Автор: Сиромятников Марк КІТ-119а

**Лабораторна робота 1**

**Тема**. Розробка програм на С#. Консольні програми

Задачі:

1. Введення та збереження анкетних даних особистої справи студента вузу: П.І.Б., дата народження, дата надходження, індекс академ. групи (а, б...), фт, спеціальність, успішність (у%). Забезпечити валідацію даних, що вводяться.
2. Доступ до особистих полів реалізувати за допомогою властивостей класу, забезпечивши необхідні обчислення та перевірку коректності встановлюваних значень. Для спрощеного доступу використовувати властивості, що автоматично реалізуються.
3. Відображення введених даних.
4. Демонстрація роботи з масивами об'єктів.

Опис класів

Student – клас, який відображує студента;

Текст програми

Student.cs

using System;

namespace syromiatnikov01

{

/// <summary>

/// Class Program

/// class that creates array of students

/// and prints it in console

/// </summary>

class Program

{

static void Main(string[] args)

{

/\*Console.WriteLine("Enter student's surname: ");

string surname = Console.ReadLine();

Console.WriteLine("Enter student's name: ");

string name = Console.ReadLine();

Console.WriteLine("Enter student's patronymic: ");

string patronymic = Console.ReadLine();

Console.WriteLine("Enter student's date of birth: ");

string dateOfBirth = Console.ReadLine();

Console.WriteLine("Enter student's date of admission: ");

string dateOfAdmission = Console.ReadLine();

Console.WriteLine("Enter student's group index: ");

string groupIndex = Console.ReadLine();

Console.WriteLine("Enter student's faculty: ");

string faculty = Console.ReadLine();

Console.WriteLine("Enter student's specialty: ");

string specialty = Console.ReadLine();

Console.WriteLine("Enter student's academic performance: ");

string academicPerformance = Console.ReadLine();\*/

// Creating array of students

var students = new Student[] { new Student("Bily", "Vadim", "Ivanovich", DateTime.Parse("12-6-2001"), DateTime.Parse("16-05-2019"), "119a", "CIT", "123 - Computer engineering", 100),

new Student("Menshakov", "Dmytro", "Olegovich", DateTime.Parse("16-11-2000"), DateTime.Parse("23-8-2019"), "119a", "CIT", "123 - Computer engineering", 90)/\*,

new Student(name, surname, patronymic, DateTime.Parse(dateOfBirth), DateTime.Parse(dateOfAdmission), Convert.ToChar(groupIndex), faculty, specialty, Int32.Parse(academicPerformance))\*/};

// Printing out students' data

for (var i = 0; i < students.Length; i++)

{

Console.WriteLine(students[i].ToString());

}

Console.ReadLine();

}

}

}

Program.cs

using System;

using System.Runtime.Serialization;

namespace syromiatnikov01

{

/// <summary>

/// Class Student

/// class that models student

/// contains student's fields and properties

/// </summary>

[DataContract]

public class Student

{

/// <summary>

/// Private fields of a class

/// </summary>

private string \_firstName;

private string \_lastName;

private string \_patronymic;

private DateTime \_dateOfBirth;

private DateTime \_dateOfAdmission;

private string \_group;

private string \_faculty;

private string \_specialty;

private int \_academicPerformance;

/// <summary>

/// Constructor with 9 parameters

/// </summary>

/// <param name="lastName"></param>

/// <param name="firstName"></param>

/// <param name="patronymic"></param>

/// <param name="dateOfBirth"></param>

/// <param name="dateOfAdmission"></param>

/// <param name="group"></param>

/// <param name="faculty"></param>

/// <param name="specialty"></param>

/// <param name="academicPerformance"></param>

public Student(string lastName, string firstName, string patronymic, DateTime dateOfBirth, DateTime dateOfAdmission, string group,

string faculty, string specialty, int academicPerformance)

{

LastName = lastName;

FirstName = firstName;

Patronymic = patronymic;

DateOfBirth = dateOfBirth;

DateOfAdmission = dateOfAdmission;

Group = group;

Faculty = faculty;

Specialty = specialty;

AcademicPerformance = academicPerformance;

}

/// <summary>

/// Public property Name

/// </summary>

[DataMember]

public string FirstName

{

get

{

return \_firstName;

}

set

{

if (string.IsNullOrEmpty(value))

{

Console.WriteLine("You've entered wrong name\n");

}

else

{

\_firstName = value;

}

}

}

/// <summary>

/// Public property Surname

/// </summary>

[DataMember]

public string LastName

{

get

{

return \_lastName;

}

set

{

if (string.IsNullOrEmpty(value))

{

Console.WriteLine("You've entered wrong surname\n");

}

else

{

\_lastName = value;

}

}

}

/// <summary>

/// Public property Patronymic

/// </summary>

[DataMember]

public string Patronymic

{

get

{

return \_patronymic;

}

set

{

if (string.IsNullOrEmpty(value))

{

Console.WriteLine("You've entered wrong patronymic\n");

}

else

{

\_patronymic = value;

}

}

}

/// <summary>

/// Public property DateOfBirth

/// </summary>

[DataMember]

public DateTime DateOfBirth

{

get

{

return \_dateOfBirth;

}

set

{

if (value < new DateTime(2000, 1, 1) || value > DateTime.Today)

{

Console.WriteLine("You've entered wrong date of birth\n");

}

\_dateOfBirth = value;

}

}

/// <summary>

/// Public property DateOfAdmission

/// </summary>

[DataMember]

public DateTime DateOfAdmission

{

get

{

return \_dateOfAdmission;

}

set

{

if (value < new DateTime(2015, 1, 1) || value > DateTime.Today)

{

Console.WriteLine("You've entered wrong date of admission\n");

}

\_dateOfAdmission = value;

}

}

/// <summary>

/// Public property GroupIndex

/// </summary>

[DataMember]

public string Group

{

get

{

return \_group;

}

set

{

if (string.IsNullOrEmpty(value))

{

Console.WriteLine("You've entered wrong group\n");

}

else

{

\_group = value;

}

}

}

/// <summary>

/// Public property Faculty

/// </summary>

[DataMember]

public string Faculty

{

get

{

return \_faculty;

}

set

{

if (string.IsNullOrEmpty(value))

{

Console.WriteLine("You've entered wrong faculty\n");

}

else

{

\_faculty = value;

}

}

}

/// <summary>

/// Public property Specialty

/// </summary>

[DataMember]

public string Specialty

{

get

{

return \_specialty;

}

set

{

if (string.IsNullOrEmpty(value))

{

Console.WriteLine("You've entered wrong specialty\n");

}

else

{

\_specialty = value;

}

}

}

/// <summary>

/// Public property AcademicPerformance

/// </summary>

[DataMember]

public int AcademicPerformance

{

get

{

return \_academicPerformance;

}

set

{

if (\_academicPerformance < 0 || \_academicPerformance > 100)

{

Console.WriteLine("You've entered wrong academic performance\n");

}

\_academicPerformance = value;

}

}

/// <summary>

/// ToString method overriding

/// </summary>

/// <returns>Full data about student</returns>

public override string ToString()

{

return $"First name: {FirstName}\nLast name: {LastName}\nPatronymic: {Patronymic}\nDate of birth: {DateOfBirth}\nDate of admission: " +

$"{DateOfAdmission}\nGroup: {Group}\nFaculty: {Faculty}\nSpecialty: {Specialty}\nAcademic performance: {AcademicPerformance}%\n";

}

/// <summary>

/// Equals method overriding

/// </summary>

/// <param name="obj"></param>

/// <returns>If objects are equal returns true otherwise false</returns>

public override bool Equals(object obj)

{

var other = obj as Student;

return other != null && GetHashCode().Equals(other.GetHashCode());

}

/// <summary>

/// GetHashCode method overriding

/// </summary>

/// <returns>Hashcode of an object</returns>

public override int GetHashCode()

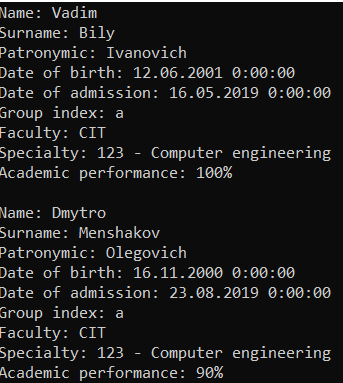
{

return (FirstName, LastName, Patronymic).GetHashCode();

}

}

}



Результати роботи програми

**Висновок:** у результаті виконання лабораторної роботи було створено клас Student, який відображує модель студента. Було створено масив таких об’єктів та виведено його у консоль.