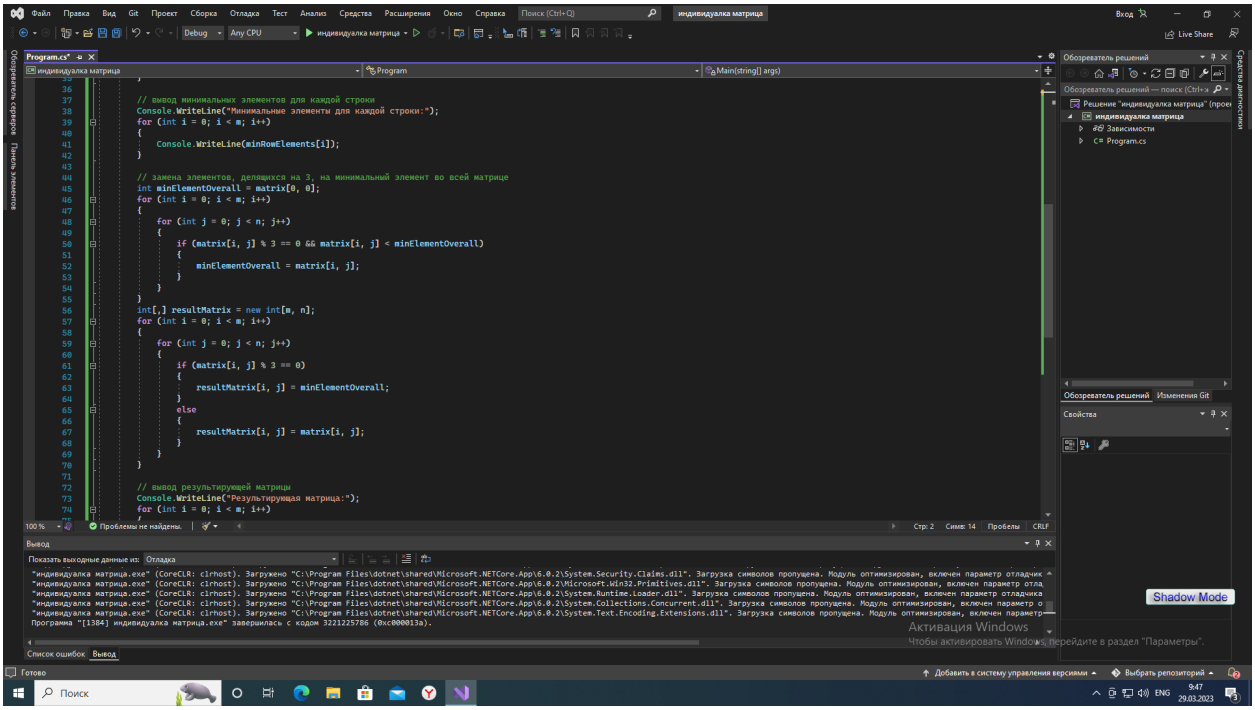
Задачи:

1. Создала консольное приложение.

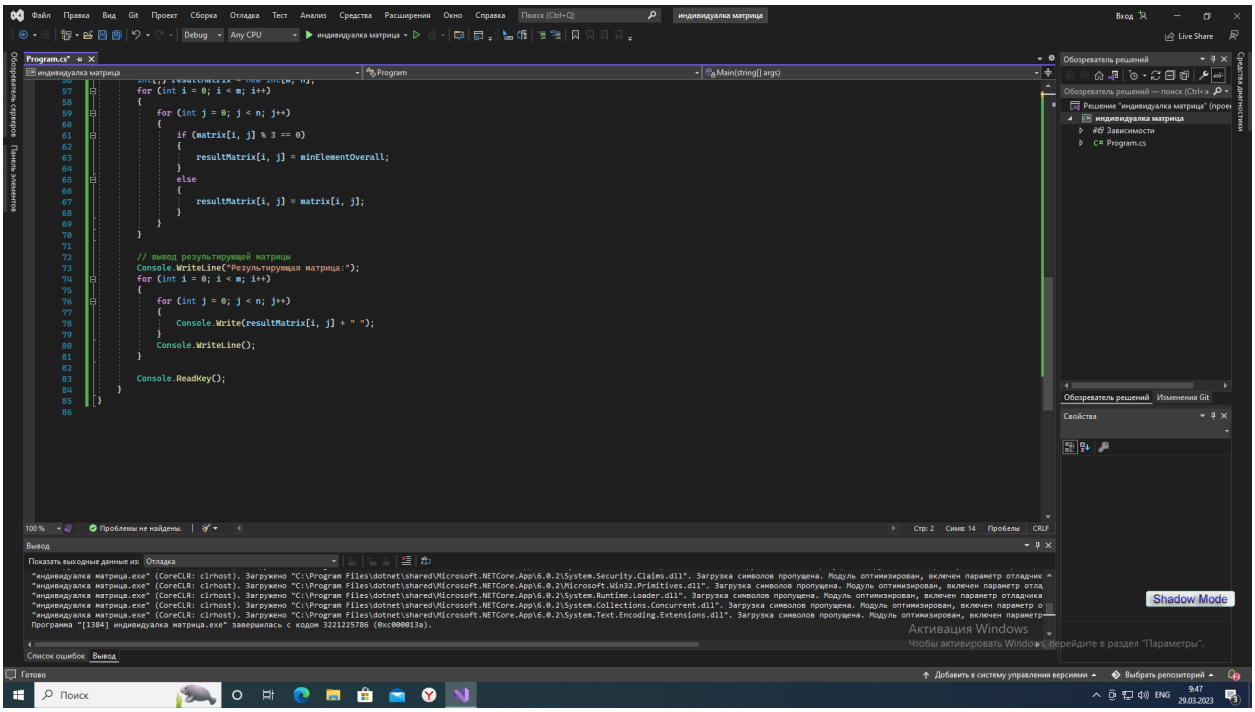
2. Изучила примеры, представленные в разделе «Теоретическое обоснование» данной лабораторной работы, повторите теоретическую часть лабораторной работы 2 (циклические конструкции).

3. Выполнила индивидуальное задание. Задания ориентированы на работу с одномерными и многомерными массивами.

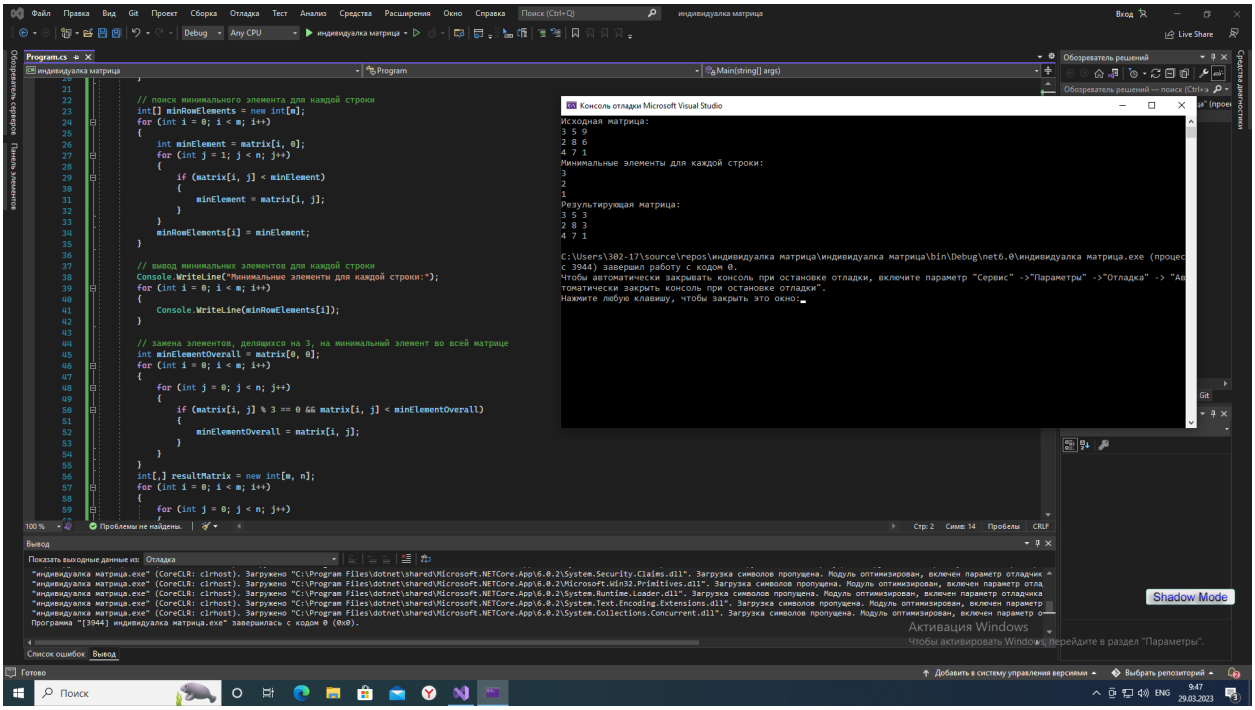




```
        // вывод минимальных элементов для каждой строки
        Console.WriteLine("Минимальные элементы для каждой строки:");
        for (int i = 0; i < m; i++)
        {
            Console.WriteLine(minRowElements[i]);
        }

        // замена элементов, делящихся на 3, на минимальный элемент во всей матрице
        int minElementOverall = matrix[0, 0];
        for (int i = 0; i < m; i++)
        {
            for (int j = 0; j < n; j++)
            {
                if (matrix[i, j] % 3 == 0 && matrix[i, j] < minElementOverall)
                {
                    minElementOverall = matrix[i, j];
                }
            }
        }
        int[,] resultMatrix = new int[m, n];
        for (int i = 0; i < m; i++)
        {
            for (int j = 0; j < n; j++)
            {
                if (matrix[i, j] % 3 == 0)
                {
                    resultMatrix[i, j] = minElementOverall;
                }
                else
                {
                    resultMatrix[i, j] = matrix[i, j];
                }
            }
        }

        // вывод результирующей матрицы
        Console.WriteLine("Результирующая матрица:");
        for (int i = 0; i < m; i++)
```



```
        for (int i = 0; i < m; i++)
        {
            for (int j = 0; j < n; j++)
            {
                if (matrix[i, j] % 3 == 0)
                {
                    resultMatrix[i, j] = minElementOverall;
                }
                else
                {
                    resultMatrix[i, j] = matrix[i, j];
                }
            }
        }

        // вывод результирующей матрицы
        Console.WriteLine("Результирующая матрица:");
        for (int i = 0; i < m; i++)
        {
            for (int j = 0; j < n; j++)
            {
                Console.Write(resultMatrix[i, j] + " ");
            }
            Console.WriteLine();
        }

        Console.ReadKey();
    }
}
```



1. Запустила среду программирования Visual Studio
2. Выбрала задание по номеру своей фамилии

   Дана исходная матрица размером $M \times N$. Вывести исходную матрицу. Вывести минимальный элемент для каждой строки и результирующую матрицу, в которой все элементы, которые делятся на 3, заменены на минимальный элемент во всей матрице.

3. Создала матрицу в языке C#
4. Задала переменные
5. Запустила проверку, в результате которой ошибок выявлено не было.

Вывод: Я научилась работать с простыми массивами и многомерными массивами.

03.04.2023 г.

ЛАБОРАТОРНАЯ РАБОТА 4. КЛАССЫ. ОБЪЕКТНОЕ МОДЕЛИРОВАНИЕ

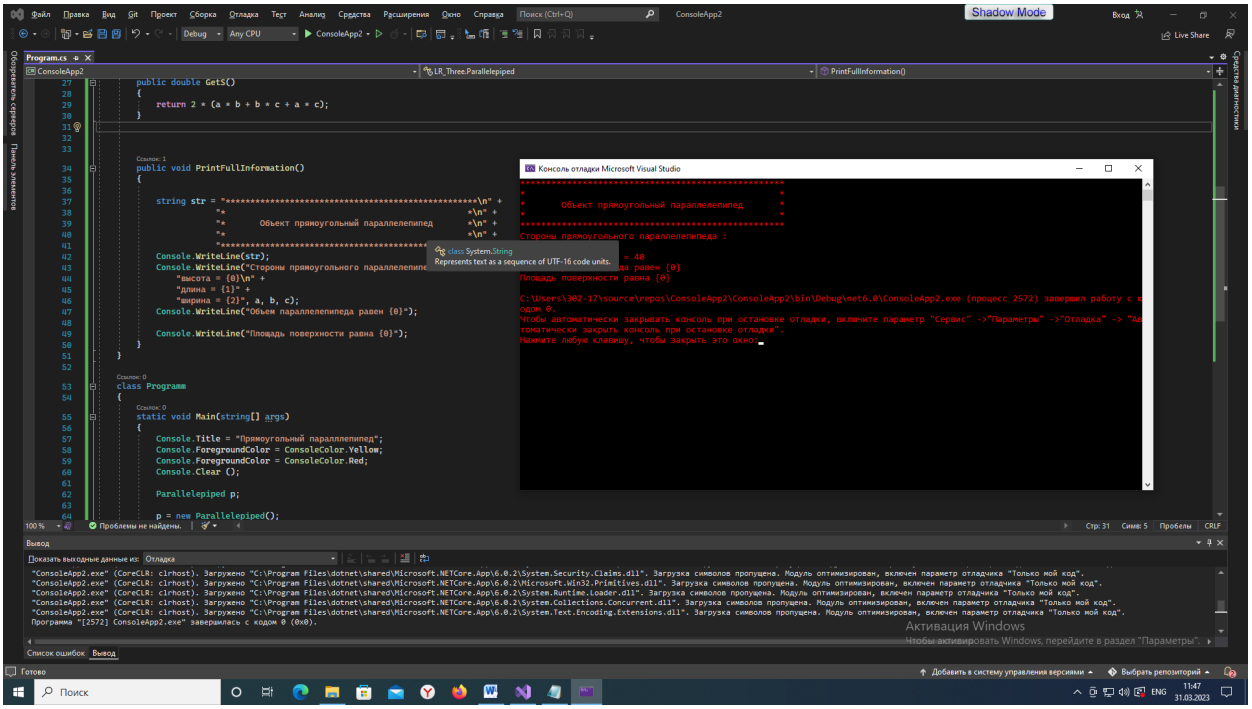
Цель лабораторной работы: изучить структуру и принципы объявления классов, освоить технологию создания экземпляров классов (объектов).

Задачи лабораторной работы:

– научиться объявлять классы;

– научиться создавать объекты классов;

– научиться работать с полями данных и методами классов.



1. Запустила среду программирования Visual Studio
2. Создайте консольное приложение в среде Visual Studio.
3. Изучила пример выполнения задания, представленный в данном разделе.
4. Выполнила индивидуальное задание. Задания ориентированы на работу с классами.
5. Запустила проверку, в результате которой ошибок выявлено не было.

Индивидуальное задание.

1. Запустила среду программирования Visual Studio
2. Выбрала задание по номеру своей фамилии

   Класс «График y=x-10». Реализовать ввод и вывод полей данных, вычисление интеграла функции от a до b (вводятся пользователем), длины отрезка функции от (a, y(a)) до (b, y(b)), а также вывод информации об объекте.
3. Задала переменные
4. Создала график в языке C#
5. Запустила проверку, в результате которой ошибок выявлено не было.

Вывод: В данной практической работе, я научилась объявлять классы; научилась создавать объекты классов; научилась работать с полями данных и методами классов.

Контрольные вопросы:

1. Класс — это логическая структура, позволяющая создавать свои собственные пользовательские типы.
2. Структура в C# является типом данных, который может содержать поля и методы, а также может быть использован для создания объектов. Она отличается от класса тем, что передает данные по значению, а не по ссылке. Структуры обычно используются для хранения небольших объемов данных или для оптимизации производительности.
3. Члены класса - это поля, методы, свойства и события, которые определяются внутри класса.
4. Я знаю три типа членов-данных: поля, свойства и события. Поля представляют данные класса, свойства обеспечивают доступ к данным класса и могут содержать логику для чтения и записи этих данных, а события позволяют классу оповещать другие части программы о происходящих внутри него событиях.
5. Я знаю три типа функций-членов класса: методы, конструкторы и деструкторы. Методы представляют поведение класса, конструкторы инициализируют объекты класса при их создании, а деструкторы освобождают ресурсы, занятые объектом класса, при его уничтожении.

6. Console.ForegroundColor = ConsoleColor.Red
7. Console.BackgroundColor = ConsoleColor.Purple
8. В C# существуют следующие модификаторы доступности членов класса:

   1. public - члены с этим модификатором доступны из любого места программы.

   2. private - члены с этим модификатором доступны только внутри класса, в котором они определены.

   3. protected - члены с этим модификатором доступны только внутри класса, в котором они определены, а также в производных классах.

   4. internal - члены с этим модификатором доступны только внутри сборки, в которой они определены.

   5. protected internal - члены с этим модификатором доступны внутри сборки, в которой они определены, а также в производных классах, даже если эти классы определены в других сборках.
9. Для объявления класса служит ключевое слово class.

03.04.2023 г.

ЛАБОРАТОРНАЯ РАБОТА 5. КОНСТРУКТОР КЛАССА. ПЕРЕГРУЗКА КОНСТРУКТОРОВ КЛАССА.

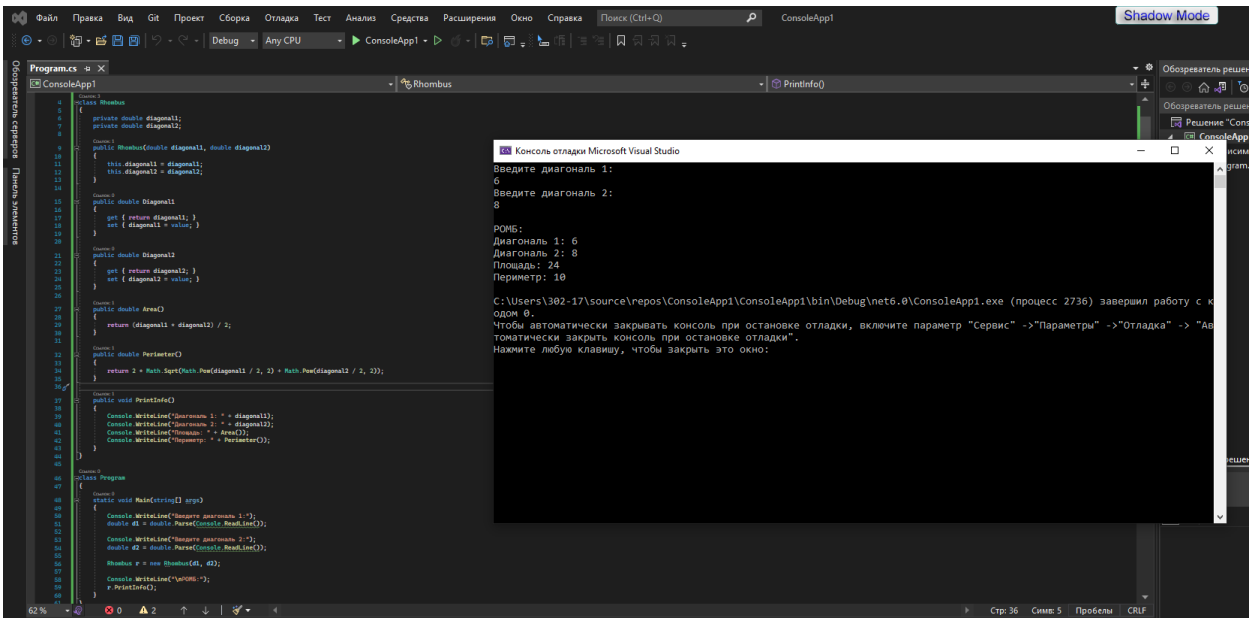
Цель лабораторной работы: понять принципы работы конструктора.

Задачи лабораторной работы:

– научиться объявлять конструктор класса;

– научиться создавать перегруженные конструкторы.

1. Для выполнения лабораторной работы, я модифицировала приложение, полученное в результате выполнения индивидуального задания лабораторной работы №4.
2. Объявила и продемонстрировала использование трех-четырех перегруженных конструкторов классов.



Контрольные вопросы:

1. Класс — это логическая структура, позволяющая создавать свои собственные пользовательские типы.
2. Конструктор класса — это специальный метод, который вызывается при создании нового объекта и используется для инициализации

полей класса значениями, а также для начальных вычислений, если они необходимы.

3. Перегрузка методов класса C# — один из мощнейших механизмов языка, который позволяет определить несколько методов с одним и тем же именем, но с различными параметрами. Кроме того, в C# по схожему принципу можно перегружать операторы и конструкторы. Такой подход делает код более гибким и понятным
4. Да, в C# один конструктор класса может вызвать другой конструктор с помощью ключевого слова this.

05.04.2023 г.

ЛАБОРАТОРНАЯ РАБОТА 6. ПРОЕКТИРОВАНИЕ ИЕРАРХИИ КЛАССОВ

Цель лабораторной работы: изучить механизм организации наследования классов.

Задачи лабораторной работы:

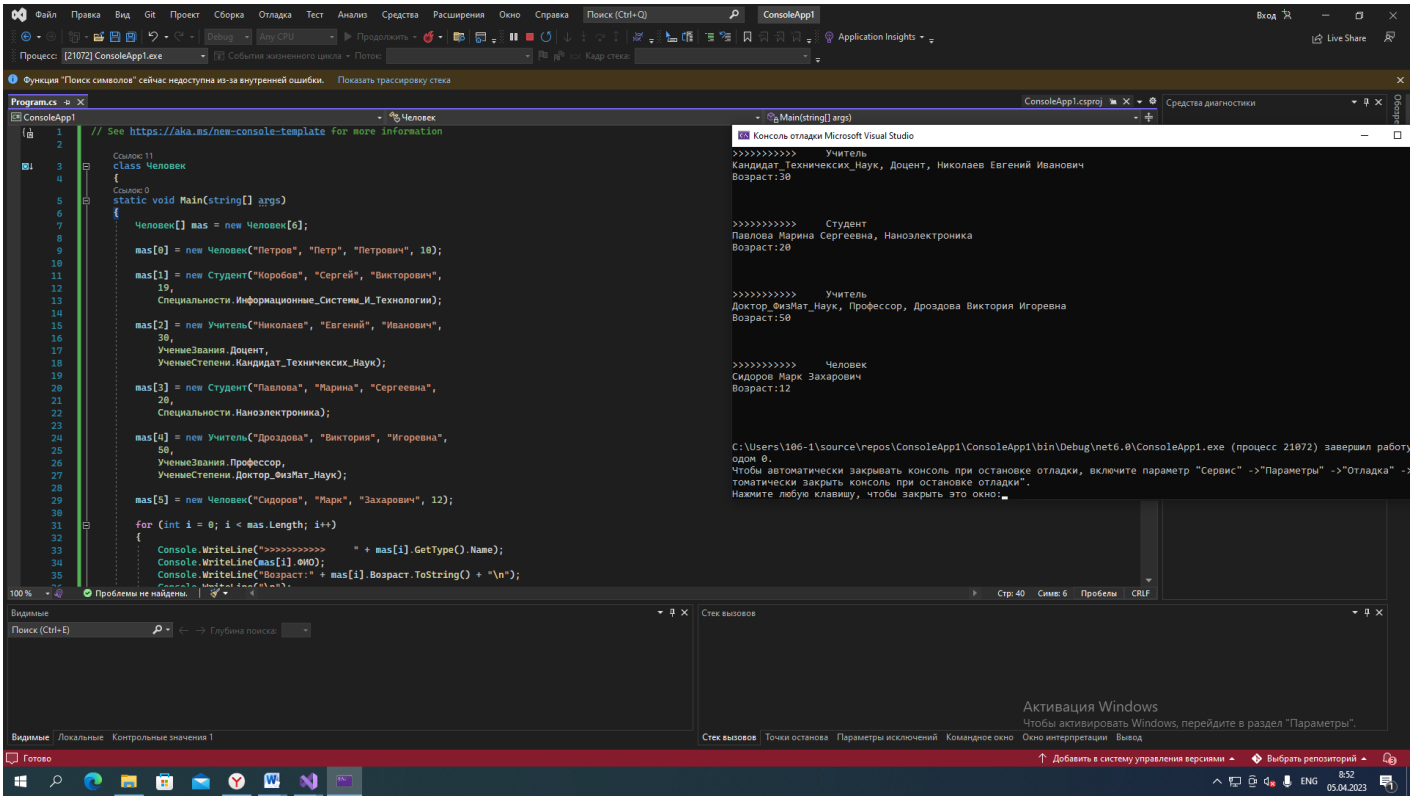
– научиться объявлять производные классы;

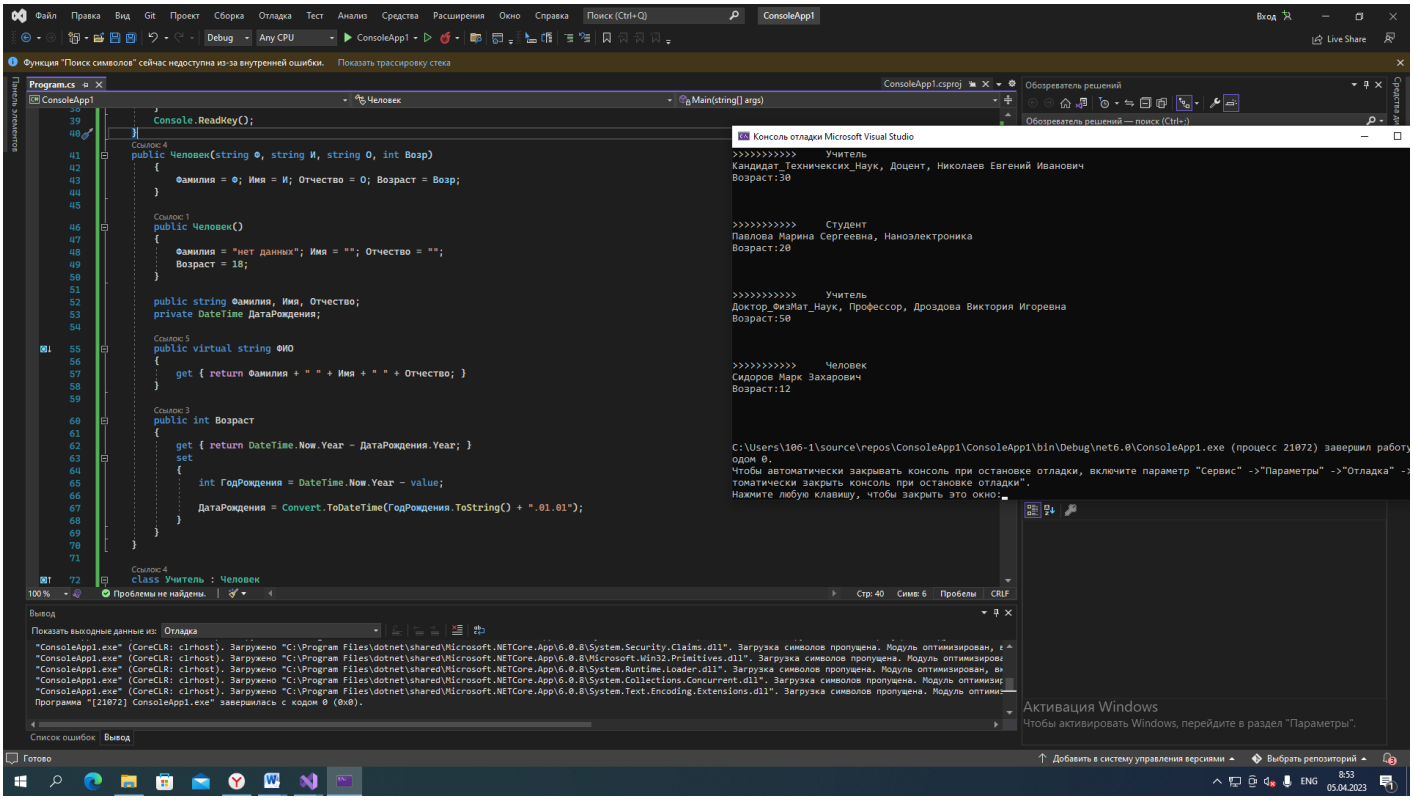
– научиться создавать иерархии классов;

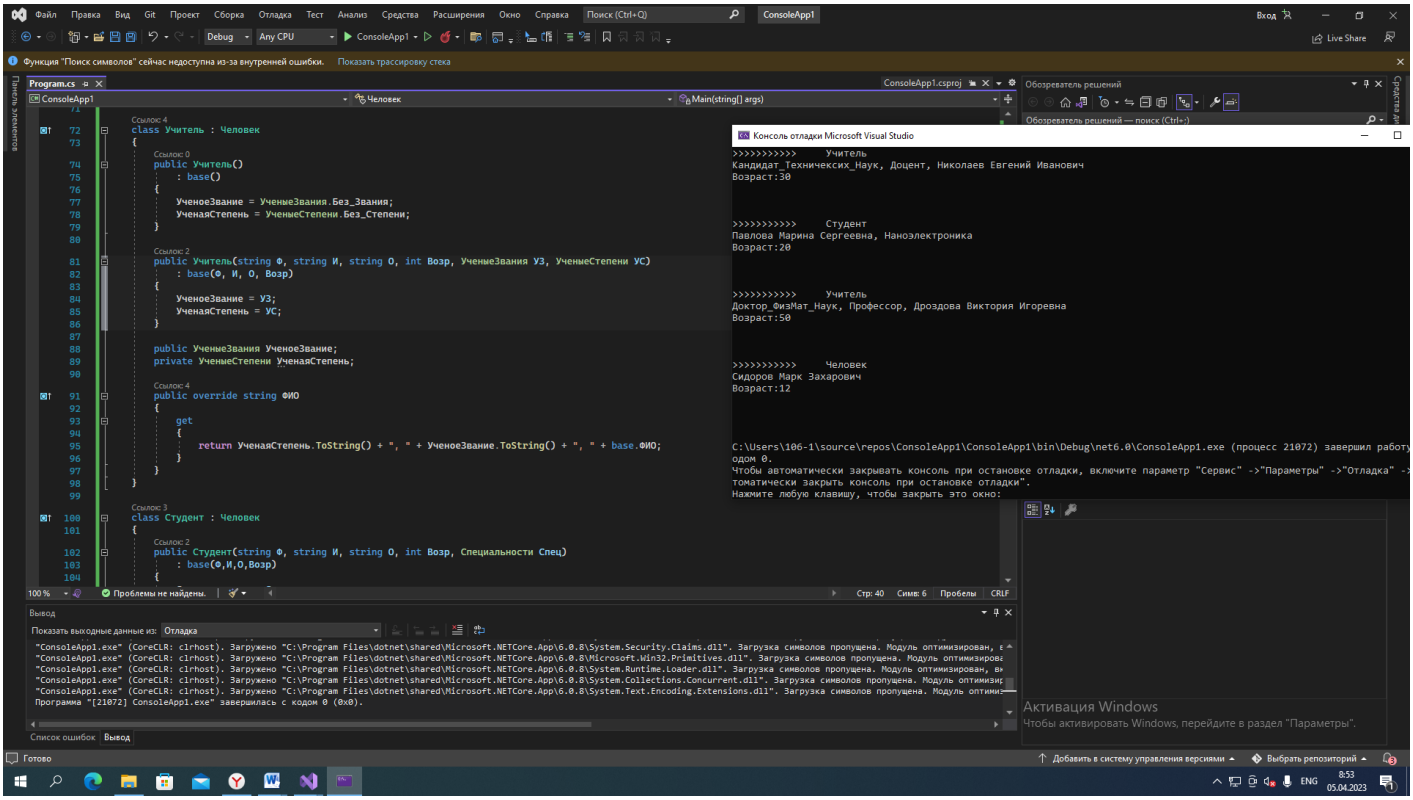
– научиться использовать механизм полиморфизма.

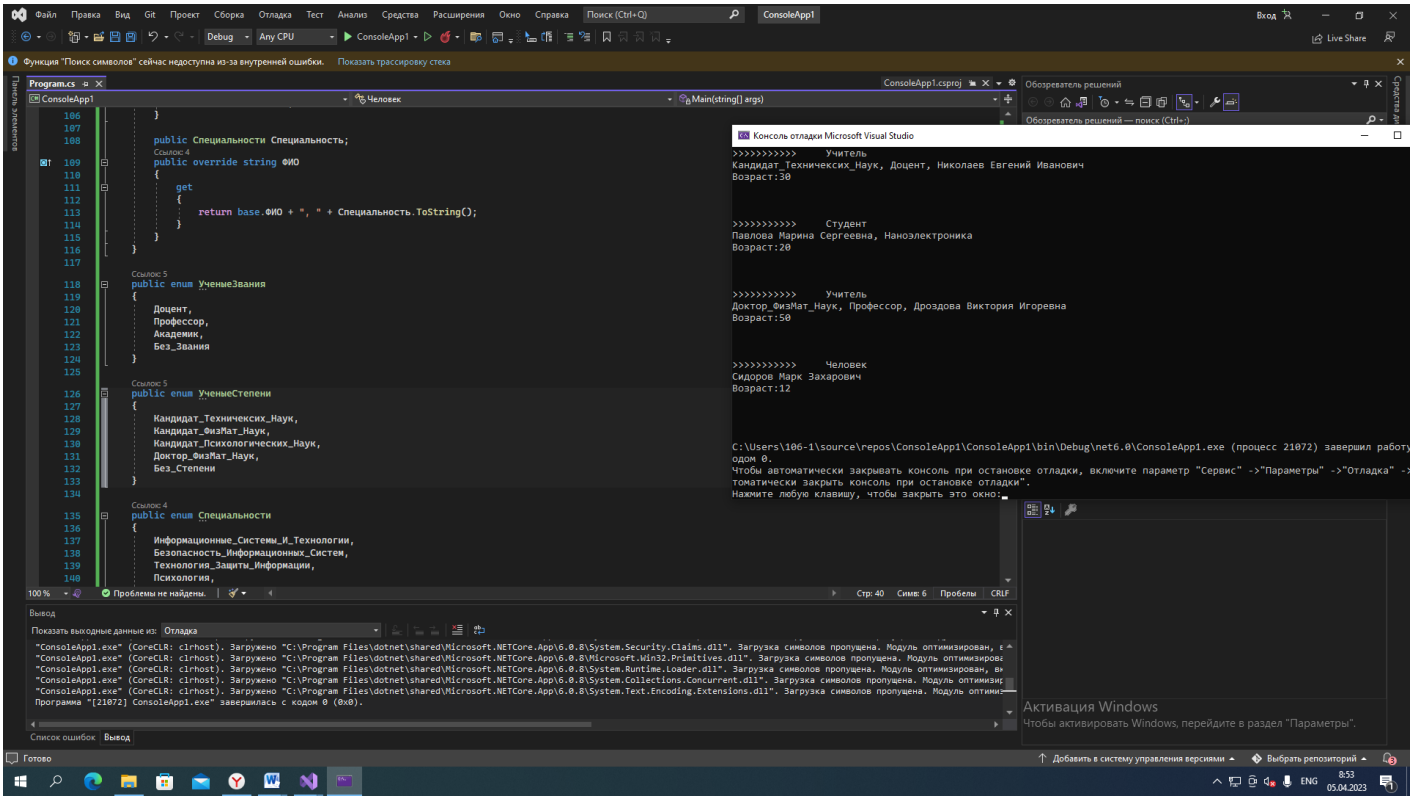
1. Запустила среду программирования Visual Studio
2. Выбрала задание по номеру своей фамилии
    a. Дана исходная матрица размером $M \times N$. Вывести исходную матрицу. Вывести минимальный элемент для каждой строки и результирующую матрицу, в которой все элементы, которые делятся на 3, заменены на минимальный элемент во всей матрице.

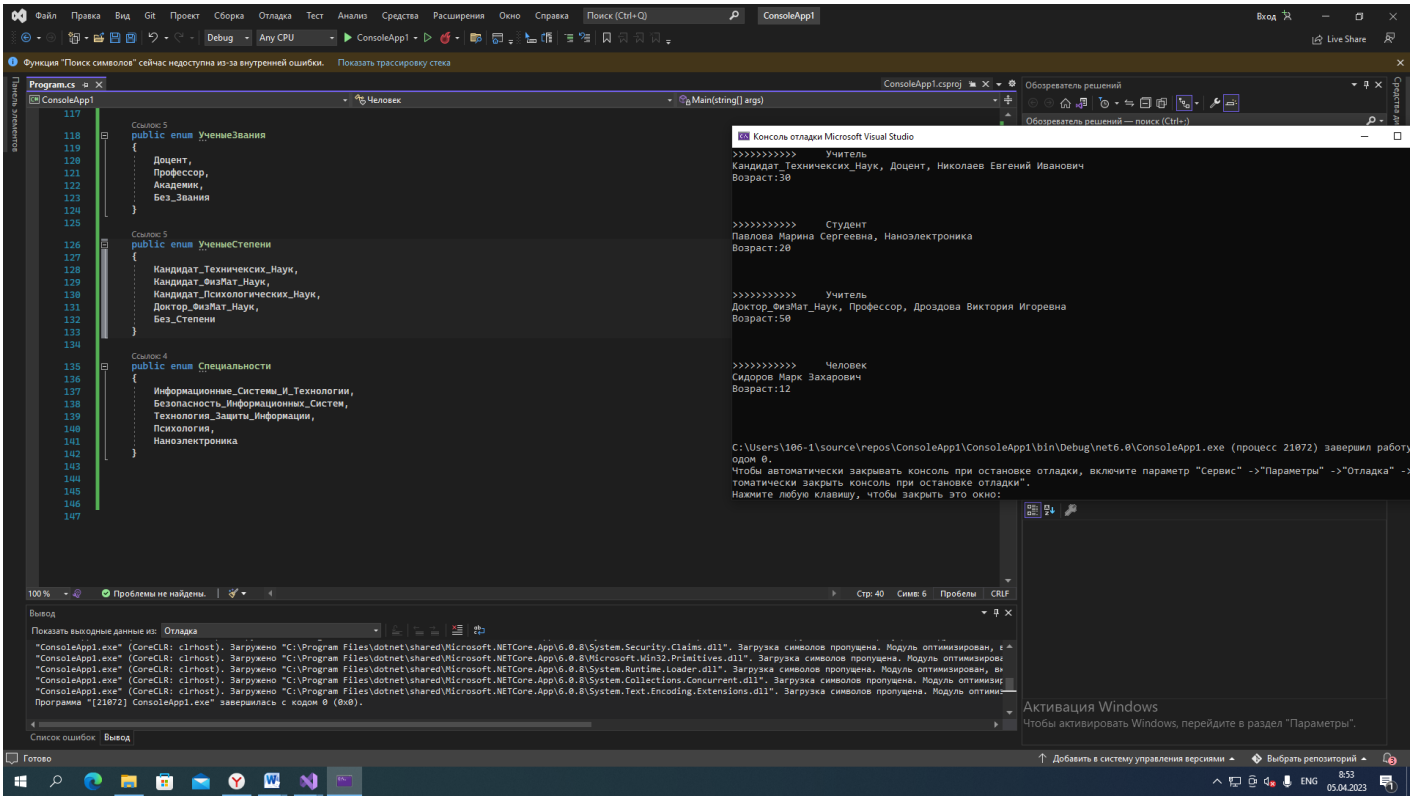
3. Создала матрицу в языке C#
4. Задала переменные
5. Запустила проверку, в результате которой ошибок выявлено не было.


// See https://aka.ms/new-console-template for more information
class Человек
static void Main(string[] args)
Человек[] mas = new Человек[6];
mas[0] = new Человек("Петров", "Петр", "Петрович", 10);
mas[1] = new Студент("Коробов", "Сергей", "Викторович",
19,
Специальности.Информационные_Системы_И_Технологии);
mas[2] = new Учитель("Николаев", "Евгений", "Иванович",
30,
УченыеЗвания.Доцент,
УченыеСтепени.Кандидат_Технических_Наук);
mas[3] = new Студент("Павлова", "Марина", "Сергеевна",
20,
Специальности.Наноэлектроника);
mas[4] = new Учитель("Дроздова", "Виктория", "Игоревна",
50,
УченыеЗвания.Профессор,
УченыеСтепени.Доктор_ФизМат_Наук);
mas[5] = new Человек("Сидоров", "Марк", "Захарович", 12);
for (int i = 0; i < mas.Length; i++)
Console.WriteLine(">>>>>>>>>>>" + mas[i].GetType().Name);
Console.WriteLine(mas[i].ФИО);
Console.WriteLine("Возраст:" + mas[i].Возраст.ToString() + "\n");
Консоль отладки Microsoft Visual Studio
>>>>>>>>>>> Учитель
Кандидат_Технических_Наук, Доцент, Николаев Евгений Иванович
Возраст:30
>>>>>>>>>>> Студент
Павлова Марина Сергеевна, Наноэлектроника
Возраст:20
>>>>>>>>>>> Учитель
Доктор_ФизМат_Наук, Профессор, Дроздова Виктория Игоревна
Возраст:50
>>>>>>>>>>> Человек
Сидоров Марк Захарович
Возраст:12
C:\Users\106-1\source\repos\ConsoleApp1\ConsoleApp1\bin\Debug\net6.0\ConsoleApp1.exe (процесс 21072) завершил работу
одом 0.
Чтобы автоматически закрывать консоль при остановке отладки, включите параметр "Сервис" ->"Параметры" ->"Отладка" ->
томатически закрыть консоль при остановке отладки".
Нажмите любую клавишу, чтобы закрыть это окно:
Активация Windows
Чтобы активировать Windows, перейдите в раздел "Параметры".


Console.ReadKey();
public Человек(string Ф, string И, string О, int Возр)
Фамилия = Ф; Имя = И; Отчество = О; Возраст = Возр;
public Человек()
Фамилия = "нет данных"; Имя = ""; Отчество = "";
Возраст = 18;
public string Фамилия, Имя, Отчество;
private DateTime ДатаРождения;
public virtual string ФИО
get { return Фамилия + " " + Имя + " " + Отчество; }
public int Возраст
get { return DateTime.Now.Year - ДатаРождения.Year; }
set
int ГодРождения = DateTime.Now.Year - value;
ДатаРождения = Convert.ToDateTime(ГодРождения.ToString() + ".01.01");
class Учитель : Человек
Консоль отладки Microsoft Visual Studio
>>>>>>>>>>> Учитель
Кандидат_Технических_Наук, Доцент, Николаев Евгений Иванович
Возраст:30
>>>>>>>>>>> Студент
Павлова Марина Сергеевна, Наноэлектроника
Возраст:20
>>>>>>>>>>> Учитель
Доктор_ФизМат_Наук, Профессор, Дроздова Виктория Игоревна
Возраст:50
>>>>>>>>>>> Человек
Сидоров Марк Захарович
Возраст:12
C:\Users\106-1\source\repos\ConsoleApp1\ConsoleApp1\bin\Debug\net6.0\ConsoleApp1.exe (процесс 21072) завершил работу
одом 0.
Чтобы автоматически закрывать консоль при остановке отладки, включите параметр "Сервис" ->"Параметры" ->"Отладка" ->
томатически закрыть консоль при остановке отладки".
Нажмите любую клавишу, чтобы закрыть это окно:
Активация Windows
Чтобы активировать Windows, перейдите в раздел "Параметры".


Program.cs
ConsoleApp1
Человек
Main(string[] args)
class Учитель : Человек
public Учитель()
: base()
УченоеЗвание = УченыеЗвания.Без_Звания;
УченаяСтепень = УченыеСтепени.Без_Степени;
public Учитель(string Ф, string И, string О, int Возр, УченыеЗвания УЗ, УченыеСтепени УС)
: base(Ф, И, О, Возр)
УченоеЗвание = УЗ;
УченаяСтепень = УС;
public УченыеЗвания УченоеЗвание;
private УченыеСтепени УченаяСтепень;
public override string ФИО
get
return УченаяСтепень.ToString() + ", " + УченоеЗвание.ToString() + ", " + base.ФИО;
class Студент : Человек
public Студент(string Ф, string И, string О, int Возр, Специальности Спец)
: base(Ф,И,О,Возр)
Консоль отладки Microsoft Visual Studio
>>>>>>>>>>> Учитель
Кандидат_Техничекских_Наук, Доцент, Николаев Евгений Иванович
Возраст:30
>>>>>>>>>>> Студент
Павлова Марина Сергеевна, Наноэлектроника
Возраст:20
>>>>>>>>>>> Учитель
Доктор_ФизМат_Наук, Профессор, Дроздова Виктория Игоревна
Возраст:50
>>>>>>>>>>> Человек
Сидоров Марк Захарович
Возраст:12
C:\Users\106-1\source\repos\ConsoleApp1\ConsoleApp1\bin\Debug\net6.0\ConsoleApp1.exe (процесс 21072) завершил работу
одом 0.
Чтобы автоматически закрывать консоль при остановке отладки, включите параметр "Сервис" ->"Параметры" ->"Отладка" ->
томатически закрыть консоль при остановке отладки".
Нажмите любую клавишу, чтобы закрыть это окно:
Вывод
Показать выходные данные из: Отладка
"ConsoleApp1.exe" (CoreCLR: clrhost). Загружено "C:\Program Files\dotnet\shared\Microsoft.NETCore.App\6.0.8\System.Security.Claims.dll". Загрузка символов пропущена. Модуль оптимизирован,
"ConsoleApp1.exe" (CoreCLR: clrhost). Загружено "C:\Program Files\dotnet\shared\Microsoft.NETCore.App\6.0.8\Microsoft.Win32.Primitives.dll". Загрузка символов пропущена. Модуль оптимизирова
"ConsoleApp1.exe" (CoreCLR: clrhost). Загружено "C:\Program Files\dotnet\shared\Microsoft.NETCore.App\6.0.8\System.Runtime.Loader.dll". Загрузка символов пропущена. Модуль оптимизирован,
"ConsoleApp1.exe" (CoreCLR: clrhost). Загружено "C:\Program Files\dotnet\shared\Microsoft.NETCore.App\6.0.8\System.Collections.Concurrent.dll". Загрузка символов пропущена. Модуль оптимизир
"ConsoleApp1.exe" (CoreCLR: clrhost). Загружено "C:\Program Files\dotnet\shared\Microsoft.NETCore.App\6.0.8\System.Text.Encoding.Extensions.dll". Загрузка символов пропущена. Модуль оптими
Программа "[21072] ConsoleApp1.exe" завершилась с кодом 0 (0x0).
Активация Windows
Чтобы активировать Windows, перейдите в раздел "Параметры".
Список ошибок
Вывод
Готово
Добавить в систему управления версиями
Выбрать репозиторий
ENG
8:53
05.04.2023





Индивидуалка

Задачи:

1. Создайте консольное приложение.
2. Изучите пример создания иерархии классов, представленный в разделе

«Теоретическое обоснование» данной лабораторной работы.

3. Постройте свою иерархию классов в соответствии с индивидуальным заданием. В результате выполнения лабораторной работы должны быть реализованы следующие механизмы:

– использование типа-перечисления (хотя бы одного);

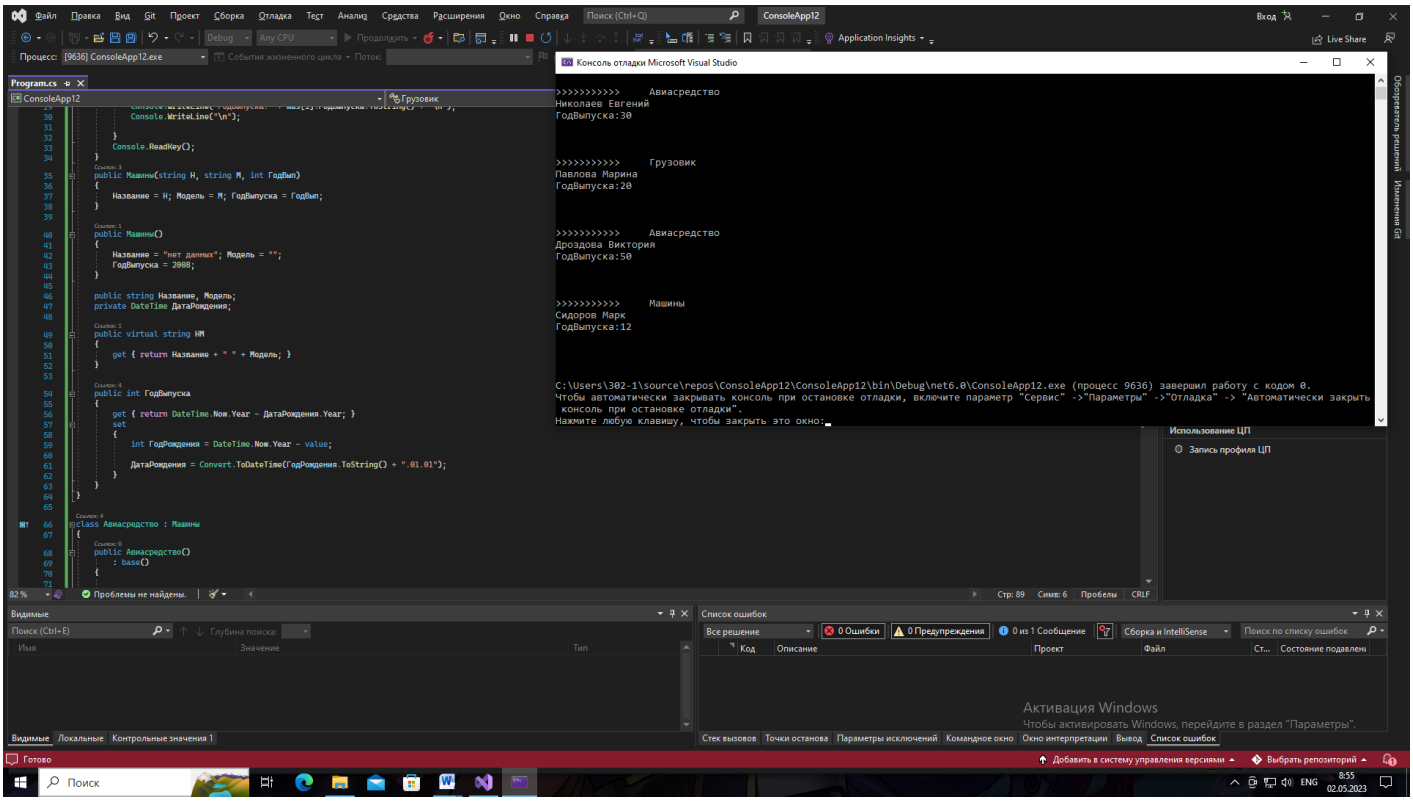
– использование переопределенного свойства (хотя бы одного);

– использование переопределенного метода (хотя бы одного);

– использование вызова базового конструктора;

– использование вызова любого базового метода (отличного от конструктора).

4. Продемонстрируйте использование классов, созданной иерархии (легче всего это сделать с использованием массивов). При защите работы укажите признаки присутствия полиморфного поведения в программе (реализация полиморфизма).



Вывод: я молодец)))

Вопросы:

1.Наследование реализации в объектно-ориентированном программировании означает возможность класса наследовать реализацию методов и свойств от другого класса. Это

позволяет избежать дублирования кода и повторного написания одинаковых методов и свойств для каждого класса.

Синтаксически наследование реализации описывается в языке C# с помощью ключевого слова "class", после которого указывается имя класса, за которым следует двоеточие и имя базового класса, от которого происходит наследование. Например:

```
class MyClass : BaseClass
{
   // код класса
}
```

Здесь класс MyClass наследует реализацию методов и свойств от базового класса BaseClass. Также возможно использовать ключевое слово "override" для переопределения методов базового класса в производном классе.
2. Ключевое слово base в языке C# используется для обращения к методам и свойствам базового класса из производного класса. Это позволяет переопределить методы базового класса в производном классе, но при этом сохранить доступ к реализации метода базового класса. Например, если в производном классе переопределен метод, но требуется вызвать реализацию базового метода внутри переопределенного метода, можно использовать ключевое слово base, например:

```
class MyBaseClass
{
   public virtual void MyMethod()
   {
      Console.WriteLine("Реализация метода в базовом классе");
   }
}
```

```
class MyDerivedClass : MyBaseClass
{
   public override void MyMethod()
   {
      base.MyMethod(); // вызов реализации метода из базового класса
      Console.WriteLine("Реализация метода в производном классе");
   }
}
```

В данном примере метод MyMethod() в производном классе MyDerivedClass переопределен и вызывает реализацию метода из базового класса с помощью ключевого слова base, а затем добавляет свою реализацию.

3. Да, в языке C# можно переопределить метод класса с помощью ключевого слова override. Также можно переопределить свойства класса с помощью ключевых слов get и set. Однако, данные класса нельзя переопределить, только изменить их значения.

4. Для переопределения метода в производном классе в C# нужно использовать ключевое слово `override` перед сигнатурой метода. Например:

```
class BaseClass
{
   public virtual void Method()
   {
      Console.WriteLine("Base method");
   }
}

class DerivedClass : BaseClass
{
   public override void Method()
```

```
    {
        Console.WriteLine("Derived method");
    }
}
```

В данном примере метод `Method()` из базового класса `BaseClass` переопределяется в производном классе `DerivedClass`. При вызове метода на объекте класса `DerivedClass`, будет выполнен переопределенный метод `Method()` из производного класса, а не базовый метод.

5. Ключевое слово virtual в C# используется для указания, что метод или свойство может быть переопределено в производном классе. Это позволяет создавать иерархию классов, где производные классы могут изменять поведение методов и свойств базового класса. Ключевое слово virtual необходимо для использования ключевого слова override при переопределении метода или свойства в производном классе.

6. Ключевое слово override используется для переопределения метода или свойства базового класса в производном классе. При использовании ключевого слова override производный класс может изменять поведение метода или свойства базового класса, что позволяет создавать более специализированные версии методов и свойств для конкретных потребностей производного класса.
7. Для изменения цвета фона в консольном приложении необходимо использовать метод Console.BackgroundColor и задать ему нужный цвет из перечисления ConsoleColor. Например, для установки фона красного цвета необходимо использовать следующий код:

```
Console.BackgroundColor = ConsoleColor.Red;
```

Затем можно выводить текст на экран с помощью метода Console.WriteLine или других методов вывода. Цвет фона будет применяться ко всему тексту, выводимому после установки его значения. После окончания работы с нужным цветом фона его можно

вернуть к стандартному значению, используя метод Console.ResetColor():

```
Console.ResetColor();
```

8. Для построения диаграммы типов данных в Visual Studio 2008/2010 необходимо выполнить следующие действия:

1. Открыть проект в Visual Studio.
2. Выбрать в меню "Project" пункт "Add New Item".
3. В появившемся окне выбрать тип элемента "Class Diagram" и задать ему имя.
4. Нажать кнопку "Add".
5. В открывшейся диаграмме перетащить на нее нужные классы и другие типы данных из проекта.
6. Расположить элементы на диаграмме и соединить их связями, указывая тип связи и ее направление.
7. Добавить комментарии и описания к элементам диаграммы, если необходимо.
8. Сохранить диаграмму и закрыть ее.

После построения диаграммы типов данных можно использовать ее для анализа структуры приложения, выявления зависимостей между классами и типами данных, а также для документирования кода.

9. В C# любой класс может иметь только один базовый класс.