**Звіт**

**До практичної роботи**

**з дисципліни «Основи інформаційної безпеки»**

Студента Київського Національного Університету ім. Тараса Шевченко

Переверзева Олексія Сергійовича

Факультет інформаційних технологій

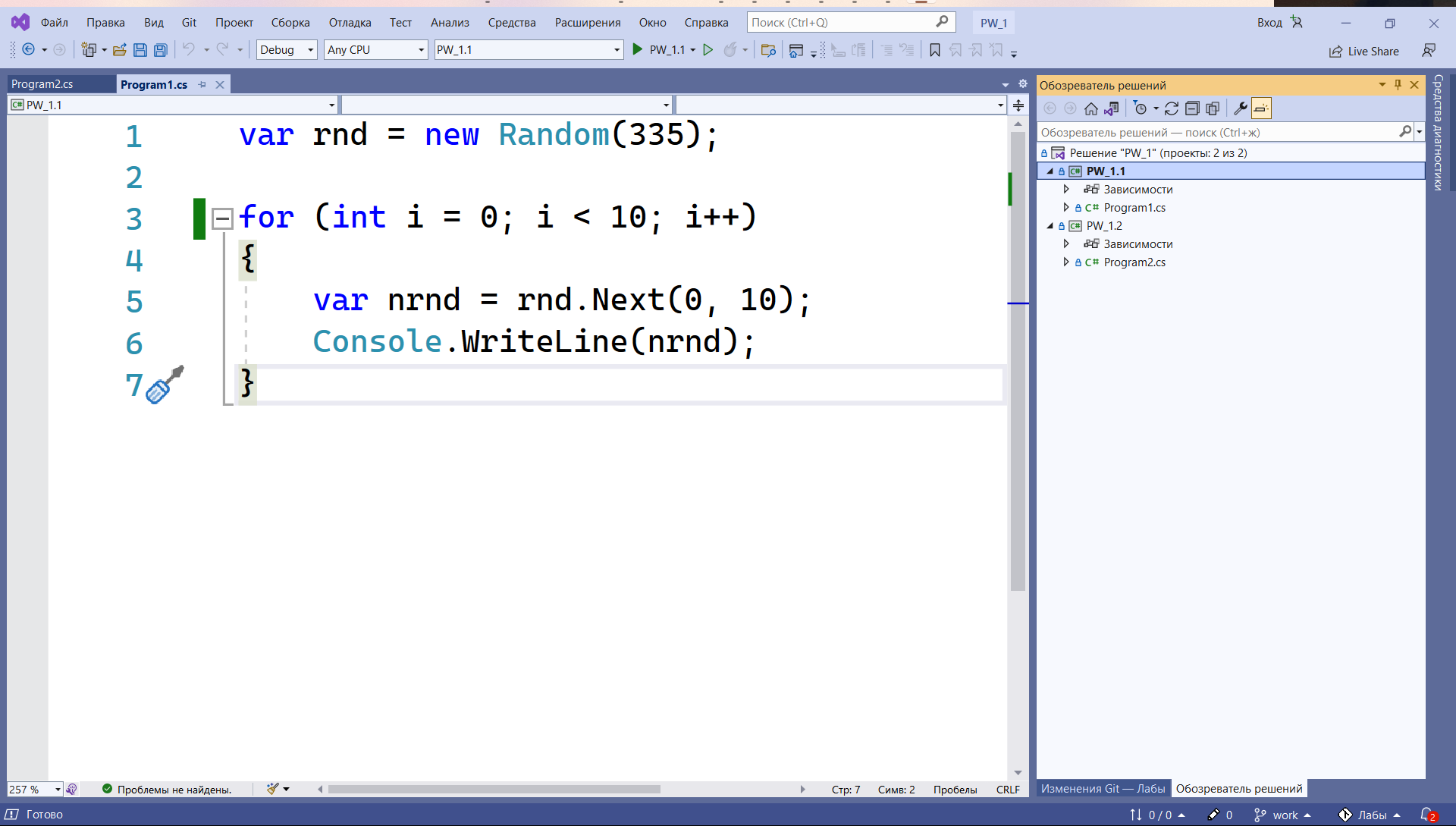
Група МІТ-21



**Практичне заняття №1**

**Тема:** Генерація послідовності випадкових чисел

1. Написати програму, яка забезпечує генерацію та виведення на екран послідовності псевдовипадкових чисел. Порівняти елементи послідовності для однакових та різних початкових значень.



Спочатку ініціюємо генератор з будь-яким початковим значенням:

var rnd = new Random(335);

Оголосили неявно типізовану змінну rnd, яка є seed для подальшого генерування псевдовипадкових чисел.

За допомогою циклу for в умові вказуємо кількість чисел, яку ми хочемо вивести на екран. Нехай це буде 10 чисел:

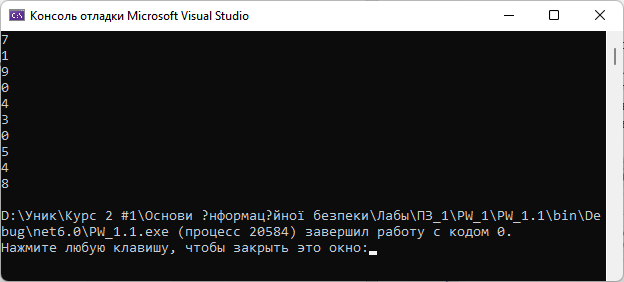
for (int i = 0; i < 10; i++)

Далі оголошуємо для зручності нову змінну, яка буде зберігати отримане псевдовипадкове число, та звертаємося до ініційованого об’єкта генератора для отримання значення:

var nrnd = rnd.Next(0, 10);

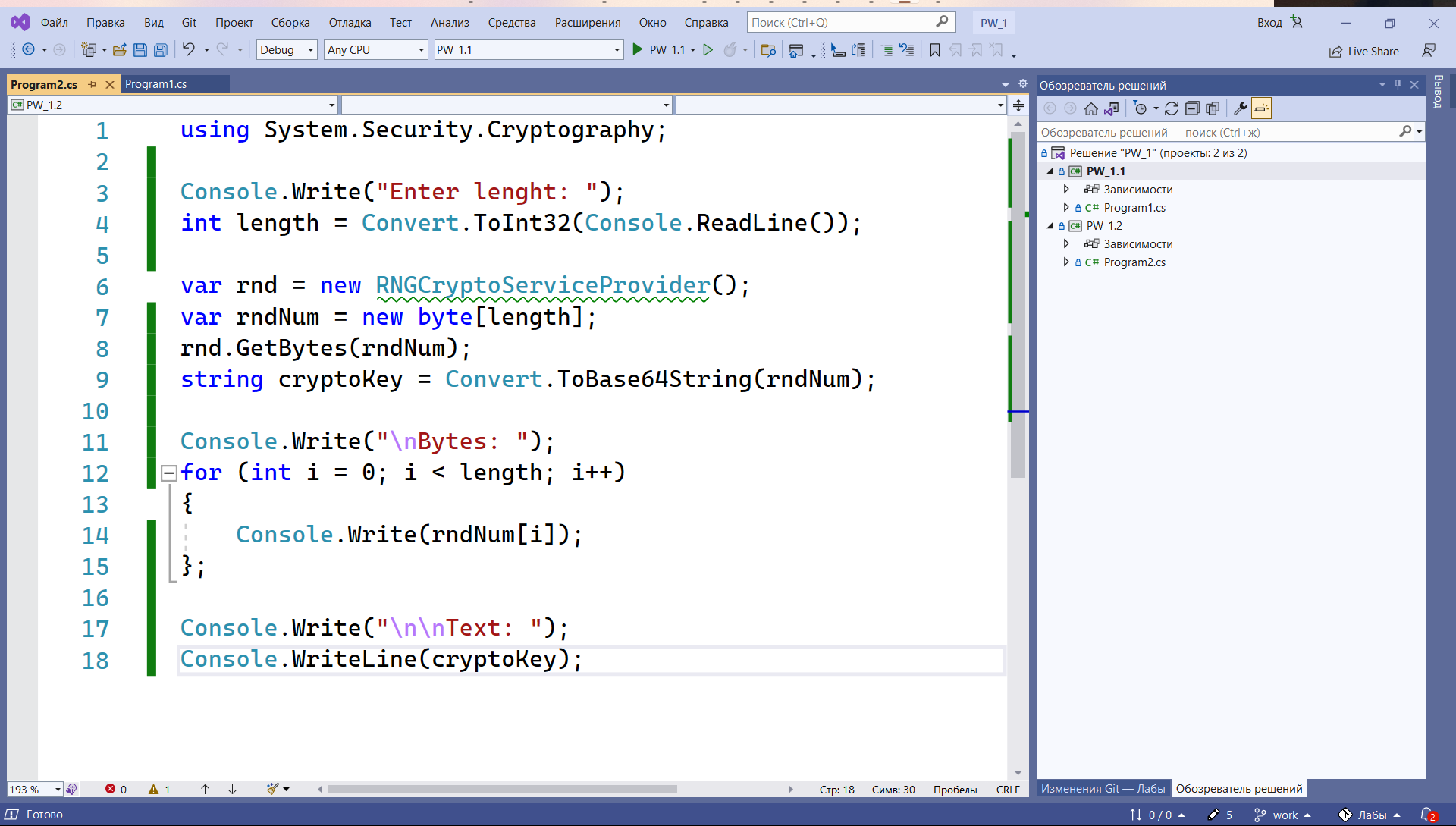
В дужках можна виставити інший діапазон, якщо потрібно.

Якщо не змінювати seed, то ми завжди будемо отримувати одні й ті самі числа.



Отже, можна стверджувати, що такий метод генерування не є досконалим, так як знаючи задане початкове значення, досить легко дізнатися значення послідновності.

1. Написати програму, яка забезпечує генерацію та виведення на екран криптографічно стійкої послідовності випадкових чисел. Порівняти елементи послідовності для декількох послідовних спроб.



Для цілей криптогафії підключаємо простір імен:

using System.Security.Cryptography;

Ініціюємо об’єкт, використовуючи потрібний клас:

var rnd = new RNGCryptoServiceProvider();

Також для зручності додав функціонал вводу з клавіатури довжини послідновності чисел:

Console.Write("Enter lenght: ");

int length = Convert.ToInt32(Console.ReadLine());

Після ініціалізації об’єкта цього класу можна отримати послідовність випадкових значень для використання у криптографії:

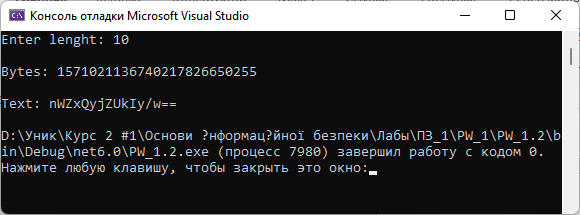
var rndNum = new byte[length];

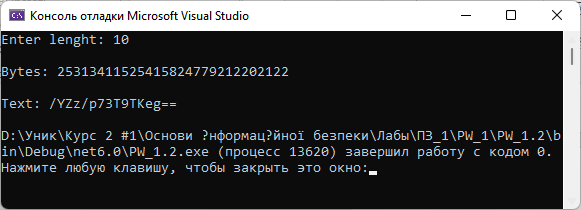
rnd.GetBytes(rndNum);

Також перетворимо послідовність байтів у текстове представлення:

string cryptoKey = Convert.ToBase64String(rndNum);

Спробуємо запустити програму декілька разів, вводячи однакову довжину





Бачимо, що ми отримали різні значення, а отже можна стверджувати, що цей метод є досить надійним для використовування у криптографії.

**Висновок:** в ході практичної роботи ми спробували різні методи створення псевдовипадкових чисел та дослідили надійність кожного з них.