Звіт

До практичного заняття №5

З теми: «Безпечне зберігання паролів»

Голомозик Кирило МІТ-21

1. Опрацювати розділ 5 літературного джерела Haunts Stephen. Applied Cryptography in .NET and Azure Key Vault.

2. Для кожного способу зберігання паролів, описаного у розділі 5 (Safely Storing Passwords), законспектувати основні переваги і недоліки. Зробити висновок про доцільність застосування даного способу зберігання паролів.

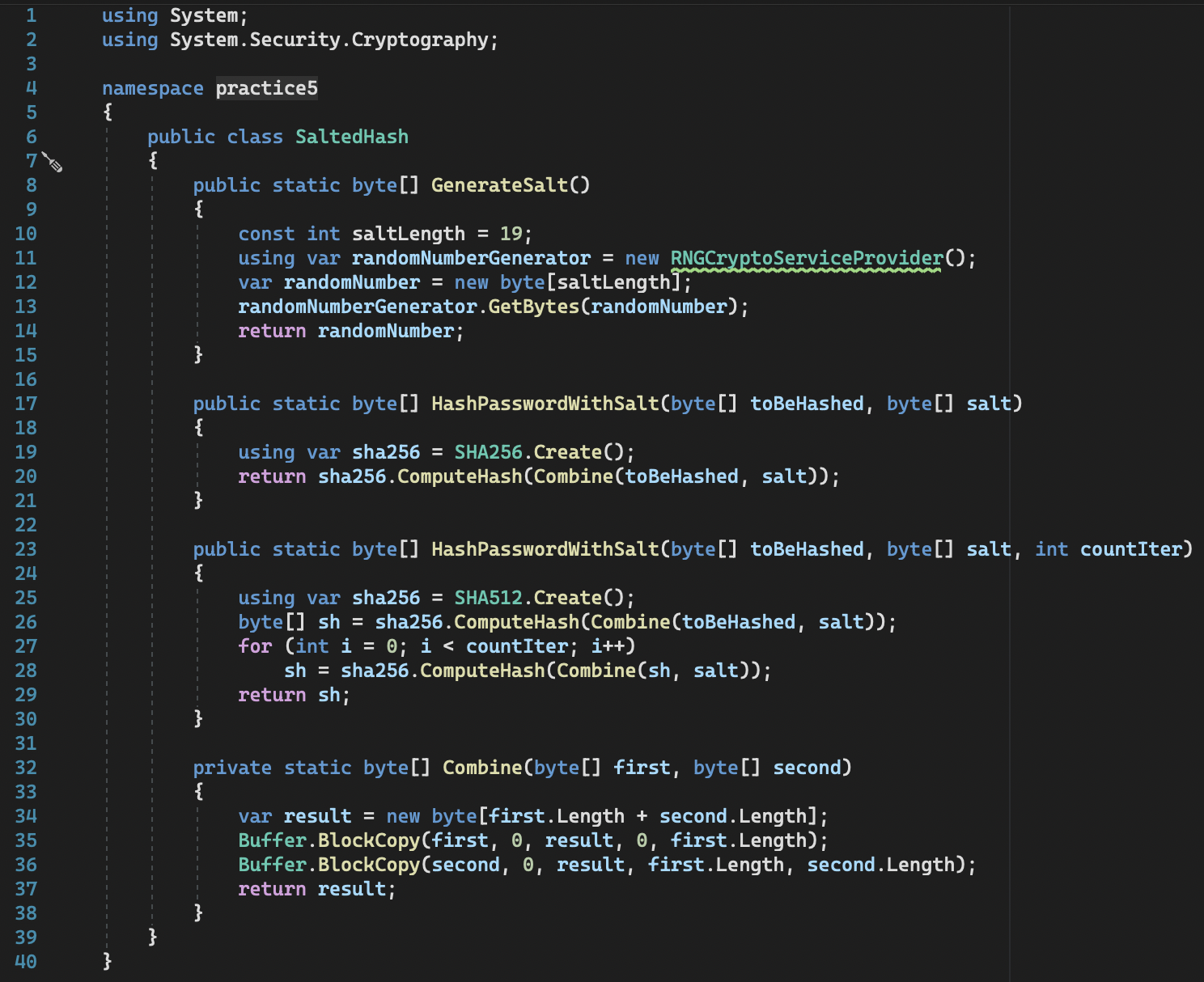
3. Розробити клас SaltedHash, що реалізує хешування паролів із додаванням додаткової ентропії. Продемонструвати роботу класу, обчислюючи хеш для заданого пароля та "солі".

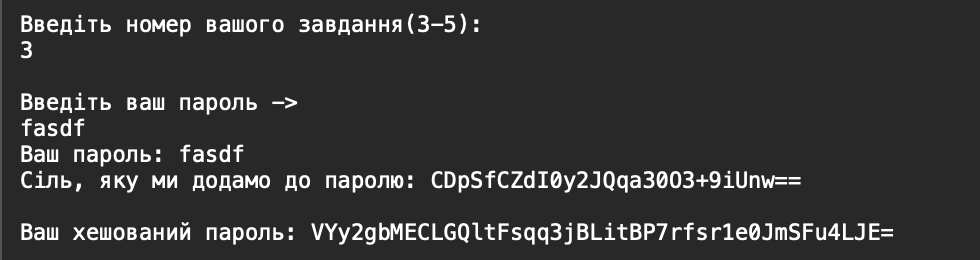
4. Розробити клас PBKDF2, що має наступну функціональність: генерує "сіль", задає алгоритм хешування (MD5, SHA1, SHA256, SHA384, SHA512) та обчислює хеш для заданого числа ітерацій. Створити програму, що обчислює час, витрачений на обчислення хешу для різного числа ітерацій (10 значень із кроком 50'000; перше значення = номер варіанта \* 10'000 ). Побудувати графік залежності витраченого часу від числа ітерацій.

5. Написати програму, що реалізує хешування введеного пароля під час реєстрації користувача та зберігає логін, пароль та "сіль" у пам'яті. Реалізувати можливість автентифікації за логіном і паролем. Число ітерацій = номер варіанта(8) \* 10'000.

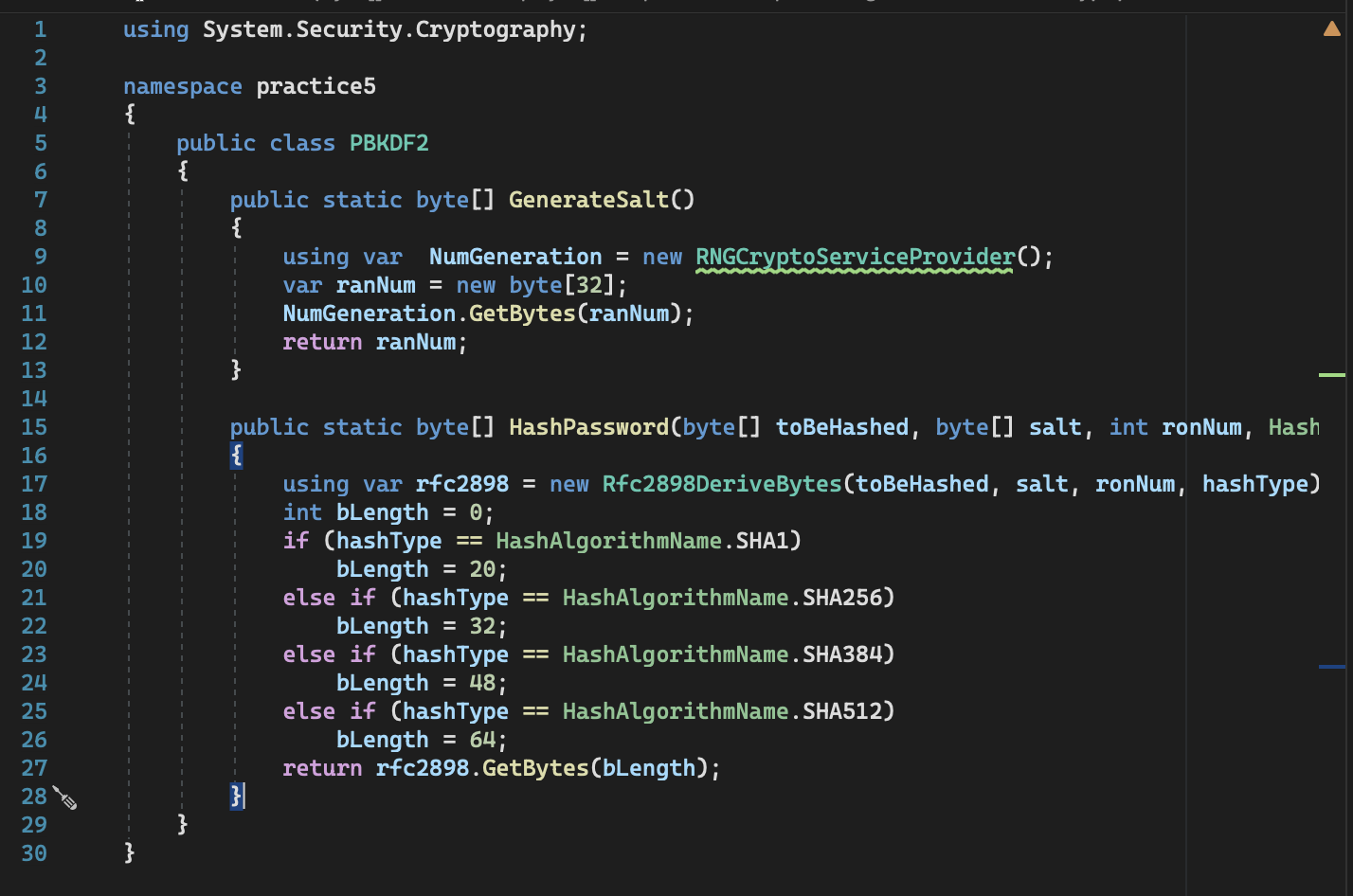
Варіант 8.

Завдання 3. Створив клас SaltedHash, який реалізує хешування паролів із додаванням додаткової ентропії. В класі Program показано приклад роботи.



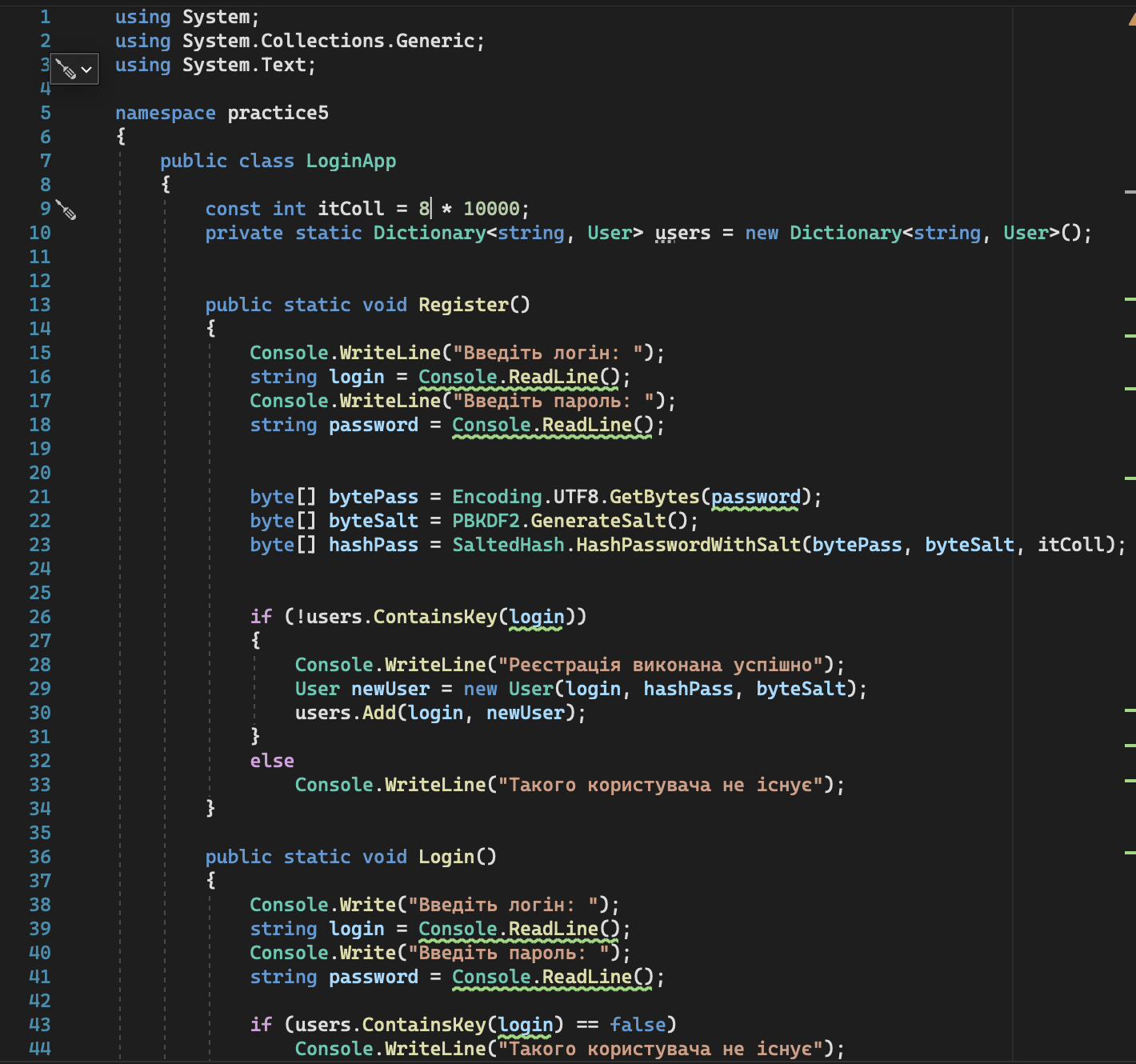


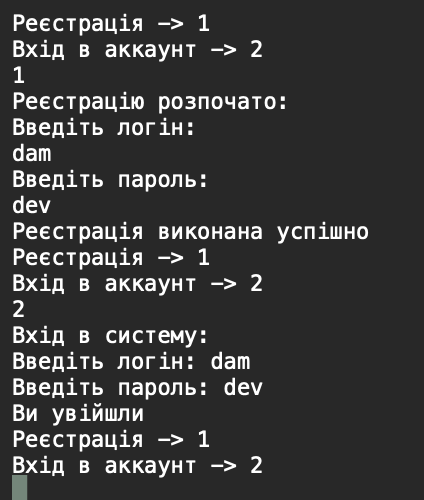
Завдання 4. Створив клас PBKDF2, який генерує "сіль", задає алгоритм хешування (MD5, SHA1, SHA256, SHA384, SHA512) та обчислює хеш для заданого числа ітерацій, а також створив програму, яка обчислює час, витрачений на обчислення хешу для різного числа ітерацій (10 значень із кроком 50'000; перше значення = номер варіанта(8) \* 10'000 )

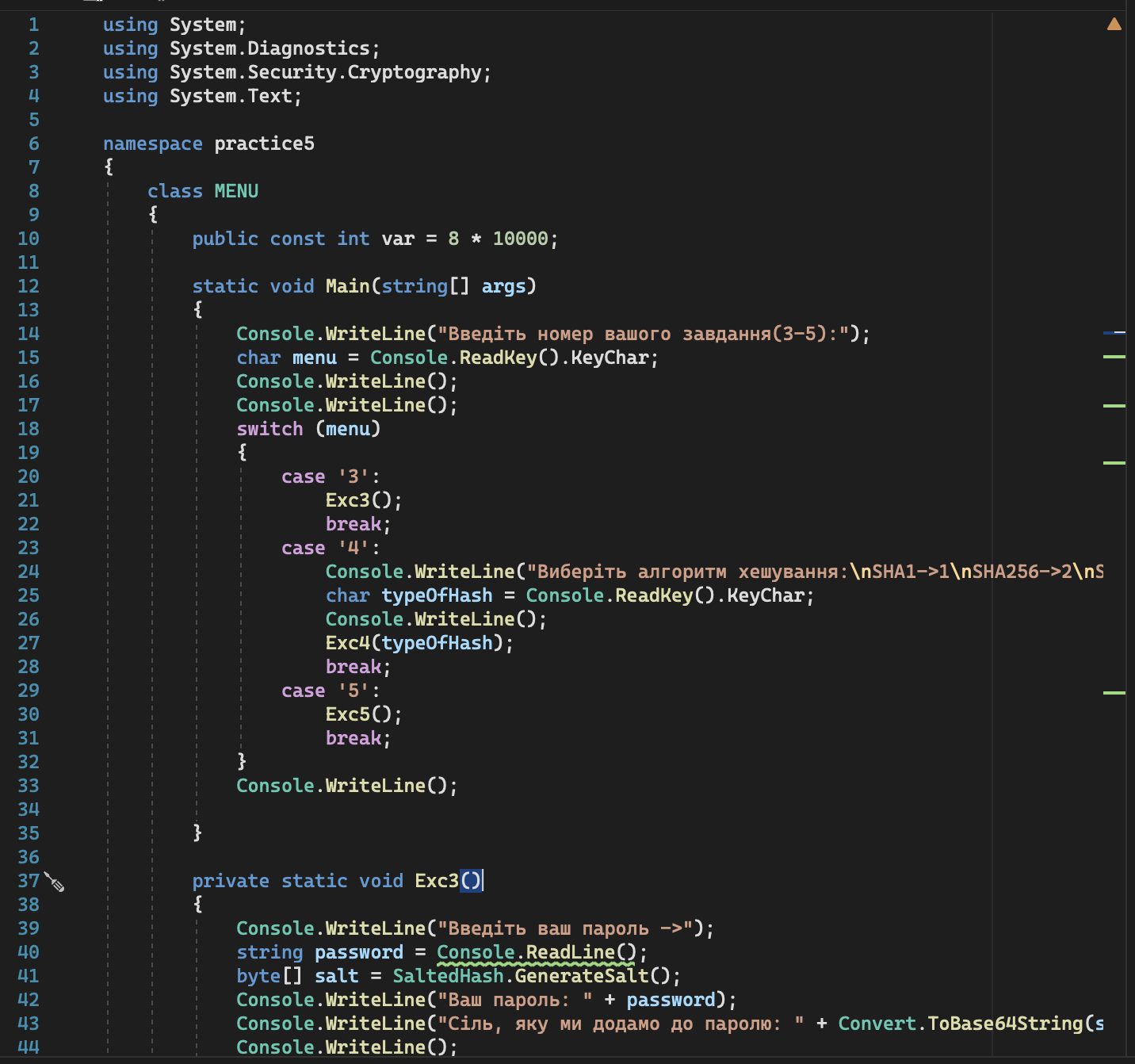




Завдання 5. Написав програму, яка реалізує хешування введеного пароля під час реєстрації користувача та зберігає логін, пароль та "сіль" у пам'яті. Можлива автентифікації за логіном і паролем. А також число ітерацій = номер варіанта(8) \* 10'000.





Все реалізував через меню класс <Menu>

Зробив висновок, що швидкість хешування залежить в основному від кількості ітерацій  
а сам алгоритм хешування сильно не впливає на час виконання процессу.

Висновок: в цій лабораторній роботі я отримав дуже цінні знання зі зберігання паролів та покращення їх безпеки.