

**ОТЧЕТ О ВЫПОЛНЕНИИ**

**практической работы №12**

**Тема: «Классы и объекты».**

Выполнил: ст.гр. Алирзаева З.Н

Специальность: 09.02.07

Информационные системы и программирование

Проверил: преподаватель Боклач Б. И.

Москва

2023

**Цель работы**:

Изучение понятия класса как пользовательского типа данных и приобретение навыков работы с классами.

**Вариант 1**.

**Задание:** Дана вещественная матрица размером 4 строки, 5 столбцов. Переставляя ее строки и столбцы, добейтесь того, чтобы наибольший элемент (один из них) оказался в верхнем левом углу.

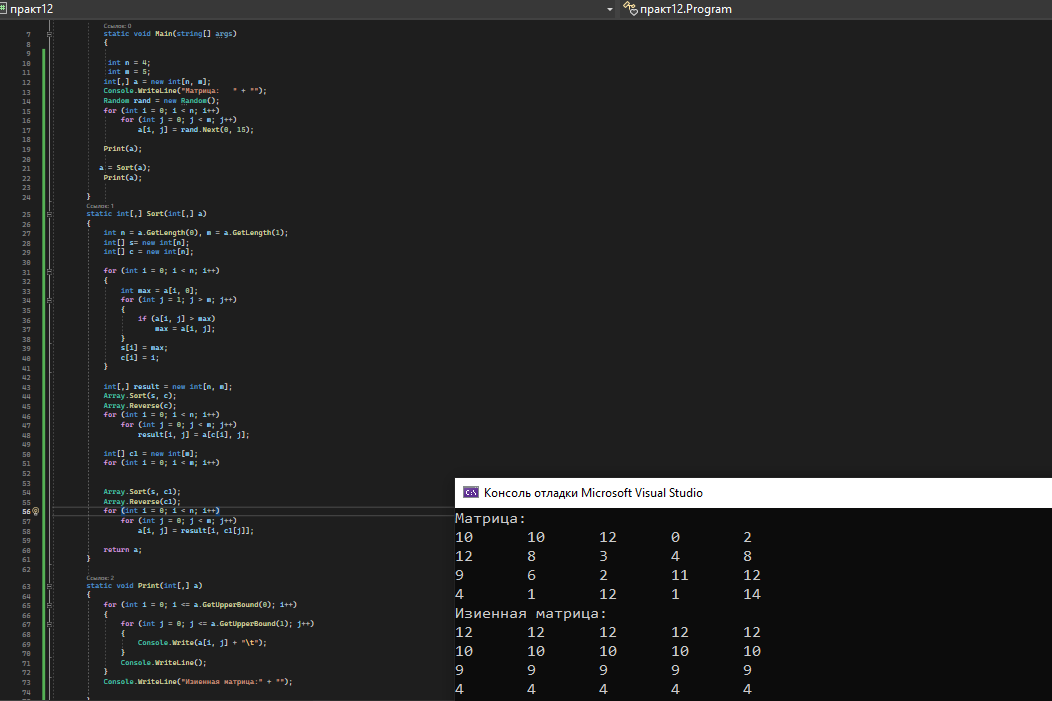


Рис 1.

**Вывод:**

В данной практической работе я познакомилась с разными задачами, которые имели каждый свой способ решения, а также овладела практическими навыками как пользовательского типа данных и приобретение навыков работы с классами.

**Контрольные вопросы:**

1. **Какие модификаторы доступа есть в C#?**

* **private.** Для закрытого (приватного) компонента класса либо структуры. ...
* **private protected. ...**
* **protected. ...**
* **internal. ...**
* **protected internal. ...**
* **public:** это публичный или общедоступный компонент структуры/класса.

1. **Если классы и члены класса не имеют никаких модификаторов, какие модификаторы доступа к ним применяются по умолчанию?**

Для классов, размещенных непосредственно в пространстве имен, модификатор доступа по умолчанию – **internal.**

1. **Что выведет на консоль следующая программа и почему?**

class Person

{

int age = 26;

string name = "Tom";

public Person(int age, string name)

{

this.age = age;

this.name = name;

}

}

class Program

{

static void Main(string[] args)

{

Person person = new Person(19, "Bob"); Console.WriteLine(person.name);

Console.ReadKey();

}

}

Программа не скомпилируется, поскольку переменная name, к которой мы хотим обратиться в методе Main, не имеет модификаторов. Следовательно, по умолчанию она имеет модификатор доступа **private**. Поэтому вне класса Person к этой переменной мы обратиться не можем

1. **Дан следующий класс:**

class Person {

public string name = "Sam";

public int age;

public Person(string name, int age)

{

this.name = name;

this.age = age; }

}

**Какое значение поле name будет иметь при выполнение следующего кода и почему? Person tom = new Person("Tom", 34) { name = "Bob", age = 29 };**

Поле **name** объекта **tom** будет иметь значение **"Bob"**. В самом начале выполняется начальная инициализация полей класса, и полю name присваивается значение **"Sam"**. Далее выполняется конструктор, который устанавливает для переменной name значение **"Tom"**. И в последнюю очередь выполняется инициализатор, который устанавливает для поля name значение **"Bob"**.