Министерство науки И ВЫСШЕГО ОБРАЗОВАНИЯ   
Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования

«Национальный исследовательский Томский политехнический Университет»



Инженерная школа автоматизации и робототехники

Отделение автоматизации и робототехники

15.04.04 «Автоматизация технологических процессов и производств»

Вариант 1

Лабораторная работа №1

**«Аутентификация. Количественная оценка стойкости парольной защиты»**

по дисциплине:

**«Информационная безопасность автоматизированных систем»**

**Исполнитель:**

студент группы 8ТМ22 Гао Аозе \_\_\_\_\_\_\_\_\_\_

**Руководитель:**

К.т.н., доцент ОАР Суходоев Михаил Сергеевич \_\_\_\_\_\_\_\_\_\_

Томск – 2023

## Цель работы

Изучить метод количественной оценки стойкости парольной защиты и программно реализовать простейший генератор паролей, обладающий требуемой стойкостью к взлому.

## Задачи

1. В таблице 1 найти для указанного варианта значения характеристик P, V, T.

2. Вычислить по формуле (1) нижнюю границу S\* для заданных P, V, T.

3. Выбрать некоторый алфавит с мощностью A и получить минимальную длину пароля L, при котором выполняется условие (2).

4. Реализовать программу для генерации паролей пользователей. Программа должна формировать случайную последовательность символов длины L, при этом должен использоваться алфавит из A символов. Сгенерированный пароль необходимо записать в текстовый файл. Сохранить программу под именем Password\_gen.

5. Внести изменения в программу таким образом, чтобы пароль длины L включал в себя одновременно буквы, цифры и символы. Сохранить программу под именем Password\_gen\_1.

6. Оформить отчет по лабораторной работе.

**Таблица 1. Варианты заданий**

|  |  |  |  |
| --- | --- | --- | --- |
| Вариант | *P* | *V* | *T* |
| 1 | 10-6 | 3 паролей/мин | 5 дней |

## Ход работы

Количественная оценка стойкости парольной защиты

Пусть A – мощность алфавита паролей (количество символов, которые могут быть использованы при составлении пароля: если пароль состоит только из малых английских букв, то A = 26), L – длина пароля, S = AL – число всевозможных паро¬лей длины L, которые можно составить из символов алфавита A, V – скорость пе¬ребора паролей злоумышленником, T – максимальный срок действия пароля.

Тогда, вероятность P подбора пароля злоумышленником в течение срока его действия V определяется по следующей формуле:

P = (V ∙ T) / S = (V ∙ T) / AL.

Задача. Определить минимальные мощность алфавита паролей A и длину па¬ролей L, обеспечивающих вероятность подбора пароля злоумышленником не бо¬лее заданной P, при скорости подбора паролей V, максимальном сроке действия пароля T.

Данная задача имеет неоднозначное решение. При исходных данных V, T, P однозначно можно определить лишь нижнюю границу S\* числа всевозможных па¬ролей. Целочисленное значение нижней границы вычисляется по формуле

S\* = [V ∙ T / P], (1)

где [] – целая часть числа, взятая с округлением вверх.

После определения нижней границы S\* необходимо выбрать такие A и L для формирования S = AL, чтобы выполнялось следующее неравенство:

S\* ≤ S = AL. (2)

При выборе S, удовлетворяющего неравенству (2), вероятность подбора па¬роля злоумышленника (при заданных V и T) будет меньше, чем заданная P.

Следует отметить, что при осуществлении вычислений по формулам (1) и (2), величины должны быть приведены к одним размерностям.

Исходные данные: P = , T = 5 дней, V = 3 (паролей / мин) = 3 · 60·24·5 = 21600 паролей в неделю. Тогда, S\* = [(21600 ∙ 1) / ] = 216 ∙

Условию S\* ≤ AL удовлетворяют, например, такие комбинации A и L, как A = 52, L = 7 (пароль состоит из семи больших и малых символов английского алфавита), A = 30, L = 7 (пароль состоит из семи символов, среди которых могут быть указанные латинские буквы и произвольные цифры и указанные символы). 30 7

Часть программирования

В данной работе для программирования используется среда python, а в качестве программного обеспечения - visual studio.

Для начала реализуем случайный выбор 7 элементов из 52 алфавитов для генерации пароля и экспорта его в текстовый файл.

Часть кода выглядит следующим образом:

import string

import random

def generate\_password(length, alphabet):

return ''.join(random.choice(alphabet) for \_ in range(length))

def write\_to\_file(filename, passwords):

with open(filename, 'w') as file:

for password in passwords:

file.write(password + '\n')

def main():

# Combine uppercase and lowercase letters to create the alphabet

alphabet = string.ascii\_uppercase + string.ascii\_lowercase

password\_length = 7

# Input the number of passwords to generate

num\_passwords = int(input("Enter the number of passwords to generate: "))

passwords = [generate\_password(password\_length, alphabet) for \_ in range(num\_passwords)]

write\_to\_file('passwords.txt', passwords)

print(f"{num\_passwords} passwords generated and saved to passwords.txt.")

if \_\_name\_\_ == "\_\_main\_\_":

main()

import string: импортирует модуль string стандартной библиотеки Python, содержащий различные константы и функции, связанные со строками.

import random: импортирует модуль random из стандартной библиотеки Python, который содержит функции для генерации случайных чисел.

def generate\_password(length, alphabet):

return ''.join(random.choice(alphabet) for \_ in range(length))

def write\_to\_file(filename, passwords):

with open(filename, 'w') as file:

for password in passwords:

file.write(password + '\n')

def main():

# Combine uppercase and lowercase letters to create the alphabet

alphabet = string.ascii\_uppercase + string.ascii\_lowercase

password\_length = 7

# Input the number of passwords to generate

num\_passwords = int(input("Enter the number of passwords to generate: "))

passwords = [generate\_password(password\_length, alphabet) for \_ in range(num\_passwords)]

write\_to\_file('passwords.txt', passwords)

print(f"{num\_passwords} passwords generated and saved to passwords.txt.")

if \_\_name\_\_ == "\_\_main\_\_":

main()

def generate\_password(length, alphabet):: Определит функцию generate\_password, принимающую два аргумента: length - длину пароля и alphabet - алфавит.

random.choice(alphabet):: выбор случайного символа из заданного алфавита.

''.join(...) : объединяет сгенерированные случайные символы в строку.

for \_ in range(length): цикл повторяется для генерации символов пароля.

def write\_to\_file(filename, passwords):

with open(filename, 'w') as file:

for password in passwords:

file.write(password + '\n')

def main():

# Combine uppercase and lowercase letters to create the alphabet

alphabet = string.ascii\_uppercase + string.ascii\_lowercase

password\_length = 7

# Input the number of passwords to generate

num\_passwords = int(input("Enter the number of passwords to generate: "))

passwords = [generate\_password(password\_length, alphabet) for \_ in range(num\_passwords)]

write\_to\_file('passwords.txt', passwords)

print(f"{num\_passwords} passwords generated and saved to passwords.txt.")

if \_\_name\_\_ == "\_\_main\_\_":

main()

def write\_to\_file(filename, passwords):: Определяет функцию write\_to\_file, которая принимает два аргумента: filename - имя файла и passwords - список паролей.

with open(filename, 'w') as file:: открывает файл для записи ('w' указывает на режим записи) и присваивает объект file переменной file. использование оператора with гарантирует, что файл будет автоматически закрыт в конце блока, независимо от того, произойдет ли исключение или нет.

for password in passwords:: Итерация по каждому паролю в списке паролей.

file.write(password + '\n'): записывает каждый пароль в файл и добавляет в конце новую строку.

alphabet = string.ascii\_uppercase + string.ascii\_lowercase

# Input the number of passwords to generate

num\_passwords = int(input("Enter the number of passwords to generate: "))

passwords = [generate\_password(password\_length, alphabet) for \_ in range(num\_passwords)]

write\_to\_file('passwords.txt', passwords)

print(f"{num\_passwords} passwords generated and saved to passwords.txt.")

if \_\_name\_\_ == "\_\_main\_\_":

main()

string.ascii\_uppercase: строковая константа, содержащая все прописные буквы.

string.ascii\_lowercase: строковая константа, содержащая все строчные буквы.

Комбинирует прописные и строчные буквы в полный алфавит, присваиваемый переменной alphabet.

passwords = [generate\_password(password\_length, alphabet) for \_ in range(num\_passwords)]

passwords = [... for \_ in range(num\_passwords)]: Генерируем список паролей с помощью деривации списка, перебирая num\_passwords несколько раз.

generate\_password(password\_length, alphabet): вызывает функцию generate\_password для генерации пароля с длиной password\_length и символами, взятыми из алфавита.

Сгенерированный пароль добавляется в список паролей passwords.

write\_to\_file('passwords.txt', passwords)

write\_to\_file('passwords.txt', passwords): вызывает функцию write\_to\_file, которая записывает сгенерированный список паролей в файл с именем passwords.txt.

if \_\_name\_\_ == "\_\_main\_\_":

main()

if \_\_name\_\_ == "\_\_main\_\_":: проверить, запущен ли текущий модуль в качестве основного приложения.

main(): если текущий модуль запущен в качестве основного приложения, вызывается функция main. Функция main содержит основную логику генерации пароля и сохранения его в файл.

Результат работы выглядит следующим образом:

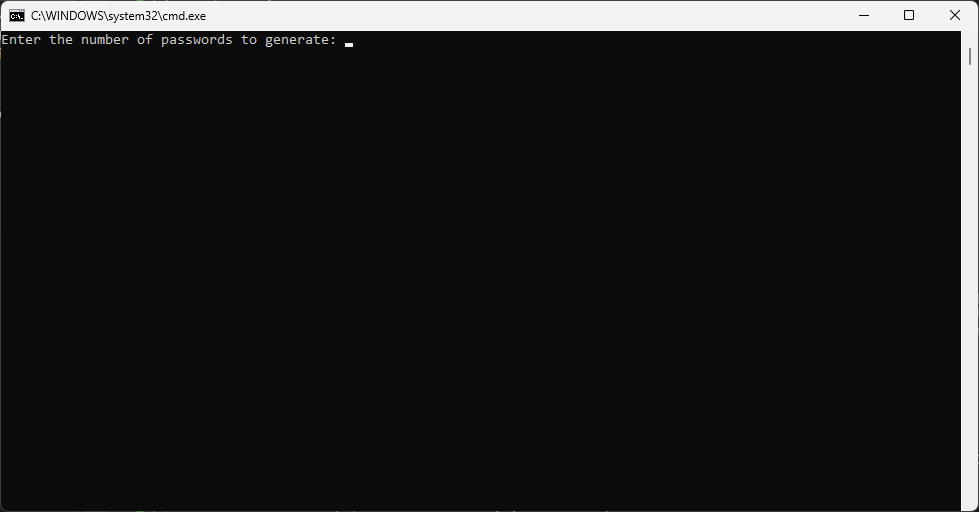


Рисунок 1 – Интерфейс времени выполнения программы

Выбираем количество генерируемых паролей

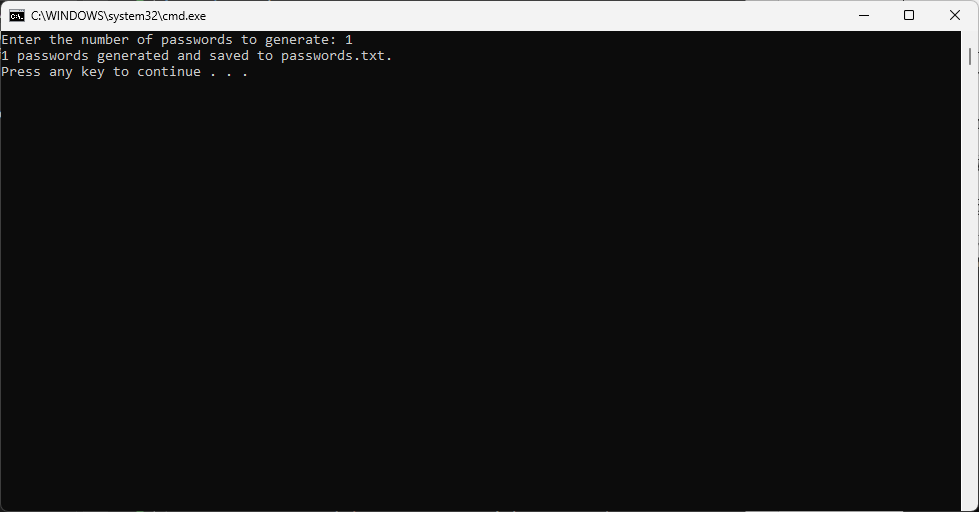


Рисунок 2 – Экран окончания программы

В месте расположения файла создается txt-файл, в котором хранится 7-значный пароль, состоящий из букв верхнего и нижнего регистра.

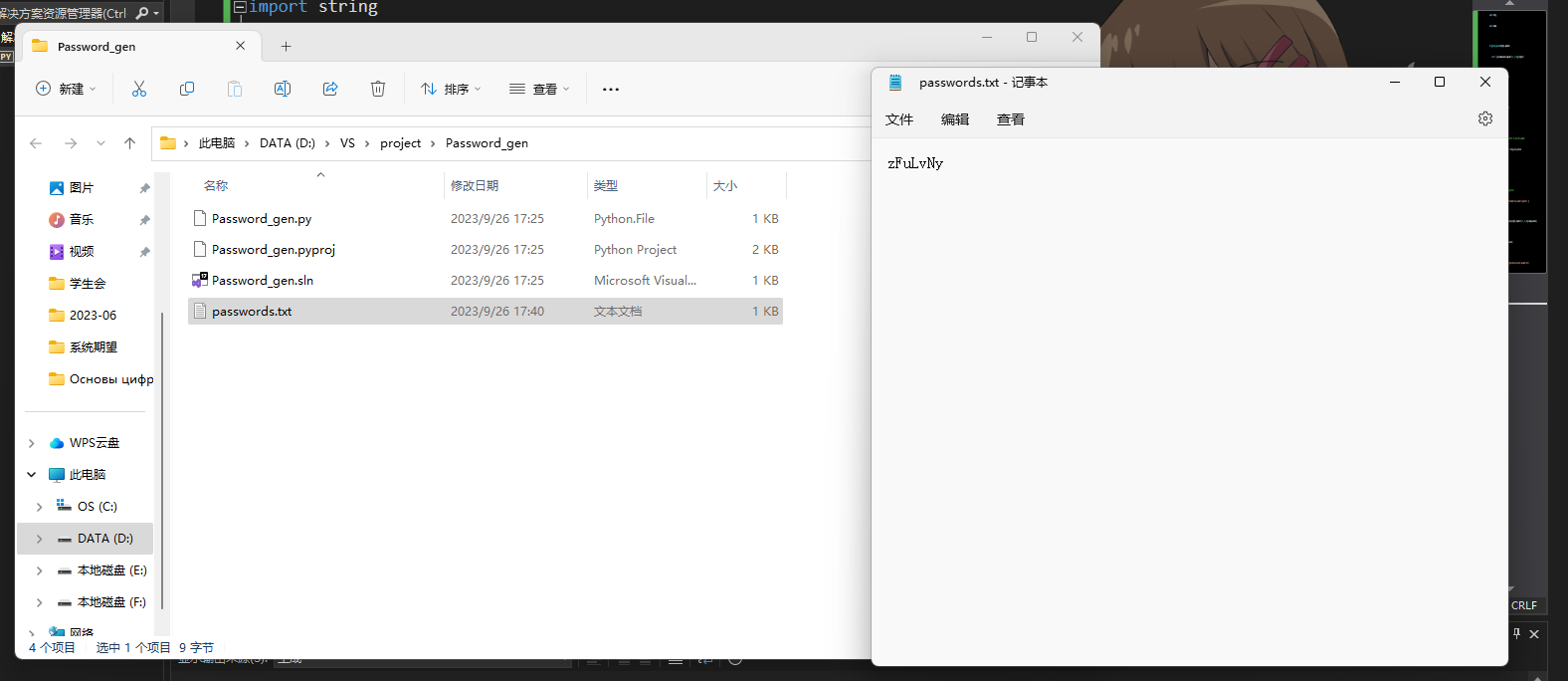


Рисунок 3 – выходной результат

Полный код приведен ниже:

import string

import random

def generate\_password(length, alphabet):

return ''.join(random.choice(alphabet) for \_ in range(length))

def write\_to\_file(filename, passwords):

with open(filename, 'w') as file:

for password in passwords:

file.write(password + '\n')

def main():

alphabet = string.ascii\_uppercase + string.ascii\_lowercase

password\_length = 7

num\_passwords = int(input("Enter the number of passwords to generate: "))

passwords = [generate\_password(password\_length, alphabet) for \_ in range(num\_passwords)]

write\_to\_file('passwords.txt', passwords)

print(f"{num\_passwords} passwords generated and saved to passwords.txt.")

if \_\_name\_\_ == "\_\_main\_\_":

main()

Далее был выполнен генератор паролей, состоящий из буквенно-цифровых и символьных символов.

Для удовлетворения требований безопасности была выбрана таблица символов, состоящая из 30 элементов, десяти букв ('A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J'), десяти цифр ('0', '1', '2', '3', '4', '5', '6', '7', '8', '9') и десяти символов ( '!' , '@', '#', '$', '%', '^', '&', '\*', '(', ')'). Генератор паролей будет генерировать 7-значный пароль, содержащий буквы, цифры и символы, и сохранять его в txt-файле.

Полный код приведен ниже:

import string

import random

custom\_alphabet = ['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J',

'0', '1', '2', '3', '4', '5', '6', '7', '8', '9',

'!', '@', '#', '$', '%', '^', '&', '\*', '(', ')']

def generate\_password():

# Ensure at least one letter, one digit, and one symbol in the password

password = [random.choice(custom\_alphabet) for \_ in range(4)]

password.append(random.choice([char for char in custom\_alphabet if char.isalpha()])) # Ensure at least one letter

password.append(random.choice([char for char in custom\_alphabet if char.isdigit()])) # Ensure at least one digit

password.append(random.choice([char for char in custom\_alphabet if char in string.punctuation])) # Ensure at least one symbol

random.shuffle(password)

return ''.join(password)

def write\_to\_file(filename, passwords):

with open(filename, 'w') as file:

for password in passwords:

file.write(password + '\n')

def main():

password\_length = 7

num\_passwords = int(input("Enter the number of passwords to generate: "))

passwords = [generate\_password(password\_length, custom\_alphabet) for \_ in range(num\_passwords)]

write\_to\_file('custom\_passwords.txt', passwords)

print(f"{num\_passwords} passwords generated and saved to custom\_passwords.txt.")

if \_\_name\_\_ == "\_\_main\_\_":

main()

Результаты прогона следующие:

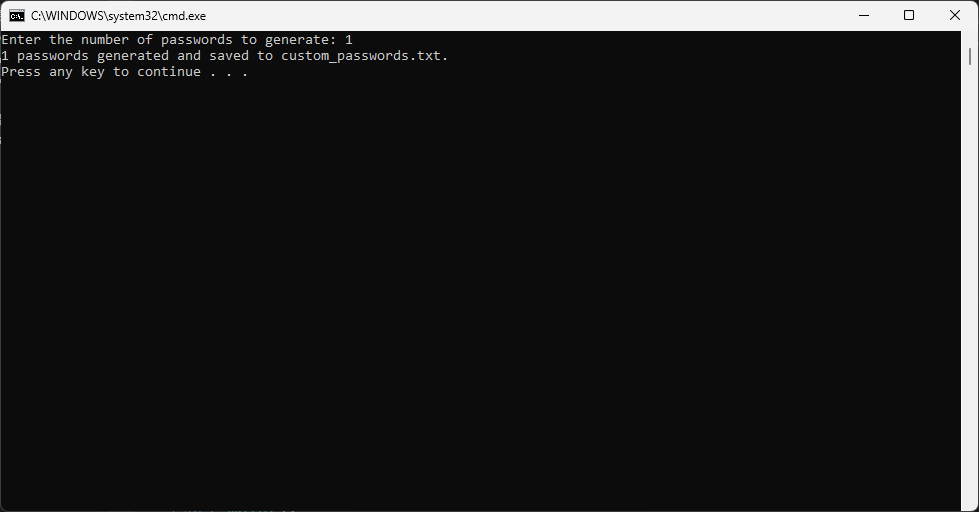


Рисунок 4 –Интерфейс времени выполнения программы

В папке был создан txt-файл, в который был записан сгенерированный пароль.

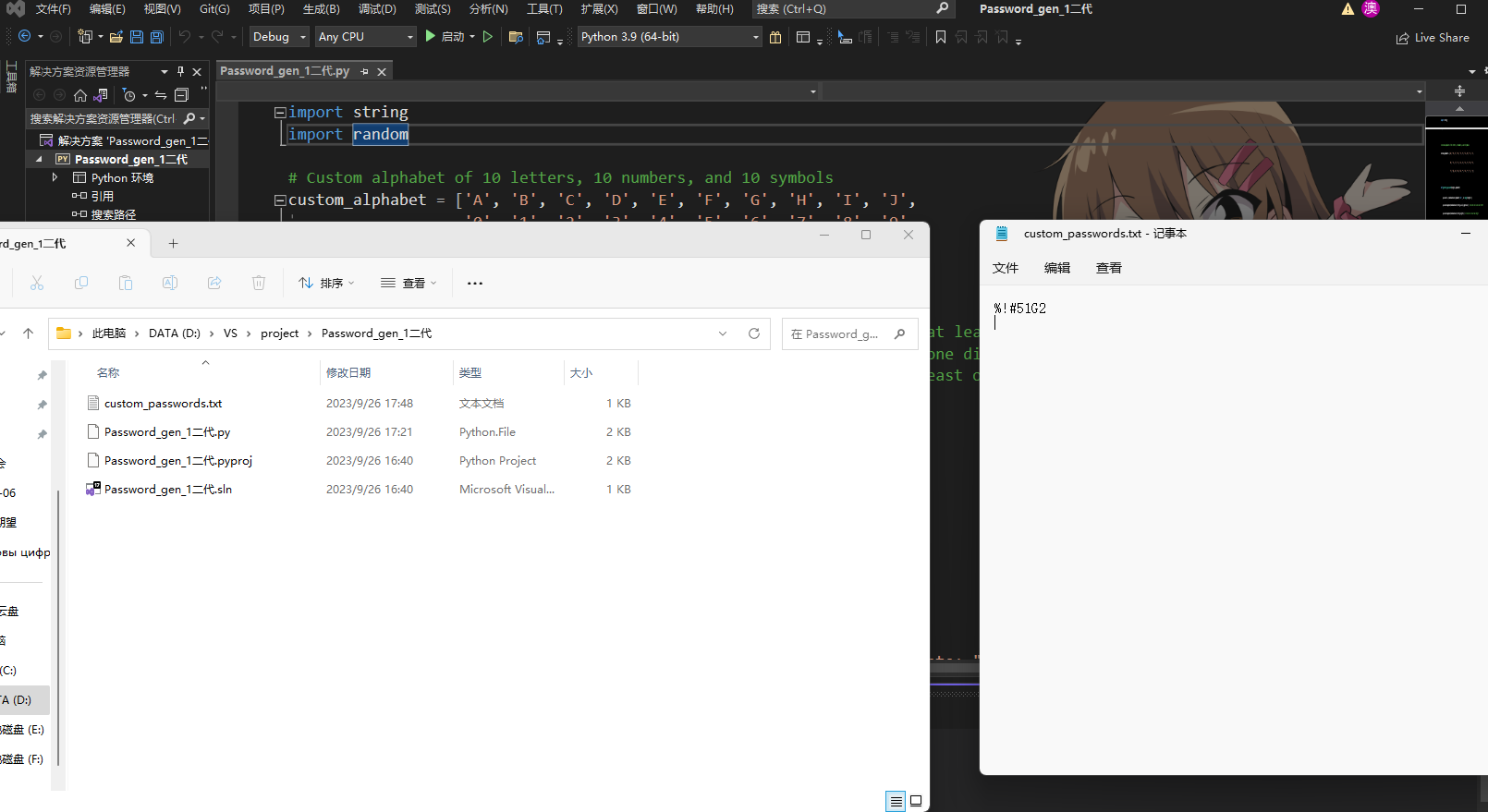


Рисунок 5 –Результаты программы

## Вывод

В данном эксперименте в теоретической части был проведен анализ метода генерации алфавита паролей и подбора номера пароля для различных скоростей взлома паролей. В экспериментальной части с помощью языка python был реализован генератор паролей, включающий генерацию паролей, содержащих только буквы, и генерацию паролей, содержащих алфавитно-цифровые и символьные знаки, а полученные результаты были выведены в текстовый файл.