

1 Самостоятельная работа

Тема: Определение классов

Задание 1

Создайте в Visual Studio проект консольного типа.
Определите в проекте ряд классов, перечисленных ниже в задании.

Требования:

- ✓ каждый класс определить в отдельном файле.
- ✓ каждый класс снабдить переопределенным методом

ToString(), возвращающим строку с полным описанием данных экземпляра класса.

определите в каждом классе специальный конструктор, принимающий значения каждого поля/свойства класса

- ✓ в методе Main() создайте по одному экземпляру каждого определённого класса. Используя конструктор задайте каждому полю/свойству значения (адекватные, читаемые)
- ✓ используя метод ToString() получите строковое представление каждого объекта. Выведите в консоль полученное строковое представление каждого объекта.

Перечень классов:

1. Студент.
Свойства: Имя, Курс, Половая принадлежность (доступно только для чтения, т.е. readonly. *Загугли, если не знаешь, что это такое*)
2. Служащий
Свойства: Имя, Профессия, Рабочий стаж
3. Цех
Свойства: Строковый шифр, количество служащих
4. Книга
Свойства: Название (Заголовок), Список фио авторов, Стоимость
5. Зачёт
Свойства: ФИО экзаменуемого, ФИО экзаменатора, Дата, Оценка
6. Адрес
Свойства: Почтовый индекс, Город, Улица, Дом, квартира
7. Товар
Свойства: Наименование, Количество, Стоимость, Срок годности
8. Учебная группа
Свойства: Шифр, Количество студентов, Год формирования, Специальность

9. Денежная купюра

Свойства: Серия и номер, Номинал, Номинал прописью

10. Компьютерная игра

Свойства: Название, Фирма разработчик, Год издания, Жанр

Листинг программы:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ConsoleApp3_Chumachenko
{
    class Student
    {
        public string Name { get; set; }
        public int Course { get; set; }
        readonly public string Gender;
        public override string ToString()
        {
            return ($"Студент: {Name} {Course} {Gender}");
        }
        public Student(string name, int course, string gender)
        {
            Name = name;
            Course = course;
            Gender = gender;
        }
    }

    class Robotyaga //служачий
    {
        public string Name { get; set; }
        public string Profession { get; set; }
        public int WorkExperience { get; set; } //рабочий стаж
        public override string ToString()
        {
            return ($"Роботяга: {Name} {Profession} {WorkExperience}");
        }
        public Robotyaga(string name, string profession, int workExperience)
        {
            Name = name;
            Profession = profession;
            WorkExperience = workExperience;
        }
    }

    class WorkShop //цех
    {
        public string StringCipher { get; set; } //строковый шифр
        public int NumberOfEmpliyees { get; set; } //кол-во служащих
        public override string ToString()
        {
            return ($" Рабочий цех: {StringCipher} {NumberOfEmpliyees}");
        }
    }
}
```

```

        public WorkShop(string stringCipher, int numberOfEmpliyees)
        {
            StringCipher = stringCipher;
            NumberOfEmpliyees = numberOfEmpliyees;
        }
    }
    class Book
    {
        public string Name { get; set; }
        public string Authors { get; set; } //список авторов
        public double Price { get; set; }
        public override string ToString()
        {
            return ("Книга: {Name} {Authors} {Price}");
        }
        public Book(string name, string authors, double price)
        {
            Name = name;
            Authors = authors;
            Price = price;
        }
    }
    class Zachet
    {
        public string StudentFio { get; set; }
        public string TeacherFio { get; set; }
        public string Date { get; set; }
        public int Rating { get; set; } //оценка
        public override string ToString()
        {
            return ("Зачет: {StudentFio} {TeacherFio} {Date} {Rating}");
        }
        public Zachet(string studentFio, string teacherFio, string date, int
rating)
        {
            StudentFio = studentFio;
            TeacherFio = teacherFio;
            Date = date;
            Rating = rating;
        }
    }
    class Addres
    {
        public int Index { get; set; }
        public string City { get; set; }
        public string Square { get; set; }
        public int NumberOfHouse { get; set; }
        public int Flat { get; set; }
        public override string ToString()
        {
            return ("Адрес: {Index} {City} {Square} {NumberOfHouse}
{Flat}");
        }
        public Adres(int index, string city, string square, int
numberOfHouse, int flat)
        {
            Index = index;
            City = city;
            Square = square;

```

```

        NumberOfHouse = numberOfHouse;
        Flat = flat;
    }
}
class Product
{
    public string Name { get; set; }
    public int Number { get; set; }
    public double Price { get; set; }
    public string ShelfLife { get; set; } // срок годности
    public override string ToString()
    {
        return ("Товар: {Name} {Number} {Price} {ShelfLife}");
    }
    public Product(string name, int number, double price, string
shelfLife)
    {
        Name = name;
        Number = number;
        Price = price;
        ShelfLife = shelfLife;
    }
}
class Group
{
    public string Shifre { get; set; } // шифр специальности
    public int NumberOfStudents { get; set; } // кол-во студентов
    public int YearOfFormation { get; set; } // год формирования
    public string Speciality { get; set; } // специальность
    public override string ToString()
    {
        return ("Группа: {Shifre} {NumberOfStudents} {YearOfFormation}
{Speciality}");
    }
    public Group(string shifre, int numberOfStudents, int
yearOfFormation, string speciality)
    {
        Shifre = shifre;
        NumberOfStudents = numberOfStudents;
        YearOfFormation = yearOfFormation;
        Speciality = speciality;
    }
}
class Banknote
{
    public int SerieesAndNumber { get; set; } // серия и номер
    public int Rating { get; set; } // номинал
    public string RatingOfString { get; set; } // номинал прописью
    public override string ToString()
    {
        return ("Купюра: {SerieesAndNumber} {Rating}
{RatingOfString}");
    }
    public Banknote(int seriesAndNumber, int rating, string
ratingOfString)
    {
        SerieesAndNumber = seriesAndNumber;
        Rating = rating;
        RatingOfString = ratingOfString;
    }
}

```

```

    }
}
class ComputerGame
{
    public string Name { get; set; }
    public string DevelopersTeamName { get; set; } // фирма разработчик
    public int YearName { get; set; }
    public string JanreName { get; set; }
    public override string ToString()
    {
        return ("Компьютерная игра: {Name} {DevelopersTeamName}
{YearName} {JanreName}");
    }
    public ComputerGame(string name, string developersTeamName, int
yearName, string janreName)
    {
        Name = name;
        DevelopersTeamName = developersTeamName;
        YearName = yearName;
        JanreName = janreName;
    }
}
class Program

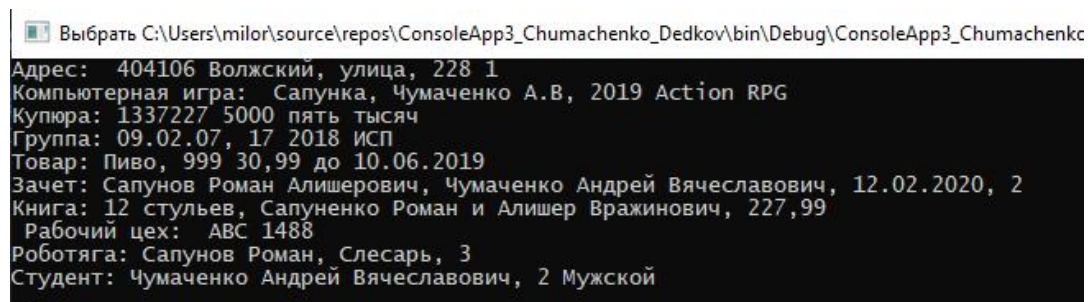
static void Main(string[] args)
{
    Adres adres01 = new Adres(404106, "Волжский", "улица", 228, 1);
    ComputerGame computerGame01 = new ComputerGame("Сапунка", "Чумаченко
А.В.", 2019, "Action RPG");
    Banknote banknote01 = new Banknote(1337227, 5000, "пять тысяч");
    Group group01 = new Group("09.02.07", 17, 2018, "ИСП");
    Product product01 = new Product("Пиво", 999, 30.99, "до 10.06.2019");
    za4et zachet01 = new za4et("Сапунов Роман Алишерович", "Чумаченко
Андрей Вячеславович", "12.02.2020", 2);
    Book book01 = new Book("12 стульев", "Сапуненко Роман и Алишер
Вражинович", 227.99);
    WorkShop workShop01 = new WorkShop("ABC", 1488);
    Robot9ga worker01 = new Robot9ga("Сапунов Роман", "Слесарь", 3);
    Student student01 = new Student("Чумаченко Андрей Вячеславович", 2,
"Мужской");

    Console.WriteLine($"{adres01}\n{computerGame01}\n{banknote01}\n{group01}\n{prod
uct01}\n{zachet01}\n{book01}\n{workShop01}\n{worker01}\n{student01}");
    Console.ReadKey();
}

}
}

```

Скриншот с результатом работы программы:



```
Выбрать C:\Users\milor\source\repos\ConsoleApp3_Chumachenko_Dedkov\bin\Debug\ConsoleApp3_Chumachenko
Адрес: 404106 Волжский, улица, 228 1
Компьютерная игра: Сапунка, Чумаченко А.В, 2019 Action RPG
Купюра: 1337227 5000 пять тысяч
Группа: 09.02.07, 17 2018 ИСП
Товар: Пиво, 999 30,99 до 10.06.2019
Зачет: Сапунов Роман Алишерович, Чумаченко Андрей Вячеславович, 12.02.2020, 2
Книга: 12 стульев, Сапуненко Роман и Алишер Вражинович, 227,99
Рабочий цех: ABC 1488
Роботяга: Сапунов Роман, Слесарь, 3
Студент: Чумаченко Андрей Вячеславович, 2 Мужской
```

Рисунок 1.1 - результат решения задачи 1

Задание 2

Создайте в Visual Studio проект консольного типа.
Определите в проекте класс, описывающий вектор в трёхмерном пространстве.

Класс должен включать:

- ✓ конструктор с параметрами в виде списка координат x, y, z
- ✓ метод, вычисляющий длину вектора по формуле:
$$\sqrt{x^2 + y^2 + z^2}$$
- ✓ метод, вычисляющий скалярное произведение этого вектора с другим по формуле:
$$x_1 x_2 + y_1 y_2 + z_1 z_2$$
- ✓ метод, вычисляющий векторное произведение с другим вектором:
$$(y_1 z_2 - z_1 y_2, z_1 x_2 - x_1 z_2, x_1 y_2 - y_1 x_2)$$
- ✓ Переопределение метода ToString(), возвращающий строковое представление вектора в формате: **vec (x:число ; y:число ; z:число)**

В методе Main() создайте два экземпляра класса, описывающего вектор. Используя конструктор задайте для каждого экземпляра, его каждому полю/свойству значения. Используя метод ToString() получите строковое представление каждого объекта. Выведите в консоль полученные строковые представления объектов.

Рассчитайте модули (длины) векторов и выведите в консоль результат с текстовым сообщением, поясняющим, что это длины векторов.

Рассчитайте скалярное произведение векторов и выведите в консоль результат с текстовым сообщением, поясняющим, что это результат скалярного произведения векторов.

Рассчитайте векторное произведение векторов и выведите в консоль результат с текстовым сообщением, поясняющим, что это результат векторного произведения векторов.

Листинг программы:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Chumachhenko
{
    class Vector
    {
        public double X { get; set; }
        public double Y { get; set; }
        public double Z { get; set; }

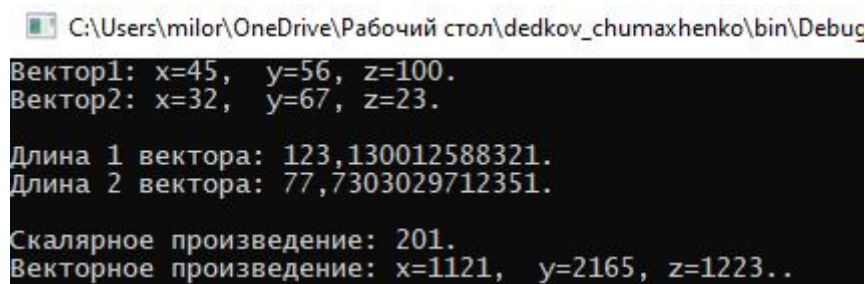
        public Vector()
            : this(0.0, 0.0, 0.0)
        {
        }
        public Vector(double x, double y, double z)
        {
            X = x;
            Y = y;
            Z = z;
        }
        public double Length()
        {
            return Math.Sqrt(Math.Pow(X, 2) + Math.Pow(Y, 2) + Math.Pow(Z, 2));
        }
        public double SkalarComposotion(Vector vec01)
        {
            return (vec01.X + vec01.Y + vec01.Z);
        }
        public double SkalarComposotion(double xVec, double yVec, double zVec)
        {
            return X * xVec + Y * yVec + Z * zVec;
        }
        public Vector Composotion(Vector vec01)
        {
            Vector res = new Vector();
            res.X = Y * vec01.Z - Z * vec01.Y;
            res.Y = Z * vec01.X - X * vec01.Z;
            res.Z = X * vec01.Y - Y * vec01.X;
            return res;
        }
        public Vector Composition(double xVec, double yVec, double zVec)
```

```

{
    Vector res = new Vector();
    res.X = Y * zVec - Z * yVec;
    res.Y = Z * xVec - X * zVec;
    res.Z = X * yVec - Y * xVec;
    return res;
}
public override string ToString()
{
    return ("x={X}, y={Y}, z={Z}.");
}
}
class Program
{
    static void Main(string[] args)
    {
        Vector vec01 = new Vector(45, 56, 100);
        Vector vec02 = new Vector(32, 67, 23);
        Console.WriteLine($"Вектор1: {vec01.ToString()}");
        Console.WriteLine($"Вектор2: {vec02.ToString()}");
        Console.WriteLine();
        Console.WriteLine($"Длина 1 вектора: {vec01.Length()}");
        Console.WriteLine($"Длина 2 вектора: {vec02.Length()}");
        Console.WriteLine();
        Console.WriteLine($"Скалярное произведение:
{vec01.SkalComposotion(vec01)}.");
        Console.WriteLine($"Векторное произведение:
{vec01.Composotion(vec02)}.");
        Console.ReadKey();
    }
}

```

Скриншот с результатом работы программы:



```

C:\Users\milor\OneDrive\Рабочий стол\dedkov_chumaxhenko\bin\Debug
Вектор1: x=45, y=56, z=100.
Вектор2: x=32, y=67, z=23.

Длина 1 вектора: 123,130012588321.
Длина 2 вектора: 77,7303029712351.

Скалярное произведение: 201.
Векторное произведение: x=1121, y=2165, z=1223..

```

Рисунок 1.1 - результат решения задачи 1