**Самостоятельная работа 1**

Тема: Определение классов

**Задание 1**

Определите в проекте ряд классов, перечисленных в задании.

Листинг Address.cs :

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ConsoleApp1

{

class Address

{

public int Zip { get; set; }

public string City { get; set; }

public string Street { get; set; }

public int Home { get; set; }

public int ApartmentNumber { get; set; }

public Address(int zip, string city, string street, int home, int apartmentNumber)

{

Zip = zip;

City = city;

Street = street;

Home = home;

ApartmentNumber = apartmentNumber;

}

public override string ToString()

{

return string.Format($"ZIP: {Zip}, город: {City}, улица: {Street}, дом: {Home}, номер квартиры: {ApartmentNumber}");

}

}

}

Листинг Book.cs :

using System;

namespace ConsoleApp1

{

class Book

{

public string Title { get; set; }

public string[] Authors { get; set; }

public double Price { get; set; }

public Book(string title, double price, params string[] authors)

{

Title = title;

Authors = authors;

Price = price;

}

public override string ToString()

{

if (Authors == null)

{

Console.WriteLine("Неизвестный автор");

}

return string.Format($"Название: {Title}, цена: {Price}, авторы: {string.Join(", ", Authors)}");

}

}

}

Листинг ComputerGame.cs :

using System;

namespace ConsoleApp1

{

class ComputerGame

{

public string Name { get; set; }

public string DeveloperFirm { get; set; }

public DateTime PublicationDate { get; set; }

public string Genre { get; set; }

public ComputerGame(string name, string developerFirm, string genre, DateTime publicationDate)

{

Name = name;

DeveloperFirm = developerFirm;

Genre = genre;

PublicationDate = publicationDate;

}

public override string ToString()

{

return string.Format($"Название: {Name}, разработчик: {DeveloperFirm}, дата релиза: {PublicationDate}, жанр: {Genre}");

}

}

}

Листинг Employee.cs :

using System;

namespace ConsoleApp1

{

class Employee

{

public string Name { get; set; }

public string Profession { get; set; }

public int WorkingExperience { get; set; }

public Employee(string name, string profession, int workingExperience)

{

Name = name;

Profession = profession;

WorkingExperience = workingExperience;

}

public override string ToString()

{

return string.Format($"Имя: {Name}, пол: {Profession}, рабочий стаж: {WorkingExperience}");

}

}

}

Листинг MonetaryDenomination.cs :

using System;

namespace ConsoleApp1

{

class MonetaryDenomination

{

public string SeriesAndNumber { get; set; }

public int Naminal { get; set; }

public string NaminalString { get; set; }

public MonetaryDenomination(string seriesAndNumber, int naminal, string naminalString)

{

SeriesAndNumber = seriesAndNumber;

Naminal = naminal;

NaminalString = naminalString;

}

public override string ToString()

{

return string.Format($"Серия и номер: {SeriesAndNumber}, наминал: {Naminal}, наминал прописью: {NaminalString}");

}

}

}

Листинг Product.cs :

using System;

namespace ConsoleApp1

{

class Product

{

public string Name { get; set; }

public int Quantity { get; set; }

public double Price { get; set; }

public DateTime ExpirationDate { get; set; }

public Product(string name, int quantity, double price, DateTime expirationDate)

{

Name = name;

Quantity = quantity;

Price = price;

ExpirationDate = expirationDate;

}

public override string ToString()

{

return string.Format($"Наименование: {Name}, количество: {Quantity}, цена: {Price}, срок годности: {ExpirationDate} дней");

}

}

}

Листинг Student.cs :

using System;

namespace ConsoleApp1

{

class Student

{

public string Name { get; set; }

public int Course { get; set; }

public readonly string Gender;

public Student(string name, int course, string gender)

{

Name = name;

Course = course;

Gender = gender;

}

public override string ToString()

{

return string.Format($"Имя: {Name}, пол: {Gender}, курс: {Course}");

}

}

}

Листинг Test.cs :

using System;

namespace ConsoleApp1

{

class Test

{

public string Examinator { get; set; }

public string Examiner { get; set; }

public DateTime Date { get; set; }

public int Estimation { get; set; }

public Test(string examinator, string examiner, DateTime date, int estimation)

{

Examinator = examinator;

Examiner = examiner;

Date = date;

Estimation = estimation;

}

public override string ToString()

{

return string.Format($"ФИО экзаменуемого: {Examinator}, ФИО экзаменатора: {Examiner}, дата: {Date}, оценка: {Estimation}");

}

}

}

Листинг TrainingGroup.cs :

using System;

namespace ConsoleApp1

{

class TrainingGroup

{

public string Сipher { get; set; }

public int NumberOfStudents { get; set; }

public int YearOfFormation { get; set; }

public string Specialty { get; set; }

public TrainingGroup(string cipher, int numberOfStudents, int yearOfFormation, string specialty)

{

Сipher = cipher;

NumberOfStudents = numberOfStudents;

YearOfFormation = yearOfFormation;

Specialty = specialty;

}

public override string ToString()

{

return string.Format($"Шифр: {Сipher}, количество студентов: {NumberOfStudents}, год формирования: {YearOfFormation}, специальность: {Specialty}");

}

}

}

Листинг Workshop.cs :

using System;

namespace ConsoleApp1

{

class Workshop

{

public string StringCipher { get; set; }

public int NumberOfEmployees { get; set; }

public Workshop(string stringCipher, int numberOfEmployees)

{

StringCipher = stringCipher;

NumberOfEmployees = numberOfEmployees;

}

public override string ToString()

{

return string.Format($"Строковый шифр: {StringCipher}, количество служащих: {NumberOfEmployees}");

}

}

}

Листинг Program.cs :

using System;

namespace ConsoleApp1

{

class Program

{

static void Main(string[] args)

{

Student student = new Student("Петр", 2, "Мужской");

Console.WriteLine(student);

Employee employee = new Employee("Иван", "Оффицер", 11);

Console.WriteLine(employee);

Workshop workshop = new Workshop("193QP39403", 45);

Console.WriteLine(workshop);

Book book = new Book("Колобок", 50.50, "Петр Петров Петрович", "Игорь Игоревичев Игоревич");

Console.WriteLine(book.ToString());

Test test = new Test("Сергей Сергевьич Сергеев", "Петр Петров Петрович", DateTime.Parse("2020.07.01"), 5);

Console.WriteLine(test);

Address address = new Address(54953, "Волжский", "Карла-Маркса", 23, 12);

Console.WriteLine(address);

Product product = new Product("Сок", 50, 25, DateTime.Parse ("2020.06.20"));

Console.WriteLine(product);

TrainingGroup trainingGroup = new TrainingGroup("295423", 20, 2018, "ATP-75");

Console.WriteLine(trainingGroup);

MonetaryDenomination monetaryDenomination = new MonetaryDenomination("HE 54237", 200, "Двести");

Console.WriteLine(monetaryDenomination);

ComputerGame computerGame = new ComputerGame("GTA 6", "RockstarGames", "Открытый мир", DateTime.Parse ("2020.12.10"));

Console.WriteLine(computerGame);

Console.ReadKey();

}

}

}

Скриншот с результатом программы :

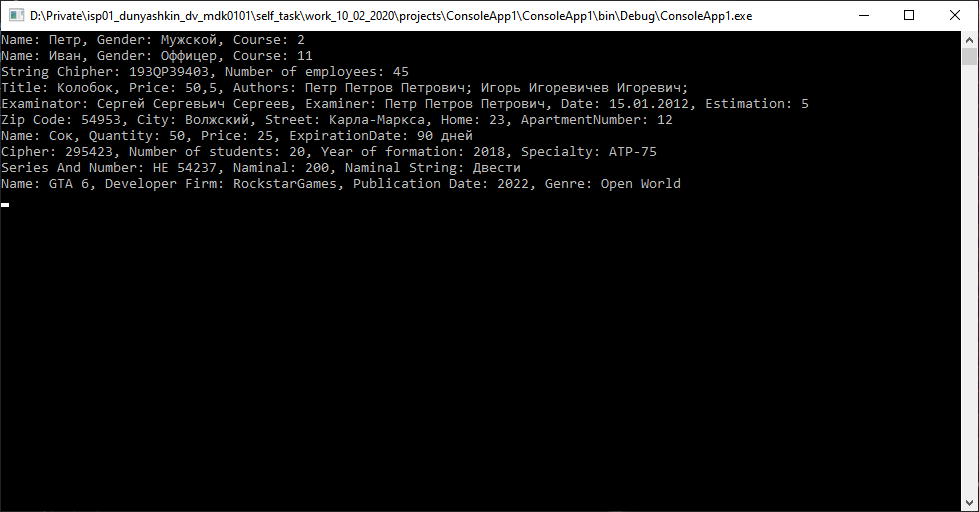


Рисунок 1 - результат решения задачи 1

**Задание 2**

Определите в проекте класс, описывающий вектор в трёхмерном пространстве.

Листинг Vector3D.cs :

using System;

namespace ConsoleApp2

{

class Vector3D

{

double X { get; set; }

double Y { get; set; }

double Z { get; set; }

public Vector3D(double x, double y, double z)

{

X = x;

Y = y;

Z = z;

}

public Vector3D VectorProduct(Vector3D other)

{

return new Vector3D(Y \* other.Z - Z \* other.Y,

Z \* other.X - X \* other.Z,

X \* other.Y - Y \* other.X);

}

public double GetLengt()

{

return Math.Sqrt(Math.Pow(X, 2) + Math.Pow(Y, 2) + Math.Pow(Z, 2));

}

public double ScalarProduct(Vector3D Vector)

{

return (X \* Vector.X + Y \* Vector.Y + Z \* Vector.Z);

}

public override string ToString()

{

return $"Vector (X: {X}, Y: {Y}, Z: {Z})";

}

}

}

Листинг Program.cs :

using System;

namespace ConsoleApp2

{

class Program

{

static void Main(string[] args)

{

Vector3D myVector = new Vector3D(35.50, 87.90, 35.10);

Vector3D otherVector = new Vector3D(50.50, 32.35, 12.70);

Console.WriteLine($"Вывод изначального вектора: {myVector.ToString()}");

Console.WriteLine($"Длина вектора: {myVector.GetLengt()}");

Console.WriteLine($"Cкалярное произведение: {myVector.ScalarProduct(otherVector)}");

Console.WriteLine($"Векторное произведение: {myVector.VectorProduct(otherVector)}");

Console.ReadKey();

}

}

}

Скриншот с результатом программы :

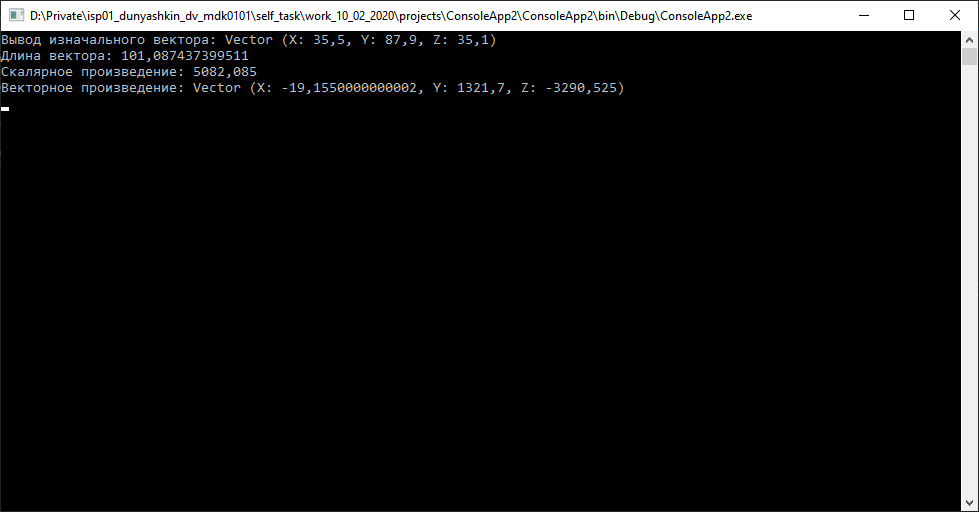


Рисунок 2 - результат решения задачи 2