

Методичні вказівки  
до лабораторної роботи №2  
«Дослідження криптостійкості паролів»  
з навчальної дисципліни вільного вибору  
«Основи інформаційної та кібербезпеки»

**Зміст практичного заняття:** Дослідити основні типи атак на інформаційні системи, а також на практиці дослідити уразливість інформаційних систем до атак на парольний захист.

**Загальні відомості**

Інформація з погляду інформаційної безпеки володіє наступними категоріями:

- **конфіденційність** – гарантія того, що конкретна інформація доступна тільки тому колу осіб, для кого вона призначена; порушення цієї категорії називається розкраданням або розкриттям інформації;
- **цілісність** – гарантія того, що інформація зараз існує в її початковому вигляді, тобто при її зберіганні або передачі не було проведено несанкціонованих змін; порушення цієї категорії називається фальсифікацією повідомлення;
- **аутентичність** – гарантія того, що джерелом інформації є саме та особа, яка заявлена як її автор; порушення цієї категорії також називається фальсифікацією, але вже автора повідомлення.

Відносно інформаційних систем застосовуються інші категорії :

- **надійність** – гарантія того, що система поводить себе в нормальному і позаштатному режимах так, як заплановано;
- **точність** – гарантія точного і повного виконання всіх команд;
- **контроль доступу** – гарантія того, що різні групи осіб мають різний доступ до інформаційних об'єктів, і ці обмеження доступу постійно виконуються;
- **контрольованість** – гарантія того, що у будь-який момент може бути проведена повноцінна перевірка будь-якого компоненту програмного комплексу;
- **контроль ідентифікації** – гарантія того, що клієнт, підключений в даний момент до системи, є саме тим, за кого себе видає;
- **стійкість до навмисних збоїв** – гарантія того, що при навмисному внесенні помилок в межах наперед обумовлених норм система поводитиметься так, як обумовлено наперед.

Кожен користувач, перш ніж здійснювати які-небудь дії в комп'ютерній системі, повинен ідентифікувати себе. У свою чергу, система повинна перевірити достовірність особи користувача, тобто що він є саме тим, за кого себе видає. Стандартним засобом перевірки достовірності (аутентифікація) є **пароль**, хоча у принципі можуть використовуватися також особисті картки, біометричні пристрої (сканування рогівки ока або відбитків пальців) або їх комбінація.

## 1. Створення надійного пароля

Щоб створити надійний пароль, необхідно дотримуватись 4 основних вимог, перерахованих далі в порядку важливості:

- 1) Користувач повинен легко запам'ятати пароль.
- 2) Інші особи не повинні бути здатні вгадати цей пароль.
- 3) Ніяка програма не повинна вміти швидко підібрати пароль.
- 4) Пароль повинен бути складним, містити цифри, символи, великі і малі літери.

Більшість організацій вимагає, щоб пароль складався з комбінації цифр, символів, заголовних і малих літер. Паролі, що відповідають цій політиці, цілком припустимі, але тільки якщо користувачеві буде легко їх запам'ятати.

Приклад типових вимог до складання пароля:

- Пароль повинен бути довжиною мінімум 8 символів.
- Пароль повинен містити великі і малі літери.
- Пароль повинен містити цифру.
- Пароль повинен містити символ (а не літеру і не цифру).

Якщо проаналізувати характеристики надійного пароля і типову політику створення пароля, представлену вище, то можна прийти до висновку, що дана політика суперечить першим двом пунктам.

Рекомендації ESET щодо створення надійного паролю:

1. Створіть унікальний пароль для кожного облікового запису та тримайте його в секреті.
2. Довший пароль забезпечує більший рівень безпеки. Використовуйте щонайменше вісім символів, але збільшуйте довжину паролю для захисту цінних даних або рахунків. Для легкості запам'ятовування використовуйте ключові фрази або менеджер паролів (наприклад, окреме програмне забезпечення або функцію антивірусного рішення).
3. Уникайте поширених слів (загальні слова, імена, дати, цифри) або очевидних варіантів, таких як 12345678, password або QWERTY.
4. Для ускладнення паролю слід додавати цифри та спеціальні символи (@, #, ! та інші) або використовувати їх для заміни деяких букв у паролі.
5. Під час заміни букв на символи не допускайте поширених помилок, таких, як заміна «а» на «@» або «і» на «l» чи «!».
6. Регулярно змінюйте паролі, особливо у разі потреби в захисті важливих даних. Чим більш важлива інформація, тим частішими мають бути оновлення даних входу.
7. Не слід повторно використовувати один і той же пароль для доступу до різних облікових записів. Застосування одного паролю для багатьох профілів підвищує уразливість Ваших конфіденційних даних, оскільки у разі його крадіжки інформація всіх акаунтів з ідентичним даними входу опиняться у руках злоумисників.

Онлайн-інструменти генерації паролів:

<https://passwordsgenerator.net/>

<https://www.random.org/passwords/>

<https://www.lastpass.com/password-generator>

...

Існує підхід, у якому для створення надійних паролів рекомендують складати ланцюжок з чотирьох або більше випадкових слів і пов'язувати їх один з одним. Пароль **correcthorsebatterystaple** надійніший, ніж **Tr0ub4dor&3**. Незважаючи на те що другий пароль повністю відповідає наведеним вище політикам, програми для злому пароля досить ефективні, щоб підібрати такий тип пароля. Хоча пароль **correcthorsebatterystaple** не буде прийнятий більшістю політик створення паролів, насправді він набагато надійніший, ніж другий пароль. Користувачеві простіше його запам'ятати, він дуже довгий, а сам набір слів настільки випадковий, що робить задачу його злому практично нездійсненною.

При використанні онлайн-інструменту <https://preshing.com/20110811/xkcd-password-generator/> (або <https://correcthorsebatterystaple.net/>) паролі створюються на основі випадкових слів. Так як слова злиті разом, їх не можна виокремити як окремі слова зі словника.

## 2. Зберігання паролів

Численні дослідження показали, що середньостатистичний користувач має десятки паролів для різних облікових записів, що ускладнює дотриманням всіх вищевказаних правил. Спростити управління низкою паролів можуть ключові фрази. Попри свою довжину, вони легко запам'ятовуються користувачем. Однак під час вибору фрази слід уникати очевидних варіантів, таких як відомі цитати з фільмів або книг. Для підвищення рівня безпеки варто додавати до ключової фрази знаки пунктуації, цифри, верхні або нижні підкреслювання та пробіли.

Найкраще зберігати паролі в голові. Туди хакерам точно не добратися. Якщо вирішите зберегти його на папірці або зберегти в нотатках смартфона, пам'ятайте - ці способи ненадійні. Тому будьте обережні.

Якщо всі паролі у вас дуже складні і унікальні, їх буде важко запам'ятати. На щастя, існують менеджери паролів, які допоможуть придумати і запам'ятати паролі до всіх акаунтів. У цьому випадку єдиний пароль, який потрібно вивчити – це пароль до менеджера.

Приклади менеджерів паролів:

1. RememBear (<https://www.remembear.com/>)
2. LastPass (<https://www.lastpass.com/>)
3. Dashlane (<https://www.dashlane.com/>)
4. KeePass (<https://keepass.info/>)
5. Sticky Password (<https://www.stickypassword.com/>)
6. [менеджер паролів Google Chrome](#)

Популярним менеджером паролів є сервіс **Last Pass**, на якому Ви можете створити пробний обліковий запис:

- Відкрийте веб-браузер і перейдіть за посиланням: <https://lastpass.com/>.
- Клацніть Отримати LastPass Free (Get LastPass Free), щоб створити пробний обліковий запис.
- Заповніть поля відповідно до інструкції.
- Задайте майстер-пароль. З цим паролем ви будете входити в свій обліковий запис LastPass.
- Завантажте та встановіть клієнт LastPass для своєї операційної системи.
- Відкрийте клієнт і увійдіть в нього зі своїм майстер-паролем LastPass.
- Вивчіть менеджер паролів LastPass.

### 3. Методи розкриття паролів

**3.1. Брутфорс** (від англійського bruteforce — повний перебір або метод «грубої сили») — один з популярних методів злому паролів на серверах і в різних програмах. Полягає він у тому, що програма-зломщик намагається отримати доступ до будь-якої програми (наприклад, до поштової скриньки) шляхом перебору паролів за критеріями, визначеним власником даної програми: за словником, по довжині, з поєднанням цифр.

Спосіб злому брутфорсом є досить тривалим, але потужним, тому залишається на озброєнні у хакерів і на сьогоднішній день, а з урахуванням збільшення потужностей комп'ютерів та пропускну здатності інтернет-каналів, залишиться на озброєнні ще на довгий час.

При брутфорсі найчастіше використовується словникова атака – підбір паролів йде з текстового файлу заздалегідь складеного словника. Даний спосіб атаки дуже ефективний при масовому зломі, коли зловмисник, припустимо, намагається зламати якийсь діапазон номерів. При цьому існує досить велика ймовірність, що йому це вдасться.

Програм для проведення брутфорсу на просторах Інтернет викладено дуже багато, також існує велика кількість безкоштовних і платних словників до них.

#### Приклад тривалості підбору паролів

У таблиці представлено приблизний час повного перебору паролів в залежності від їх довжини. Передбачається, що в паролі можуть використовуватися 36 різних символів (латинські літери одного регістру та цифри), а швидкість перебору становить 100 000 паролів в секунду.

Кількість знаків	Кількість варіантів	Час перебору
1	36	менше секунди
2	1296	менше секунди
3	46 656	менше секунди

4	1 679 616	17 секунд
5	60 466 176	10 хвилин
6	2 176 782 336	6 годин
7	78 364 164 096	9 днів
8	$2,821\,109\,9 \times 10^{12}$	11 місяців
9	$1,015\,599\,5 \times 10^{14}$	32 роки
10	$3,656\,158\,4 \times 10^{15}$	1162 роки
11	$1,316\,217\,0 \times 10^{17}$	41 823 роки
12	$4,738\,381\,3 \times 10^{18}$	1 505 615 років

Таким чином, паролі довжиною до 6 символів у загальному випадку не є надійними.

### ***3.2. Атака з використанням райдужної таблиці***

Комп'ютерні системи, які використовують паролі для автентифікації, повинні якимось чином визначати правильність введеного пароля. Найпростішим способом вирішення даної проблеми є зберігання списку всіх допустимих паролів для кожного користувача. Мінусом даного методу є те, що в разі несанкціонованого доступу до списку, зломисник дізнається всі паролі. Більш поширений підхід полягає у зберіганні значень криптографічної хеш-функції від паролів.

Слово хеш походить від англійського «hash», яке можна перекласти як "плутанина", "мішанина", "фарш". Часто ще про сам процес говорять «хешування», від англійського "hashing" (рубати, подрібнювати).

Хеш-функція – функція, що перетворює вхідні дані будь-якого (як правило великого) розміру в дані фіксованого розміру. Прості хеш-функції (ненадійні, але які просто і головне швидко обчислюються) застосовують для перевірки цілісності передачі пакетів по мережевому протоколу TCP/IP та інших для виявлення апаратних помилок – так зване "надлишкове кодування". Якщо хеш отриманого пакета даних співпадає з відправленим разом з пакетом даних (так званою "контрольною сумою), то це може означати, що втрати чи помилок при передачі пакета даних по мережі не сталося.

Великі по складності хеш-функції використовують у криптографії. Головна умова для них — неможливість по кінцевому результату (хешу) обчислити початковий масив даних. Ще одна головна умова — стійкість до колізій, тобто низька ймовірність отримати два однакових хеша з двох різних масивів даних при обробці їх такою функцією. Розрахунки по таким алгоритмам складні і ресурсовитратні, але тут вже головне не швидкість, а надійність.

Для доступу до сайтів та серверів по логіну і паролю часто використовують хешування. Зберігати паролі у відкритому вигляді для подальшої зв'язки з тим, що ввів користувач, досить ненадійно з точки зору можливості їх викрадення. Тому зберігають хеші усіх паролів. Користувач вводить пароль, миттєво розраховується його хеш і звіряється з тим, що є у базі даних. Надійно і дуже просто. Як правило для

такого типу хешування використовують складні функції з дуже високою криптостійкістю, щоб по хешу було неможливим відновити пароль.

Райдужна таблиця – це список попередньо обчислених хешів (числових значень зашифрованих паролів), що використовуються більшістю сучасних систем. Таблиця включає в себе суми всіх можливих комбінацій паролів для будь-якого виду алгоритму хешування. Час, необхідний для злому пароля за допомогою райдужної таблиці, зводиться до того часу, який потрібен, щоб знайти захешований пароль у списку.

Один з поширених методів захисту від взлому за допомогою райдужних таблиць – використання необоротних хеш-функцій, які включають salt («сіль», «модифікатор»).

Сіль збільшує довжину і складність пароля. Якщо таблиця розрахована на деяку довжину або на деякий обмежений набір символів, то сіль може запобігти відновленню пароля.

**3.3. Фішинг** (англ. phishing від fishing — риболовля, вивуджування) – вид інтернет-шахрайства, метою якого є отримання доступу до конфіденційних даних користувачів – логінів і паролів. Це досягається шляхом проведення масових розсилок електронних листів від імені популярних брендів, а також особистих повідомлень всередині різних сервісів, наприклад, від імені банків або всередині соціальних мереж. У листі часто міститься посилання на сайт, який зовні не відрізняється від справжнього, або на сайт з перенаправленням. Після того, як користувач потрапляє на підроблену сторінку, шахраї намагаються різними психологічними прийомами спонукати користувача ввести на підробленій сторінці свої логін і пароль, які він використовує для доступу до певного сайту, що дозволяє шахраям отримати доступ до акаунтів і банківських рахунків.

Найпростіший спосіб злому – запитати у користувача його пароль. Фішингове повідомлення направляє читача на підроблені сайти онлайн-банкінгу, платіжних систем або інші сайти, на яких потрібно обов'язково ввести особисті дані, щоб "виправити якусь страшну проблему з безпекою".

### **3.4. Соціальна інженерія**

Соціальна інженерія дотримується тієї ж концепції, що і фішинг - "запитати у користувача пароль", але не за допомогою поштової скриньки, а у реальному світі.

Найчастіший випадок використання соціальної інженерії – зателефонувати в офіс під виглядом співробітника ІТ-безпеки та просто попросити пароль доступу до мережі.

### **3.5. Шкідливе програмне забезпечення**

Програма перехоплення інформації, що вводиться з клавіатури або виводиться на екран, може бути встановлена шкідливим ПЗ, яке фіксує всю інформацію, яка

вводиться, або створює скріншоти під час процесу авторизації, а потім направляє копію цього файлу хакерам.

Деякі шкідливі програми шукають існуючий файл з пароллями веб-браузера клієнта, потім копіюють цей файл, який (крім добре зашифрованих) буде містити легкодоступні збережені паролі з історії сторінок, відвіданих користувачем.

### ***Клавіатурні шпигуни***

Клавіатурні шпигуни - це програми для прихованого запису інформації про натискання користувачем клавіші.

Як правило, сучасні клавіатурні шпигуни не просто записують коди, які вводяться клавішами - він "прив'язує" клавіатурний ввід до поточного вікна і елемента вводу.

Записана інформація зберігається на диску і більшість сучасних клавіатурних шпигунів можуть формувати різні звіти, можуть передавати їх по електронній пошті або http/ftp протоколу. Крім того, ряд сучасних клавіатурних шпигунів користується RootKit технологіями для маскуванню слідів своєї присутності в системі.

### ***Клавіатурний шпигун на базі драйвера***

Даний метод більш ефективний, ніж описані вище методи. Можливі як мінімум два варіанти реалізації цього методу – написання та встановлення в систему свого драйвера клавіатури замість штатного.

### ***Апаратні клавіатурні шпигуни***

У ході вирішення завдань щодо захисту від витоку інформації часто розглядають тільки різні програмні засоби для шпигунства за роботою користувача. Однак крім програмних можливі й апаратні засоби:

- встановлення пристрою стеження в розрив кабелю клавіатури
- (наприклад, пристрій може бути виконано у вигляді перехідника PS/2);
- вбудовані пристрої стеження в клавіатуру;
- зчитування даних шляхом реєстрації ПЕМВН (побічних електромагнітних випромінювань і наведень);
- візуальне спостереження за клавіатурою.

Апаратні клавіатурні шпигуни зустрічаються набагато рідше, ніж програмні.

## **Методики пошуку клавіатурних шпигунів**

- ✓ Пошук за сигнатурами. Даний метод не відрізняється від типових методик пошуку вірусів.
- ✓ Евристичні алгоритми. Це методики пошуку клавіатурного шпигуна за його характерними особливостями. Цей метод найбільш ефективний для пошуку клавіатурних шпигунів самого поширеного типу – заснованих на пастках. Проте такі методики дають багато помилкових спрацьовувань. Дослідження показали, що існують сотні безпечних програм, які не є клавіатурними шпигунами, але встановлюють пастки для стеження за клавіатурним введенням і мишею.

- ✓ Моніторинг API функцій, використовуваних клавіатурними шпигунами. Дана методика заснована на перехопленні ряду функцій, що застосовуються клавіатурним шпигуном – зокрема, функцій SetWindowsHookEx, UnhookWindowsHookEx, GetAsyncKeyState, GetKeyboardState.
- ✓ Відстеження процесів та сервісів, що використовуються системою драйверів. Це універсальна методика, застосовна не тільки проти клавіатурних шпигунів.

**3.6. Аналізатор трафіку або сніфер** (від англ. to sniff — нюхати) — програма або програмно-апаратний пристрій, призначений для перехоплення і подальшого аналізу, або тільки аналізу мережного трафіку призначеного для інших вузлів.

Перехоплення трафіку може здійснюватися:

- звичайним «прослуховуванням» мережевого інтерфейсу (метод ефективний при використанні в сегменті концентраторів (хабів) замість комутаторів (світців));
- підключенням сніфера в розрив каналу;
- відгалуженням (програмним або апаратним) трафіку і спрямуванням його копії на сніфер;
- через аналіз побічних електромагнітних випромінювань і відновлення трафіку, що таким чином прослуховується;
- через атаку на канальному (MAC-spoofing) або мережевому рівні (IP-spoofing), що приводить до перенаправлення трафіку жертви або всього трафіку сегменту на сніфер з подальшим поверненням трафіку в належну адресу.

За допомогою сніфера, можна отримати будь-яку інформацію, яка була передана в мережі, як паролі, пошта, конфіденційні документи, і будь-яку іншу незашифровану інформацію. По суті, сніфер діє як реєстратор програм, встановлених на машинах, від яких передаються пакети.

#### **4. Робота з програмами злому на прикладі KRYLack ZIP Password Recovery**

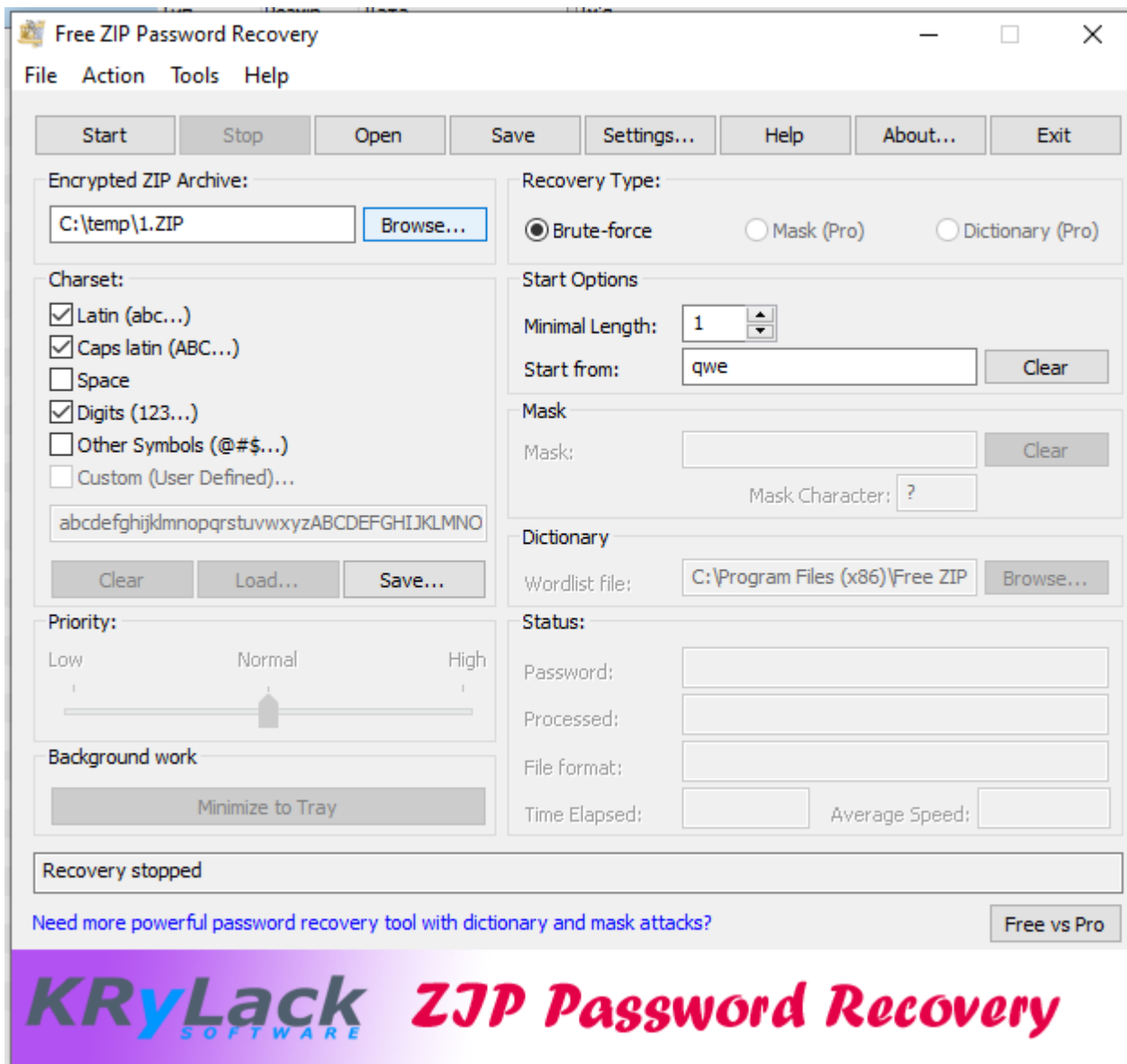
У даній лабораторній роботі використовується програмний продукт для розкриття закритих паролем архівів: KRYLack ZIP Password Recovery.

Програма ZIP Password Recovery використовується для відновлення забутих паролів ZIP-архівів. Існує два способи розкриття паролів: перебір (brute force) і атака по словнику (dictionary-based attack) у платній версії програми.

Завантажити програму **KRYLack ZIP Password Recovery** можна на офіційному сайті <https://www.krylack.com/free-zip-password-recovery/>.

Перш за все, потрібно відкрити архів, пароль до якого втрачено або він невідомий. Для цього потрібно або натиснути на кнопку **Browse...** Виберіть архів і натисніть на кнопку **Відкрити**. На вкладці **Charset** пропонуються опції на вибір символів, які можуть становити пароль до архіву: малі літери (abc...), великі літери (ABC...), пробіл (Space), цифри (Digits), інші символи (@#\$...).





У полі **Minimal Length** (Мінімальна довжина) визначається мінімальна довжина пароля.

За замовчуванням пропонується найбільш надійний і простий тип атаки: **Brute - force** (Перебір).

Тип **Mask** (Маска) дозволяє визначити маску в паролі, якщо Ви знаєте хоча б кілька символів з пароля. Тим самим можна зменшити час сканування. Наприклад, Ви знаєте, що пароль буде містити 8 символів. При цьому пароль починається з «х», і закінчується на «99». Інші символи є малими або великими літерами. Відомі символи вказуються в явному вигляді, а невідомі символи вказуються у вигляді знаку питання (?). Маска буде виглядати так: «х?????99».

Тип **Dictionary** (За словником) призначений для пошуку пароля на основі словника.

Інші програми пропонують додаткові опції, наприклад, **Smart mutations** (Розумні мутації).

Ось короткий і неповний перелік того, які правила можна використовувати:

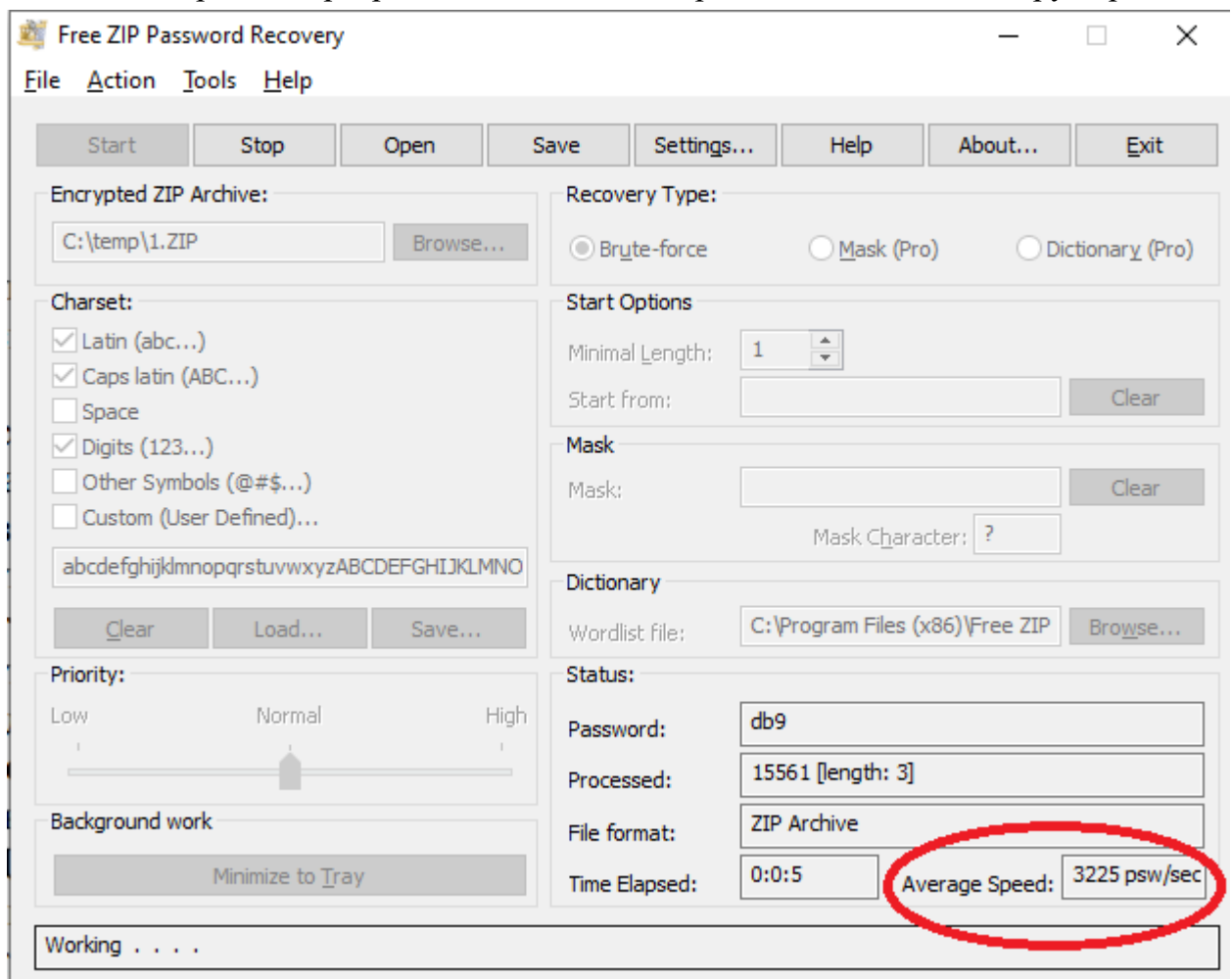
- {Ротація: password -> asswordp

- с Великі букви: password -> Password
- d Дублювання слова: password -> passwordpassword
- q Дублювання символів: password -> ppaasssswwoorrrd
- r Зворотний порядок: password -> drowssap
- V Великі приголосні: password -> PaSSWoRD
- yN Дублювати перші N знаків
- \$ X Додати символ X в кінець слова
- ^ X Вставити символ X в початок слова
- @X Видалити всі символи X з слова
- ! X Не пробувати пароль, якщо в ньому міститься символ X
- / X Не пробувати пароль, якщо в ньому НЕ міститься символу X

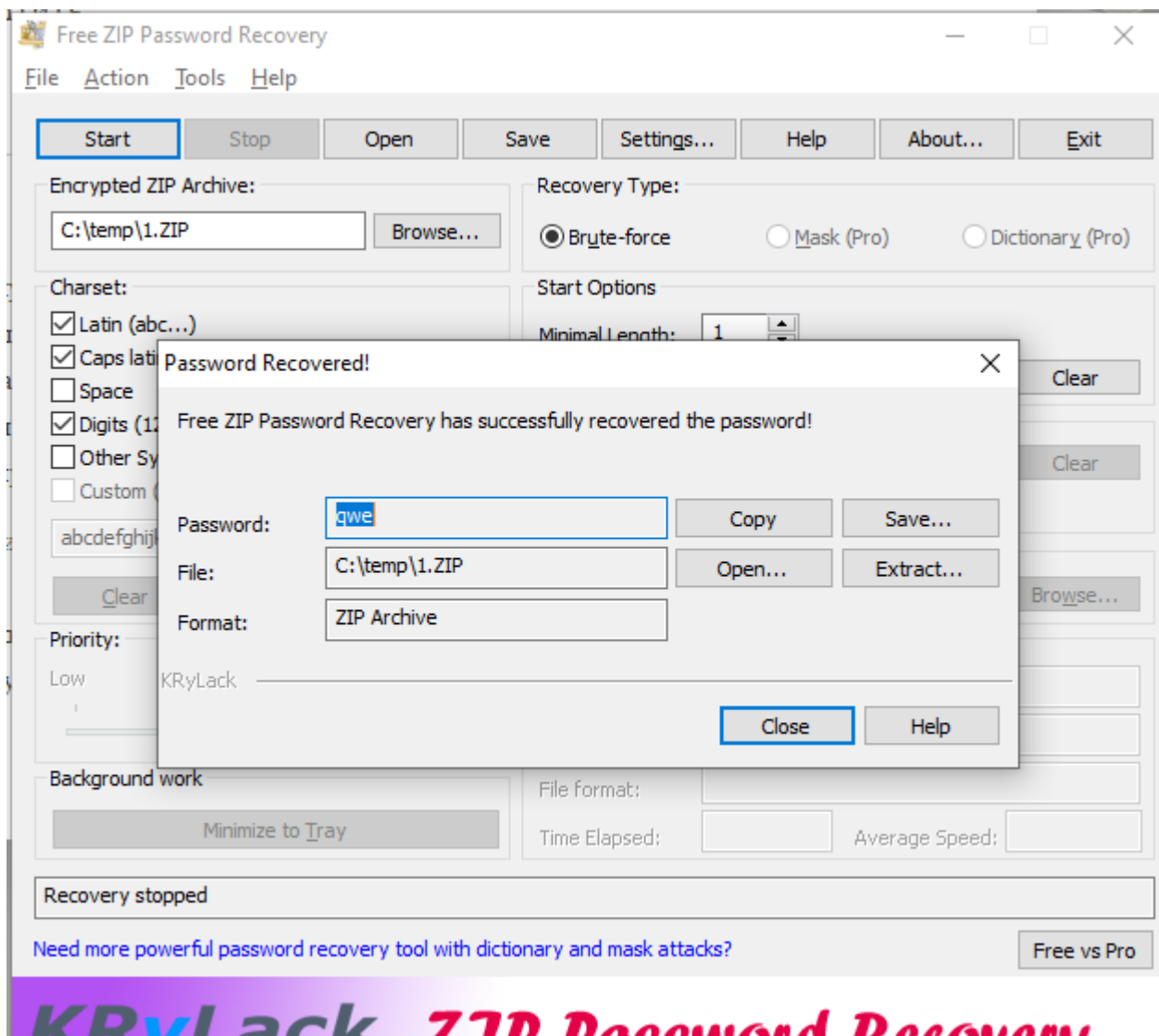
Для запуску шуканого пароля натисніть на кнопку **Start** (Старт).

Можна обмежити кількість тестованих паролів, задавши початковий пароль.

Під час роботи програми можна бачити середню швидкість підбору паролей:



Якщо пароль буде знайдений, то він виводиться в спеціальне вікно. Використовуючи цей пароль можна зразу розархівувати відповідний архів.



## Завдання до лабораторної роботи

**Завдання А.** Розкриття закритих паролем архівів у програмі KRyLack ZIP Password Recovery.

1. За допомогою програми архівації даних (наприклад, 7-zip <https://7-zip.org.ua/download/>) створіть архівний файл zip і встановіть на нього пароль завдовжки 2 маленькі латинські літери.
2. Відкрийте створений файл в програмі KRyLack ZIP Password Recovery (<https://www.krylack.com/free-zip-password-recovery/>) (можна використати іншу програму зі списку <https://geekflare.com/best-zip-password-recovery-tools/>) і вкажіть використовуваний набір - малі латинські символи, мінімальну довжину пароля – 1 і виконаєте перебір. Зафіксуйте час перебору і середню швидкість в таблиці 1.1.

Таблиця 1.1 - Результати перебору паролів в програмі  
KRyLack ZIP Password Recovery

№ з/п	Набір символів	Довжина пароля, символів	Час перебору, с	Швидкість перебору, паролів/с
1.				
2.				
...				

3. Повторіть дослідження архівного файлу захищеного паролями довжиною 3, 4, 5 малих латинських символів і занесіть результати в табл. 1.1.
4. За допомогою програми архівації даних створіть архівний файл і встановіть на нього пароль завдовжки 2 символи з наборів малих і великих латинських символів і занесіть результати в табл. 1.1..
5. Повторіть дослідження архівного файлу захищеного паролями 3, 4, 5 малих і великих латинських символів; результати занесіть в табл. 1.1.
6. Повторіть дослідження архівного файлу захищеного паролями довжиною 2, 3, 4, 5 малих, великих латинських символів і цифр, результати занесіть в табл. 1.1.
7. Повторіть дослідження архівного файлу захищеного паролями довжиною 2, 3, 4, 5 малих, великих латинських символів, цифр і спеціальних символів, результати занесіть в табл. 1.1.
8. Побудуйте графіки залежності часу підбору пароля від довжини пароля для різних використаних наборів символів.
9. Сформуйте звіт за результатами виконаного завдання (зразок звіту додається).

**Завдання Б.** Напишіть програму, яка відновлює пароль з хешу.

### *Передісторія*

Уявимо себе в ролі хакера, який отримав доступ до бази даних з хешами паролів. Для нас не важливо яким чином отримана ця база даних. Є безліч способів щоб її отримати і такі речі, нажаль, досить часто трапляються, зокрема навіть з таким сервісами як Facebook. Наша ціль показати наскільки буває просто зламати пароль від гіпотетичного сервісу, та те що безпека ваших даних залежить в основному тільки від вас самих і надійності вибраного вами паролю.

### *Теорія*

У наш час на більшості систем, присутній файл який зберігає у собі імена користувачів та паролі. На щастя, паролі там не зберігаються «у чистому вигляді» натомість, вони зашифровані із використанням «односторонньої хеш-функції». Коли користувач входить до системи, вводячи ім'я користувача і пароль, пароль шифрується за допомогою тієї ж хеш-функції, і результат порівнюється із відповідним записом у цьому файлі. Якщо вони збігаються, користувачу дозволяють вхід. Якщо ви колись забували якийсь пароль, вам могли сказати, що технічна підтримка не може подивитись ваш пароль, але може змінити його для вас. Швидше за все, технічна підтримка може бачити тільки хеш вашого пароля, а не сам пароль. Але вони можуть створити для вас новий хеш.

Доволі часто зловмисники, у випадку отримання ними цього файлу, можуть здогадатись (і перевірити) користувацький пароль, чи зламати його за допомогою повного перебору (тобто спробувавши всі можливі паролі).

Нижче ви можете побачити, як може виглядати спрощений такий файл де кожен рядок відформатовано як **пароль : хеш**.

```
12345 : 827ccb0eea8a706c4c34a16891f84e7b
abcd : e2fc714c4727ee9395f324cd2e7f331f
надійний : 80920db081c182107eb70f71d4ee02f6
пароль : e242f36f4f95f12966da8fa2efd59992
надійний пароль : 805aa81b1b4e886b665265b21f2d9449
```

Паролі ліворуч зашифровані за допомогою хеш-функції MD5. Вона була надзвичайно популярною для хешів паролів в проміжку з 1991 до 2008 років. Зараз її не використовують для збереження паролів, в даний момент використовують більш надійні хеш-функції, такі як SHA-3 або інші. Для нас не важливо яка саме функція це буде, для нас є важливим розуміння принципу як відбувається підбір паролю.

### *Практика*

Цього разу ми скористаємось мовою C++ (надзвичайно популярна мова для вирішення складних завдань). У вас вже має бути компілятор, після попереднього завдання якщо ні, то переходимо за посиланням та інсталуємо компілятор.

## Win:

Переходимо за посиланням та інсталуємо компілятор gcc

<http://www.equation.com/servlet/equation.cmd?fa=fortran>

(наприклад файл gcc-8.3.0-64.exe)

Як встановити компілятор gcc / g ++ для Windows дивіться файл Додаток 3 до лабораторної роботи 1.

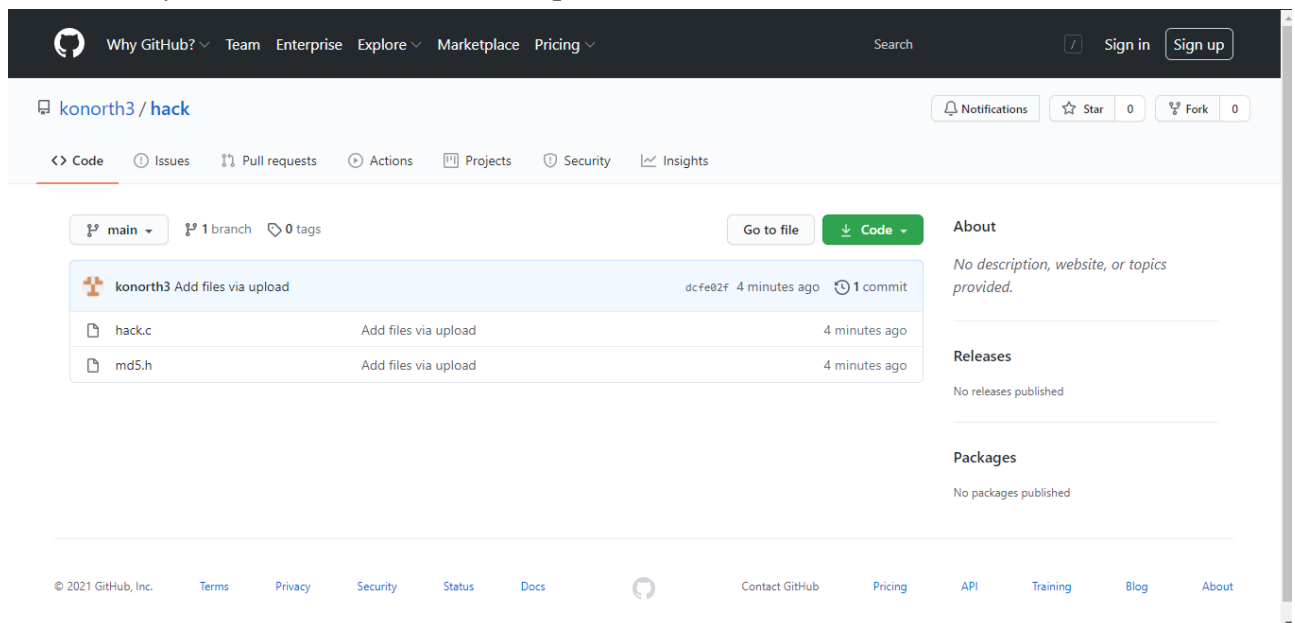
## Завдання

Написати програму, яка відновлює пароль з хеш-файлу.

Нам буде потрібен код програми, знайти його можна за посиланням

<https://github.com/konorth3/hack>

або у вкладці «Навчальні матеріали».

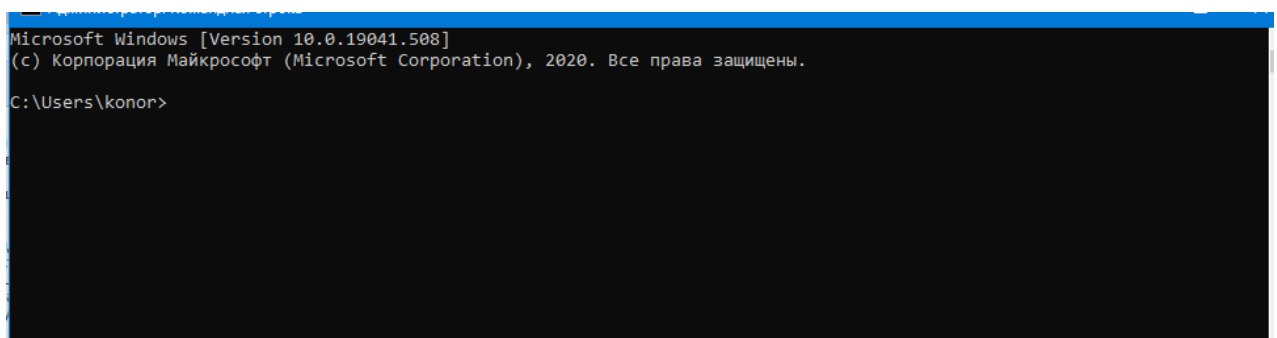


Натискаємо на зелену клавішу “**Code**” та завантажуємо zip архів.

Розпаковуємо його та переходимо в папку з кодом.

Відкріємо командний рядок.

Для цього заходимо в меню пуск і в полі пошук вводимо **cmd** та обираємо командний рядок.



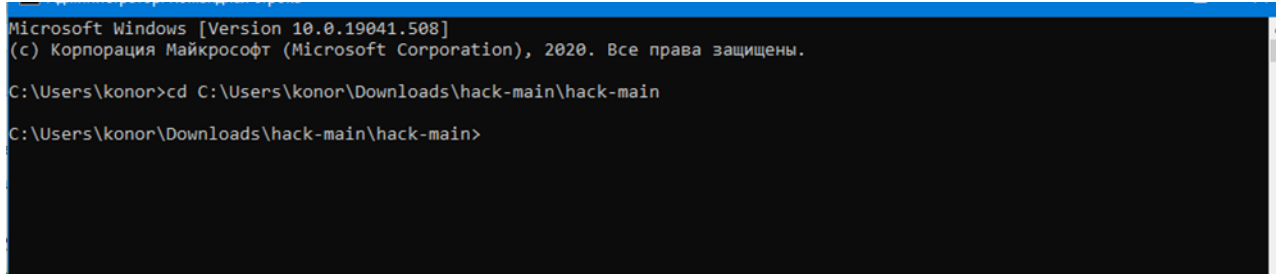
Копіюємо шлях до нашого коду.

У моєму випадку він буде таким

```
C:\Users\konor\Downloads\hack-main\hack-main
```

та пишемо команду

```
cd C:\Users\konor\Downloads\hack-main\hack-main
```

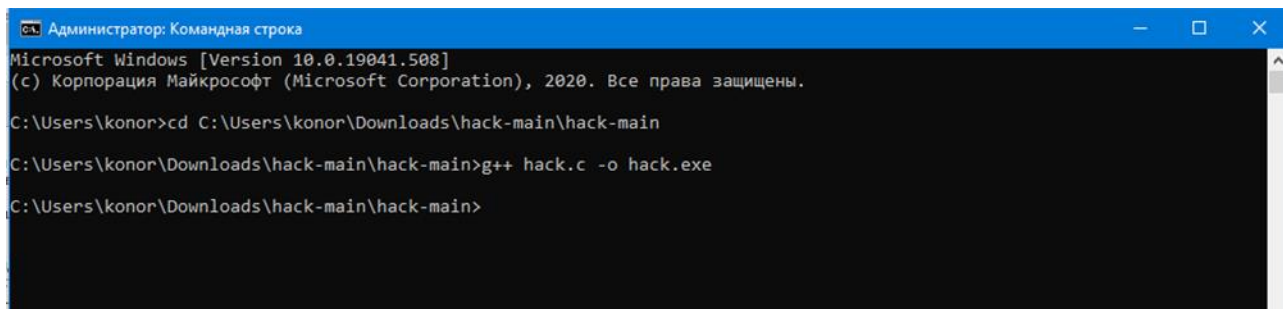


```
Microsoft Windows [Version 10.0.19041.508]
(c) Корпорация Майкрософт (Microsoft Corporation), 2020. Все права защищены.

C:\Users\konor>cd C:\Users\konor\Downloads\hack-main\hack-main

C:\Users\konor\Downloads\hack-main\hack-main>
```

Далі пишемо команду `g++ hack.c -o hack.exe`



```
Администратор: Командная строка

Microsoft Windows [Version 10.0.19041.508]
(c) Корпорация Майкрософт (Microsoft Corporation), 2020. Все права защищены.

C:\Users\konor>cd C:\Users\konor\Downloads\hack-main\hack-main

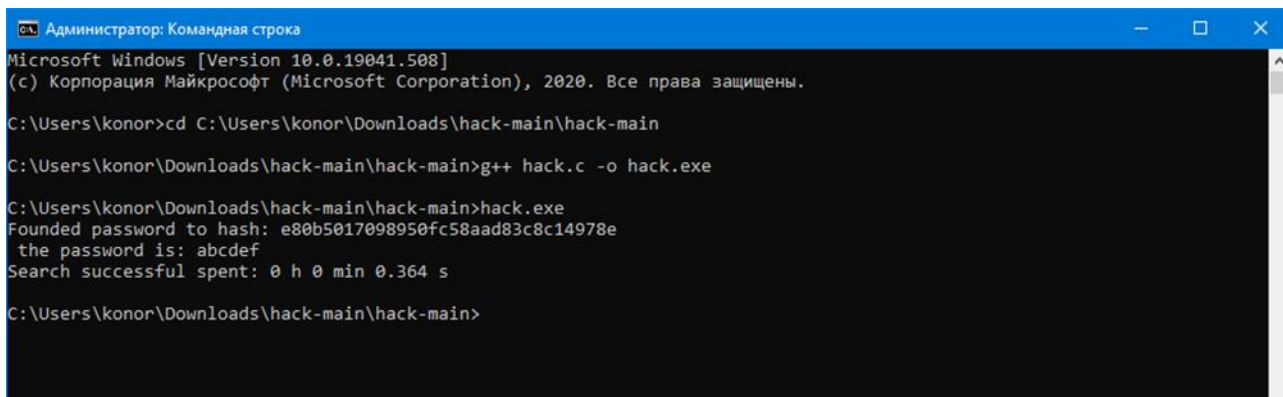
C:\Users\konor\Downloads\hack-main\hack-main>g++ hack.c -o hack.exe

C:\Users\konor\Downloads\hack-main\hack-main>
```

Та запускаємо програму командою

```
hack.exe
```

Ми повинні побачити приблизно такий результат. Час виконання буде відрізнятись в залежності від характеристик комп'ютера.



```
Администратор: Командная строка

Microsoft Windows [Version 10.0.19041.508]
(c) Корпорация Майкрософт (Microsoft Corporation), 2020. Все права защищены.

C:\Users\konor>cd C:\Users\konor\Downloads\hack-main\hack-main

C:\Users\konor\Downloads\hack-main\hack-main>g++ hack.c -o hack.exe

C:\Users\konor\Downloads\hack-main\hack-main>hack.exe
Founded password to hash: e80b5017098950fc58aad83c8c14978e
the password is: abcdef
Search successful spent: 0 h 0 min 0.364 s

C:\Users\konor\Downloads\hack-main\hack-main>
```

Отже, переходимо до самого цікавого.

Відкриємо файл `hack.c` звичайним текстовим редактором, це може бути звичайний блокнот.

Я використовую `notepad++` проте вам його не обов'язково встановлювати.

Ми побачимо цей код:

```
1  #include <iostream>
2  #include <ctime>
3  #include "md5.h"
4
5  using namespace std;
6
7  int scrollSymbol(int index, string& word, const string& drum) {
8      if (index > word.length() - 1 || index < 0) {
9          return -1;
10     }
11     int indexNextChar = drum.find(word[index]) + 1;
12     if (indexNextChar < drum.length()) {
13         word[index] = drum[indexNextChar];
14         return 0;
15     }
16     else {
17         word[index] = drum[0];
18         return scrollSymbol(index - 1, word, drum);
19     }
20 }
21
22 int main()
23 {
24     int passwordLen = 6;
25     bool shorterAlso = 1;
26     string charsToUse = "abcdef\\'\"?";
27     string targetHash = "e80b5017098950fc58aad83c8c14978e";
28     int showEachNPassword = 1000;
29
30     MD5 md5 ;
31     char* hashWord;
32     int charsCount = charsToUse.length();
33     int i = 0;
34     if (!shorterAlso) {
```

Трішки детальніше про нього.

У 24 рядку є змінна за назвою `passwordLen` – це очікувана довжина паролю. В нашому випадку вона зараз рівна 6. Це означає що ми шукаємо пароль довжиною до 6 символів.

Ви можете встановити своє значення, у будь-якому випадку очікувані значення є цілі додатні числа від 1 до 2147483647 проте не рекомендуємо виставляти значення більше 20.

У 25 рядку є змінна за назвою `shorterAlso` – це вказівка на те чи потрібно розглядати коротші паролі. Вона встановлена як 1 – це означає що ми очікуємо що пароль може бути коротший і потрібно розглянути і короткі паролі також.

Наприклад якщо ми виставимо цю змінну в 0, то ми будемо шукати пароль довжиною тільки в зазначену кількість символів (у нас це 6). Коли встановлено 1 то будуть розглядатись такі паролі як «a», «b»,..., «f»,..., «aa», «ab»,..., «aaaaaa», «aaaaab», ..., «ffffff»,...

У 26 рядку є змінна за назвою `charsToUse` – це перелік символів, які будуть використовуватись для перебору. Тут ми можемо записати символи які ми очікуємо в паролі, хочу зазначити що літера «b» та літера «B» хоч однакова буква, проте для



комп'ютера це різні символи і потрібно зазначати обидва, якщо ви хочете розглядати великі та маленькі літери.

Ще один важливий момент, символи «\», «'»,«"»,«?» - особливі символи для C++ які використовуються для керування програмою, якщо ви бажаєте розглянути їх, вам потрібно ставити два символи які будуть означати тільки один символ. Тобто використовуємо: «\\» щоб позначити друкований символ «\» , «\'» для позначення «'», «\"» для «"» та «\?» для «?».

Отже, рядок « string charsToUse = "abcdef\\\\"?"; » в програмі означає що будуть розглядатись 10 символів (a, b, c, d, e, f, \, ', ", ?).

Рядок « string charsToUse = "B"; » тільки символи «B», а « string charsToUse = "\""; » тільки символи «"», та « string charsToUse = "\\\""; » тільки символи «\»,«"»

У 27 рядку є змінна за назвою targetHash – це хеш який ми хочемо розшифрувати. Просто замініть вміст лапок на інший хеш, щоб шукати інший пароль.

У 28 рядку є змінна за назвою showEachNPassword – у ній вказується, який по порядку пароль відображати. Тобто вам в командний рядок буде виводитись який пароль зараз намагається попробувати програма, це необхідно лише для того щоб візуалізувати процес підбирання паролю. Якщо встановити змінну рівну 1 то будуть показані всі паролі, які пробуються, при 1000 тільки кожен 1000-й який пробується. Це не впливає на перебрані паролі, а керує тільки відображенням чим зараз займається машина. Процес виведення зображення на екран досить клопітка робота і чим частіше ми просимо виводити текст тим довше буде підбиратись пароль. Отже при значені 1 вам буде показаний кожен пароль який пробується, при значені 1000 тільки кожен 1000-й. Щоб не показувати зовсім поставте 0. Буде виведено тільки результат роботи програми.

Якщо ви встановили якісь невірні значення і програма «зависла» натисніть комбінацію CTRL+C – це зупинить роботу програми.

Після того як Ви змінили параметри програми збережіть зміни в файлі, та знову виконайте команди

```
g++  hack.c -o hack.exe
hack.exe
```

Спробуйте відновити паролі з наявних хешів:

```
21c2e59531c8710156d34a3c30ac81d5
c4ca4238a0b923820dcc509a6f75849b
92a870e23eaac7b3c576e91b807f2a60
5bc8a34031b373314c039c28e04fd8da
d077f244def8a70e5ea758bd8352fcd8
67e719cf8070862aa3af5121302fea69
```

893b56e3cfe153fb770a120b83bac20c  
999a06f28346262ef9e1699d9fc1ab56  
ff356e2e1d2843d48ac3e85a731620af  
25d55ad283aa400af464c76d713c07ad

### Завдання Б для звіту

У звіті про виконання **завдання Б** цієї лабораторної роботи потрібно надати **відтворений пароль** до одного із 50 хешів із таблиці.

Номер варіанту у звіті визначається номером Вашої залікової книжки:

- якщо дві останні цифри номера залікової  $xx \leq 49$  то вибираєте варіант **xx**.
- якщо дві останні цифри номера залікової  $xx = 50$  то вибираєте варіант **0**.
- якщо дві останні цифри номера залікової  $xx \geq 51$ , проведіть арифметичну дію  $yy = xx - 50$  і виберіть варіант **yy**.

Тобто студент із номером залікової книжки 2119124С у звіті представляє пароль до хеша під номером 24, а студент із номером залікової книжки 3119086С – хеш 36.

Номер варіанту	Хеш MD5
0.	5eb252d099f784590511ed099047a6ec
1.	0b30f446f76f01b6e162ac5e8f31c145
2.	0580eac0ecb7f503a57e5d70113d08f9
3.	184174c3cdb913d01dbcf5a59c232d56
4.	7eabb7c77ae4b44231878e8196cfd221
5.	d94054d83c1572799c1b23212285f933
6.	9fbb3fdc741698faf5a3b3dc5442a14d
7.	cc741eb44c220c6f1d2cbdeb4ef4a71b
8.	62df3c909a70e185a2158bd5d15a4adb
9.	a3b34d86f7b624d9cf34060d12f173f8
10.	977440c6c6607bdafcafcc5d31be682a
11.	66d83c11ace1f9b7c2a37d8e468cf796
12.	6a09c4679c57d0fbd2fe5724ab4ac4d3
13.	1ea17e1ed45a86cd766da25befc0b0b4
14.	13e59908b11e8b9c74c25f4c701e2fce
15.	a7586cfe113f48697f2d0b960739e7e6
16.	427691e0c1d5f5a9d44be3a47de69d04
17.	27811fa9e7c57c5ecfa2a6532ebc4817
18.	a7d1c3f8d20c80dc30c33cfbd2edac9a
19.	66ed4d12fe2e86dd099931b5f271f4bf
20.	985cfd0663944d9fa9897d0f915c4c9d
21.	f2c7074b9167aab7894abce04dad18c2
22.	41620a80d2f3ff6f51ce1794c83bc10c
23.	079bb3776eb77c2dba9540d412e5bd5f

24.	862506d25386c6fd60ee165904982a8b
25.	a1526a39242e120fb4efe081a5b87297
26.	90ec962657099a0ca6d8243a8b1abfe4
27.	442da28b50b23706fe23450ebd89ff8f
28.	ea0e4c493b6c2fefe8c0d272e5e41764
29.	a8ad5654d87c81221cb1326f04f33a19
30.	9c77060cb86a8e2e370f6029b72e1d48
31.	0abd0e654c5cfb62d0bc1e77e38c8a31
32.	e99dfbaf42d92b047f6b3c40f4e02cc5
33.	6b16bb4c4dc63ae2cc20855b4c74e617
34.	0deb2566dc743b58d4ea752b74ed48d9
35.	e40dc5bcc01269ad5f929866f547e9b9
36.	d70b5f3e7b97e6d15284d083e5bcb691
37.	c2cef5742a3f583bab2b0e30184ad2dc
38.	82390c45825d0c03e46c4692d451a1e6
39.	5e51bc276b434b4dd29380353c9f0da7
40.	6772fe770dd13d77f4caee774506e7c7
41.	5144113504167d03d2854a68b8f50ee6
42.	d17410ecfb69b98bdd71aa46001d0418
43.	0ebc23a6eb421ed58402bf21cb2b9e5f
44.	9871927171f7a278f864cacfe43ac816
45.	831ff7a8ea836a45246c8b5b9d30ef1b
46.	94d3365e554d25c32c6e405e1d31c749
47.	b0f81d4044642ce625254bc4b2bb11a9
48.	5ec9115658d981fe14ca350084fd2c20
49.	e5d79598baf0a8adc797f3486dfa4c2f