Міністерство освіти і науки

Київський національний університет імені Тараса Григоровича Шевченка

Кафедра Мережевих та Інтернет технологій

**Звіт**

з практичної роботи **№1**

з дисципліни: “Основи інформаційної безпеки”

**«Генерація послідовності випадкових чисел»**

Виконав:

Студент групи МІТ-21

Горбун О. М.

**Результати виконання завдань роботи:**

За завданнями практичної роботи було створено програми, які можна переглянути за посиланням:

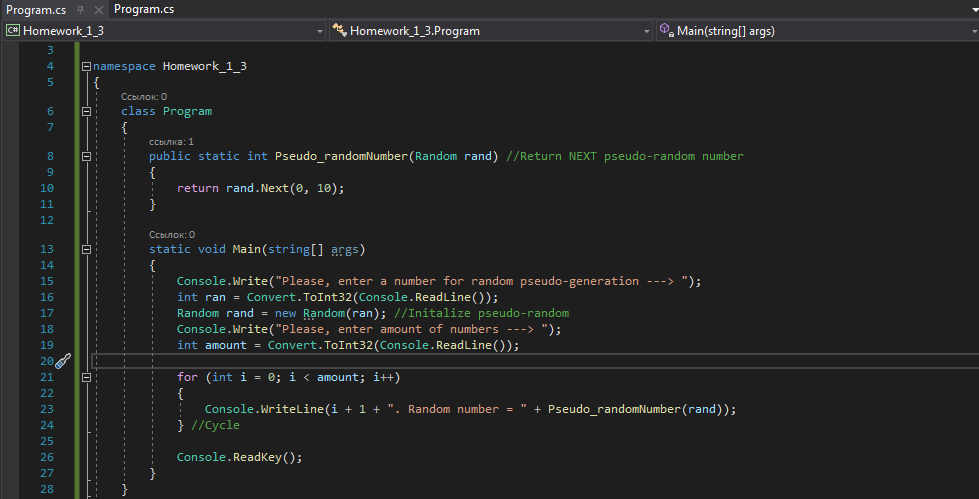
<https://github.com/OleksandrHorbun/Basics_Cybersecurity.git> (Посилання на відкритий репозиторій у веб-сервісі GitHub) у вітці (Branch) Main.

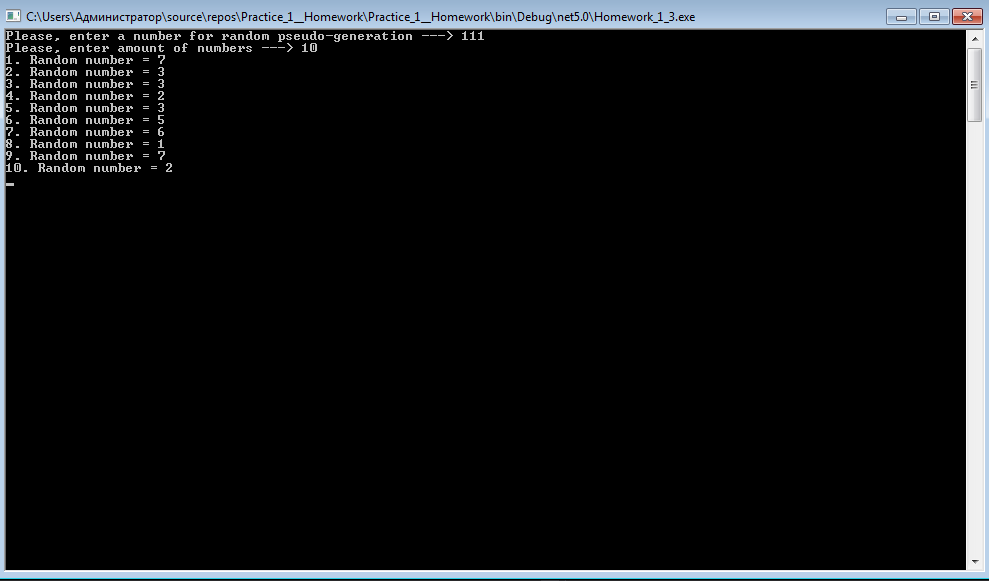
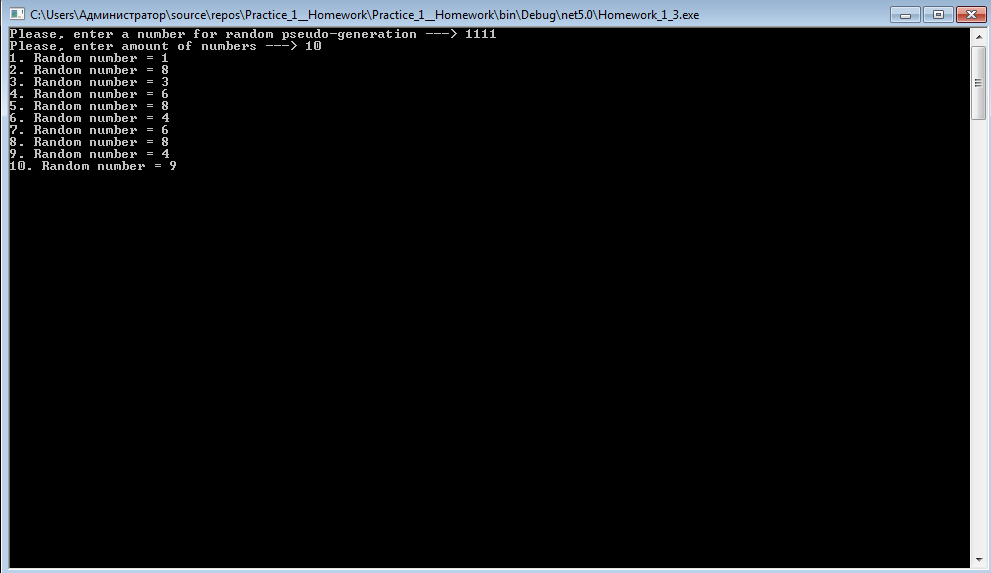
**I частина (третє завдання)**

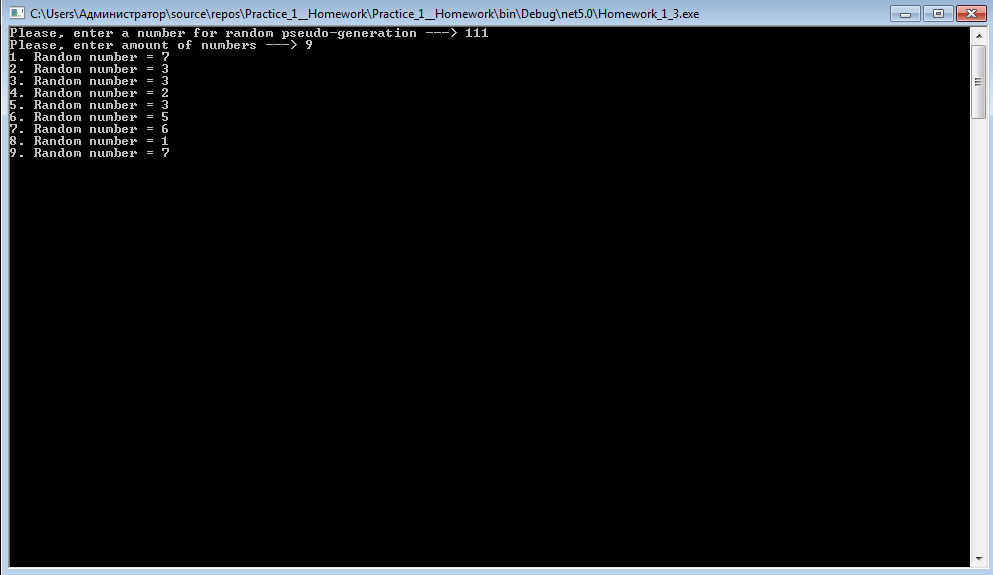
Як бачимо, програма вимагає введення «зерна» (Seed) генератора та кількість чисел, які будуть згенеровані. Ми ініціалізуємо генератор псевдо-рандомних чисел з «зерном», від нього залежить з яких саме псевдовипадкових чисел ми отримаємо послідовність. Після цього за допомогою циклу for та функції *Pseudo\_randomNumber* (просто повертає наступне псевдовипадкове число від 0 (включно) до 10 (не включно)) виводимо по одному числу на екран.

Отримані числа ми називаємо «псевдовипадковими», адже якщо ініціалізувати та вписувати однакове «зерно» генератору ми, як результат, отримаємо абсолютно ідентичну послідовність чисел. Очевидно що такий спосіб генерації у інформаційній безпеці не підійде тому, що увесь час у нас буде однакова послідовність чисел, і потенційний зловмисник підібравши необхідне «зерно» легко зможе їх отримати.

Все описане вище можна побачити на знімках екрану:





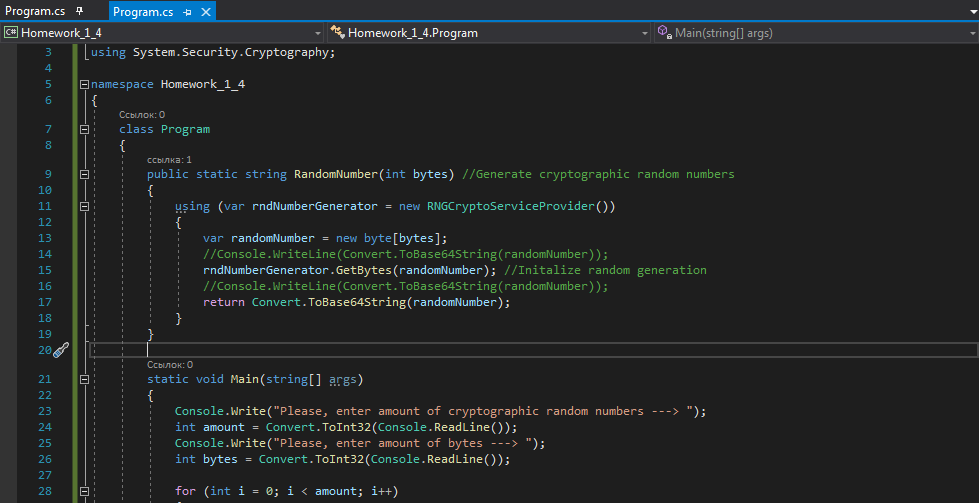


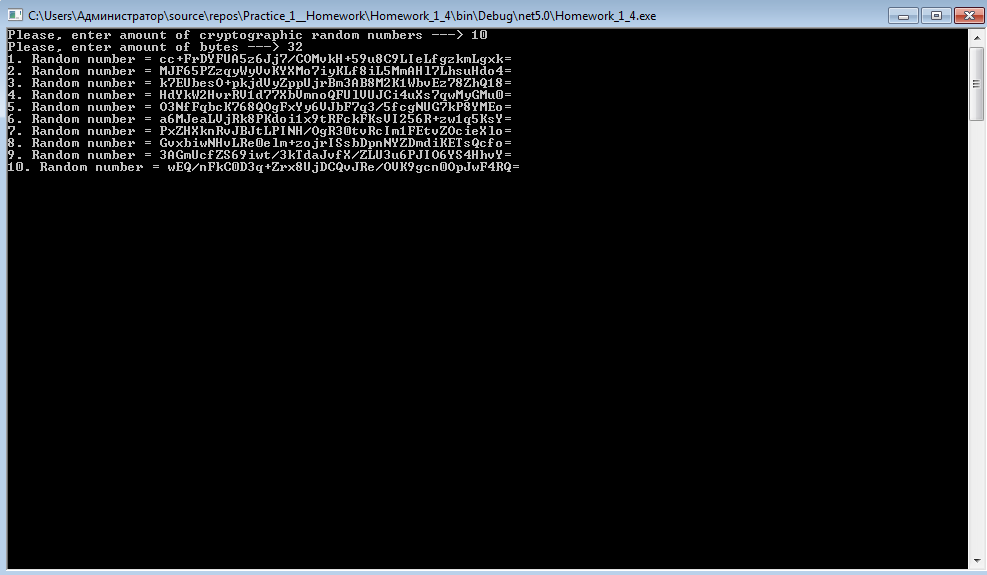
**IІ частина (третє завдання)**

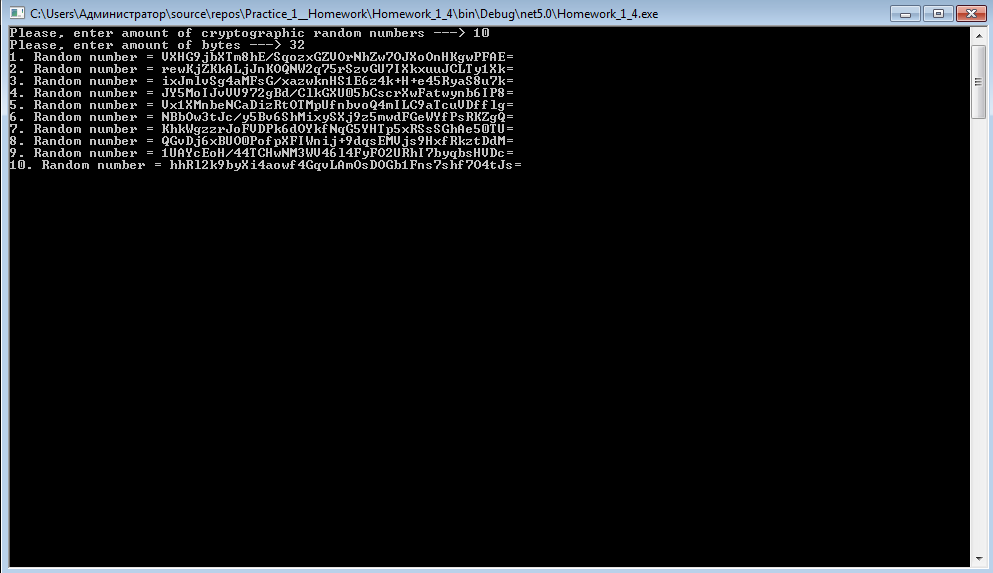
Друга програма теж потребує введення кількості випадкових чисел, які будуть згенеровані, та кількість байтів для кожного числа («довжина» числа у байтах; 1 байт = 8 біт). Після цього за допомогою циклу та функції *RandomNumber* (створює змінну, по суті, з порожніми бітами (ААААА), створює певну послідовність випадкових чисел з буквами та символами, кожного разу заповнює змінну новими значеннями, які одразу ж переводяться у рядок та повертає змінну у цикл). Усі отримані значення по одному, з нумерацією по порядку, виводяться на екран.

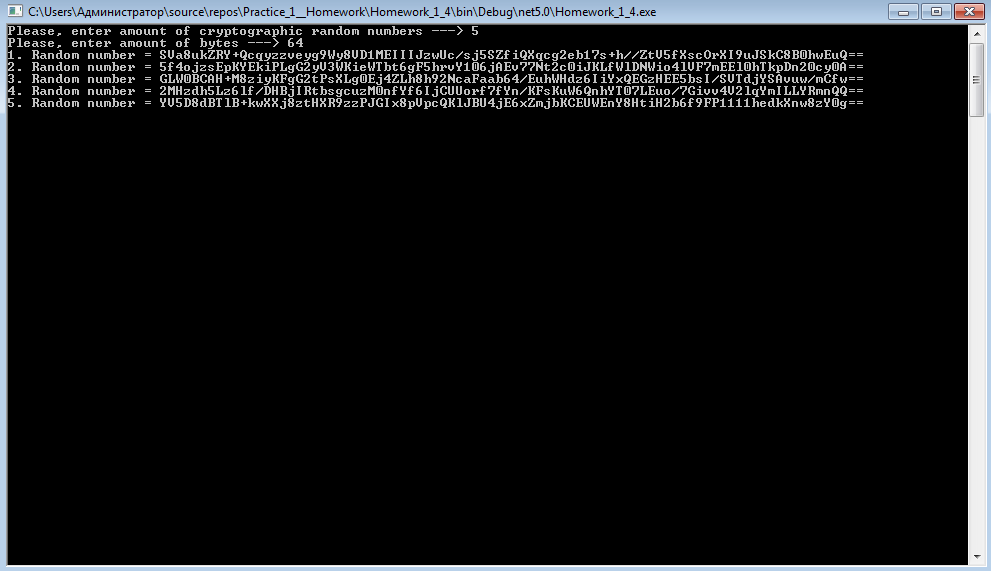
Якщо проаналізувати та порівняти цей спосіб з минулим, виявимо що так генерувати випадкові числа набагато краще, цей механізм можна вважати більш-менш надійним та використовувати в інформаційній безпеці для, наприклад, шифрування, адже при кожній новій компіляції програми ми отримаємо абсолютно різні значення, які непросто прочитати та не повторюються з попередніми. Крім того таким чином можна ще й легко змінювати «довжину» (кількість байтів і, відповідно, кількість символів).

Все описане вище можна побачити на знімках екрану:









**Висновок:**

Виконуючи цю роботу я згадав як працювати у середовищі програмування Visual Studio 2019 з мовою програмування C#, ознайомився з двома варіантами генерацій випадкових чисел, зрозумів як їх та який з них краще застосовувати у своїх подальших програмах.