**Самостоятельная работа 1**

**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).

1. Служащий

Свойства: Имя, Профессия, Рабочий стаж

1. Цех  
   Свойства: Строковый шифр, количество служащих
2. Книга

Свойства: Название (Заголовок), Список фио авторов, Стоимость

1. Зачёт  
   Свойства: ФИО экзаменуемого, ФИО экзаменатора, Дата, Оценка
2. Адрес  
   Свойства: Почтовый индекс, Город, Улица, Дом, квартира
3. Товар  
   Свойства: Наименование, Количество, Стоимость, Срок годности
4. Учебная группа  
   Свойства: Шифр, Количество студентов, Год формирования, Специальность
5. Денежная купюра  
   Свойства: Серия и номер, Наминал, Наминал прописью
6. Компьютерная игра  
   Свойства: Название, Фирма разработчик, Год издания, Жанр

Листинг Student.cs

using System;

namespace Classes

{

class Student

{

private int course;

public readonly string Gender;

public string Name { get; set;}

public int Course

{

get => course;

set

{

if (value < 1)

throw (new Exception("Enter invalid value"));

else

course = value;

}

}

public override string ToString()

{

string info = $"Course: {Course} " +

$"Name: {Name} " +

$"Gender: {Gender}";

return info;

}

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

{

Course = course;

Name = name;

Gender = gender;

}

}

}

Листинг Employee.cs

using System;

namespace Classes

{

class Employee

{

private double workStage;

public string Name { get; set; }

public string Profession { get; set; }

public double WorkStage

{

get => workStage;

set

{

if (value < 0)

throw (new Exception("Enter invalid value"));

else

workStage = value;

}

}

public override string ToString()

{

string info = $"Work Stage: {workStage} " +

$"Name: {Name} " +

$"Profession: {Profession}";

return info;

}

public Employee(string name, string profession, double workStage = 0)

{

WorkStage = workStage;

Name = name;

Profession = profession;

}

}

}

Листинг Workshop.cs

using System;

namespace Classes

{

class Workshop

{

public readonly string Id;

private int numberOfWorkers;

public int NumberOfWorkers

{

get => numberOfWorkers;

set

{

if (value < 0)

throw (new Exception("Enter invalid value"));

else

numberOfWorkers = value;

}

}

public override string ToString()

{

string info = $"ID: {Id} " +

$"Number of Workers : {NumberOfWorkers} ";

return info;

}

public Workshop(string id, int numberOfWorkers = 0)

{

Id = id;

NumberOfWorkers = numberOfWorkers;

}

}

}

Листинг Book.cs

using System;

namespace Classes

{

class Book

{

private double price;

public string Title { get; set; }

public string[] Authors { get; set; }

public double Price

{

get => price;

set

{

if (value < 0)

throw (new Exception("Enter invalid value"));

else

price = value;

}

}

public override string ToString()

{

string info = $"Title: {Title} " +

$"Price: {price} " +

"Authors: ";

foreach (string author in Authors)

info += author;

return info;

}

public Book(string[] authors, double price = 0, string title = "Noname")

{

Authors = authors;

Price = price;

Title = title;

}

}

}

Листинг Credit.cs

using System;

namespace Classes

{

class Credit

{

private int grade;

public string FIOExaminee { get; set; }

public string FIOExaminer { get; set; }

public DateTime Date { get; set; }

public int Grade

{

get => grade;

set

{

if (value < 0 || value > 5)

throw (new Exception("Enter invalid value"));

else

grade = value;

}

}

public override string ToString()

{

string info = $"Full name Examiner's: {FIOExaminer} " +

$"Full name Examinee's: {FIOExaminee} " +

$"Date: {Date} " +

$"Grade {Grade}";

return info;

}

public Credit(DateTime date, int grade, string fioExaminee, string fioExaminer)

{

Grade = grade;

FIOExaminee = fioExaminee;

FIOExaminer = fioExaminer;

Date = date;

}

public Credit(string fioExaminer, int grade, string fioExaminee = "Noname")

:this (DateTime.Now, grade, fioExaminee, fioExaminer)

{

}

}

}

Листинг Address.cs

using System;

namespace Classes

{

class Address

{

public int MailIndex { get; set; }

public string City { get; set; }

public string Street { get; set; }

public string Home { get; set; }

public int Flat { get; set; }

public override string ToString()

{

string info = $"Mail index: {MailIndex} " +

$"City: {City} " +

$"Street: {Street} " +

$"Home: {Home} " +

$"Flat: {Flat}";

return info;

}

public Address(int mailIndex, string city, string street, string home, int flat)

{

MailIndex = mailIndex;

City = city;

Street = street;

Home = home;

Flat = flat;

}

}

}

Листинг Product.cs

using System;

namespace Classes

{

class Product

{

public string Title { get; set; }

public int Amount { get; set; }

public double Price { get; set; }

public double StorageLife { get; set; }

public override string ToString()

{

string info = $"Prodeuct's title: {Title} " +

$"Amount of products: {Amount} " +

$"Price: {Price} " +

$"Storage life: {StorageLife} ";

return info;

}

public Product(int amount = 0, string title = "Noname", double price = 0, double storageLife = 0)

{

Title = title;

Amount = amount;

Price = price;

StorageLife = storageLife;

}

}

}

Листинг EducationGroup.cs

using System;

namespace Classes

{

class EducationalGroup

{

private int amountStudents;

private int yearFormation;

public string Id { get; set; }

public int AmountStudents

{

get => amountStudents;

set

{

if (value < 0)

throw (new Exception("Enter invalid value"));

else

amountStudents = value;

}

}

public int YearFormation

{

get => yearFormation;

set

{

if (value < 1)

throw (new Exception("Enter invalid value"));

else

yearFormation = value;

}

}

public string Specialty { get; set; }

public override string ToString()

{

string info = $"ID: {Id} " +

$"Specialty: {Specialty} " +

$"Amount: {AmountStudents} " +

$"Year Formation: {YearFormation} ";

return info;

}

public EducationalGroup(string id, int yearFormation, int amountStudents = 0, string specialty = "Indefined specialty")

{

AmountStudents = amountStudents;

YearFormation = yearFormation;

Id = id;

Specialty = specialty;

}

}

}

Листинг MonetaryNominal.cs

using System;

namespace Classes

{

class MonetaryNominal

{

public string SerialAndNumber { get; set; }

public int Nominal { get; set; }

public string NominalInWords { get; set; }

public override string ToString()

{

string info = $"Serial And Number: {SerialAndNumber} " +

$"Nominal: {Nominal} " +

$"Nominal value in words: {NominalInWords} ";

return info;

}

public MonetaryNominal(string serialAndNumber, int nominal, string nominalInWords)

{

SerialAndNumber = serialAndNumber;

Nominal = nominal;

NominalInWords = nominalInWords;

}

}

}

Листинг ComputerGame.cs

using System;

namespace Classes

{

class ComputerGame

{

public string Title { get; set; }

public string DeveloperFirm { get; set; }

public int YearPublication { get; set; }

public string Genre { get; set; }

public override string ToString()

{

string info = $"Title: {Title} " +

$"Developer firm: {DeveloperFirm} " +

$"Year publication: {YearPublication} " +

$"Genre: {Genre}";

return info;

}

public ComputerGame(string developerfirm, string genre, int yearPublication, string title = "Untitle Game")

{

Title = title;

DeveloperFirm = developerfirm;

Genre = genre;

YearPublication = yearPublication;

}

}

}

Листинг Program.cs

using System;

namespace Classes

{

class Program

{

static void Main(string[] args)

{

Student student01 = new Student("Male", 1, "Иванов И.И.");

Employee employee01 = new Employee("Петров П.П.", "Слесарь", 1);

Workshop workshop01 = new Workshop("WS01", 5);

string[] authors = { "Аркадий Стругацкий", "Борис Стругацкий" };

Book book01 = new Book(authors, 2002, "Пикник на обочине");

Credit credit01 = new Credit("Васильев В.В.", 5, "Сидоров К.А.");

Address address01 = new Address(445501, "Волгоград", "Пушкина", "Колотушкина", 3);

Product product01 = new Product(50, "Бомбастер", 5, 1000);

EducationalGroup group01 = new EducationalGroup(19, "1-18", 2018, "ИСП");

MonetaryNominal monetary01 = new MonetaryNominal("qwerty 12345", 10, "десять рублей");

ComputerGame game01 = new ComputerGame("Dev01", "Action", 2000, "Untitle Game");

Console.WriteLine("Student: " + student01.ToString() + "\n");

Console.WriteLine("Employee: " + employee01.ToString() + "\n");

Console.WriteLine("Workshop: " + workshop01.ToString() + "\n");

Console.WriteLine("Book: " + book01.ToString() + "\n");

Console.WriteLine("Credit: " + credit01.ToString() + "\n");

Console.WriteLine("Address: " + address01.ToString() + "\n");

Console.WriteLine("Product: " + product01.ToString() + "\n");

Console.WriteLine("Educational Group: " + group01.ToString() + "\n");

Console.WriteLine("Monetary Denomination: " + monetary01.ToString() + "\n");

Console.WriteLine("Computer Game: " + game01.ToString() + "\n");

Console.ReadKey();

}

}

}