Politechnika Śląska

Wydział Matematyk Stosowanej

Kierunek Informatyka

Gliwice, 01.10.2020

Programowanie I

**projekt zaliczeniowy**

**"*Password manager*"**

**Dawid Grygiel gr. 1 lab. 1**

**1. Opis projektu.**

Projekt jest programem do zarządzania hasłami poprzez zaszyfrowany zapis i odczyt haseł w plikach tekstowych. Generuje i wczytuje też klucz niezbędny do szyfrowania tekstu. Ma też funkcjonalność wygenerowania bezpiecznego hasła które można skopiować. Kod źródłowy jest dostosowany pod unix’owe systemy operacyjne jak i system Windows.

**2. Wymagania**

*Program przy uruchomieniu automatycznie wyświetla menu z opcjami. Po wybraniu i zakończeniu opcji program znowu wyświetli menu jeżeli nie będzie to opcja zakończ program.*

1. **Wyświetl hasła** – Przy tej opcji program wyświetla strukturę katalogów w folderze „passwd” który został wcześniej utworzony. W przypadku unix’owych systemów operacyjnych niebędna będzie instalacje programu „tree”.
2. **Stwórz nowe hasło** - Program przy tej opcji podejmie próbę stworzenia struktury hasła. Wymagane jest wcześniejsze załadowanie klucza szyfrującego. Jeżeli klucz jest załadowany, to program poprosi użykownika o strukturę i treść hasła, po czym wytworzy odpowiednie katalogi z plikiem zawierającym zaszyfrowane hasło.
3. **Odczytaj hasło** - Program przy tej opcji podejmie próbę odczytania hasła z pliku przy wcześniejszym poproszeniu użytkownika o strukturę hasła. Wymagane jest wcześniejsze załadowanie klucza szyfrującego.
4. **Wygeneruj bezpieczne hasło** – Program po wczytaniu przez użytkownika długości hasła wygeneruje bezpieczne hasło.
5. **Załaduj klucz szyfrujący** – Program przy tej opcji poprosi użytkownika o ścieżkę do pliku tekstowego z kluczem szyfrującym. Po podaniu ścieżki program załaduje dane niezbędne do szyfrowania tekstu.
6. **Stwórz klucz szyfrujący** – Program przy tej opcji wygeneruje klucz z danymi niezbędnymi do szyfrowania tekstu i zapisze go do pliku klucz.txt w katalogu gdzie znajduje się program wykonywalny .exe
7. **Zakończ program** – Kończy program

**3. Przebieg realizacji**

- crypto.cpp, crypto.h – plik zawierający klasę z funkcjami niezbędnymi do szyfrowania tesktu (funkcje matematyczne, funkcje szyfrujące itd).

- path.cpp, path.h – plik zawierający klasę z funkcjami które służą do tworzenia struktur haseł.

- menu.cpp, menu.h – plik z klasą która obsługuje całe menu jak i pojedyńcze opcje programu.

- main.cpp – plik główny który tworzy katalog passwd jeżeli nie istnieje oraz inicjuje menu.

W pierwszej kolejności stworzyłem obieky z obsługą menu, czyli wypisywanie i czyszczenie terminala. Odrazu tworzyłem ten projekt z zamysłem odpalenia tego kodu na wielu systemach operacyjnych.

Później stworzyłem moduł szyfrujący poprzez tworzenie poszczególnych funkcji matematycznych i konwertujących tekst na zaszyfrowany algorytmem podobnym do RSA.  
Algorytm: *1. Wygeneruj liczby pierwsze p i q gdzie q > p*

*2. oblicz n = p \* q*

*3.wylicz funkcję eulera fi(n) = (p - 1) \* (q - 1)*

*4.wygeneruj liczbę względnie pierwszą e której wartośc należy do przydziału (1, fi(n))*

*5. wylicz d =*

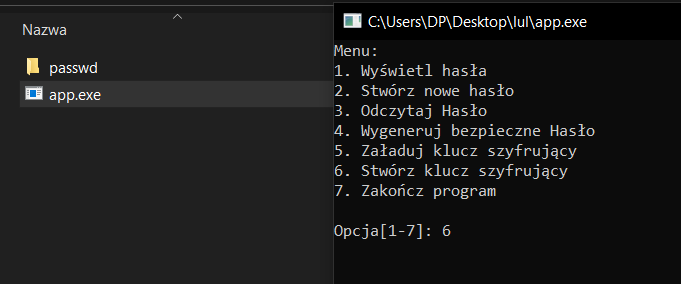
*6. By zaszyfrować teskt każdą literę należy zamienić na pasującą liczbę A z tablicy ASCI i wstawić ją do funkcji i tę nową liczbę zamienić na literę z tablicy ASCI*

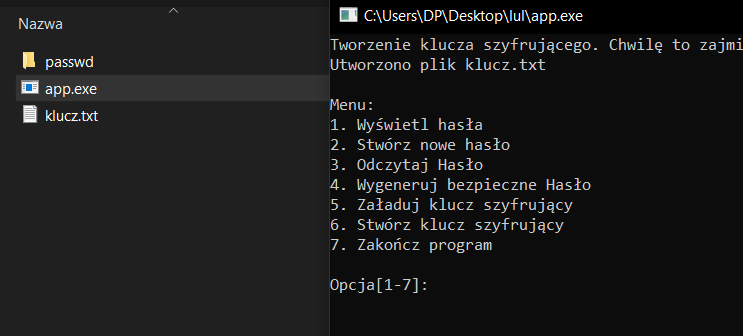
*7. . By odszyfrować teskt każdą literę należy zamienić na pasującą liczbę C z tablicy ASCI i wstawić ją do funkcji i tę nową liczbę zamienić na literę z tablicy ASCI*

Następnie zająłem się funkcjonalnością odczytywania haseł po podaniu przez użytkownika ścieżki (od passwd) do pliku z hasłem. W międzyczasie powoli wyszukiwałem możliwości tworzenia struktur katalogów w C++. Kiedy mogłem już odczytywać hasła wykorzystałem bibliotekę std::filesystem do zarządzania systemami plików i katalogów zaimplementowaną w 2017 roku. Dzięki niej stworzyłem system tworzenia struktór katalogów podanych przez użytkownika. W ostatnim kroku przerobiłem lekko kod i zdefiniowałem stałe niezbędne do możliwości korzystania z kodu na wielu systemach operacyjnych.

**4. Instrukcja użytkownika**

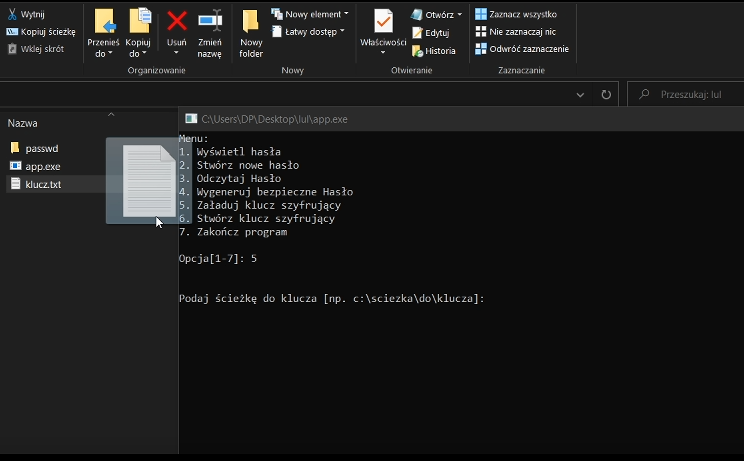
*Po odpaleniu programu wyskakuje menu dzięki któremu łatwiej jest nam dowiedzieć się jak uruchomić poszczególne opcje:  
I. Stworzenie klucza:  
 By stworzyć klucz szyfrujący w menu należy wybrać opcję 6. Wtedy w katalogu gdzie znajduje się palikacja powstanie plik klucz.txt. Jeżeli klucz.txt już istnieje w folderze z app.exe program zapyta się ciebie czy chcesz go nadpisać.*

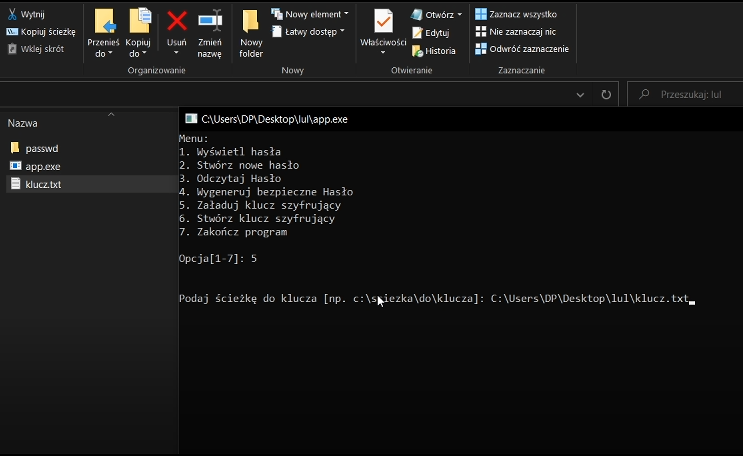
**

**

*II. Załadowywanie klucza szyfrującego:*

*By załadować klucz należy wybrać opcję 5. Program wtedy zapyta się nas o ścieżkę do klucza szyfrującego (dzięki temu można przetrzymywać klucz na bezpiecznym dysku zewnętrznym). Terminal windowsa pozwala nam na przeniesienie pliku z eksploratora pliku bezpośrednio do programu co przyczyni się do wpisania ścieżki do pliku automatycznie*

**

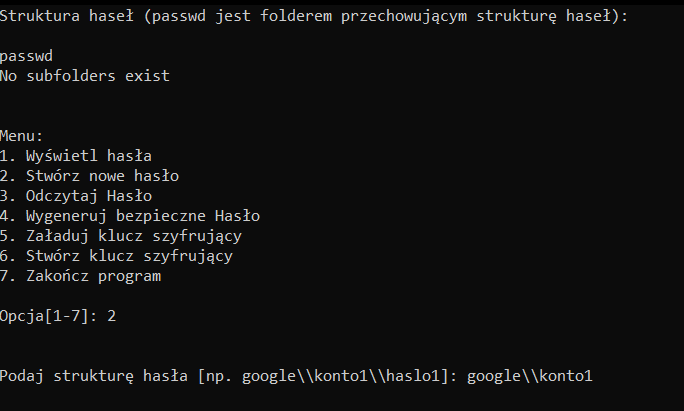
**

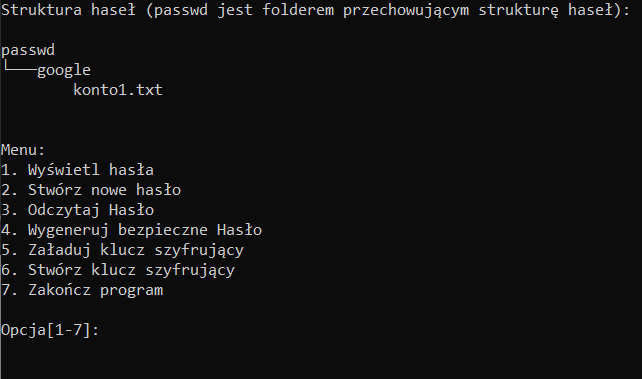
*III. Wyświetlanie haseł*

*By wyświetlić hasła należy w menu wybrać opcję 1. Po tym wyświetli się struktura haseł w folderze passwd. Jeżeli nie stworzyłeś żadnych haseł wyświetli się sam folder passwd sugerujący, że jest pusty.*

*IV.Tworzenie haseł.*

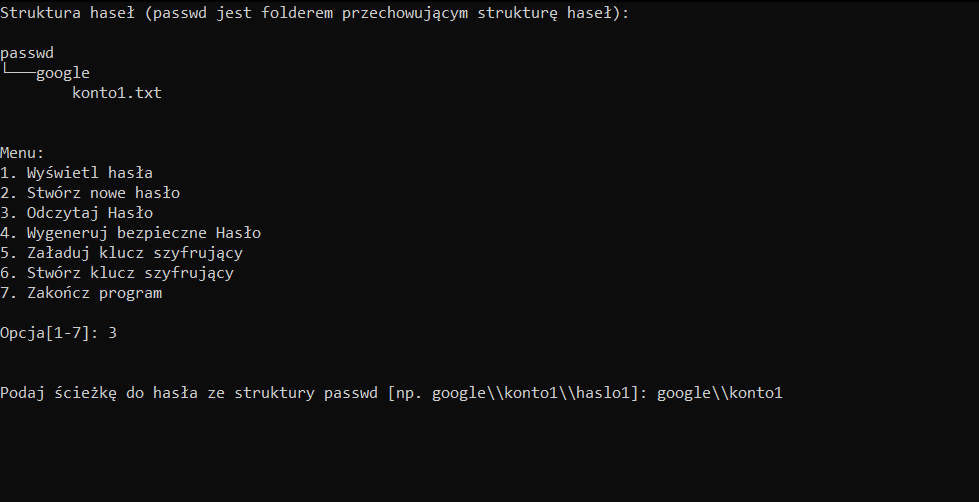
*By stworzyć hasło w menu należy wybrać opcję 2. Należy mieć załadowany klucz szyfrujący (punkt II). Jeżeli już jest załadowany, to program poprosi cię o podanie struktury hasła. Chodzi tu o podanie informacji w jakich podfolderach powinno być umieszczone hasło. Jeżeli nie chcemy by hasło było w jakimkolwiek podfolderze umieszczone należy wpisać tylko nazwę pliku jakie to hasło ma mieć (nie trzeba wpisywać .txt na samym końcu). Jednak te podfoldery są pomocne by rozpoznawać serwisy z jakich te hasła pochodzą np. google/konto1 (oczywiście format ścieżek zależy od systemu operacyjnego. Obok prośby o podanie ścieżki powinna być informacja jak ona ma wyglądać). Po podaniu ścieżki jeżeli wszystko jest wporząku, to program poprosi o podanie hasła jakie ma zapisać w pliku.*

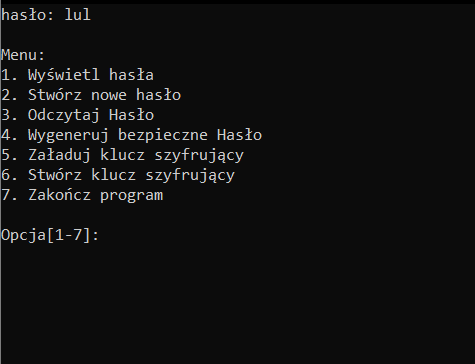
**

**

*V. Odczytywanie hasła*

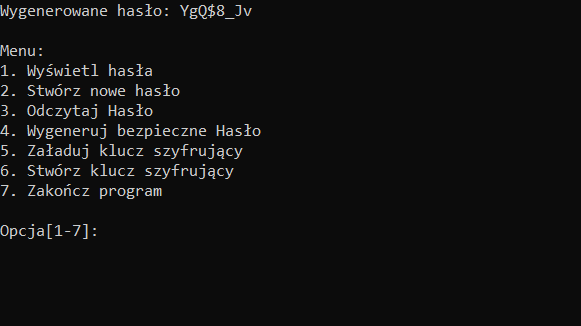
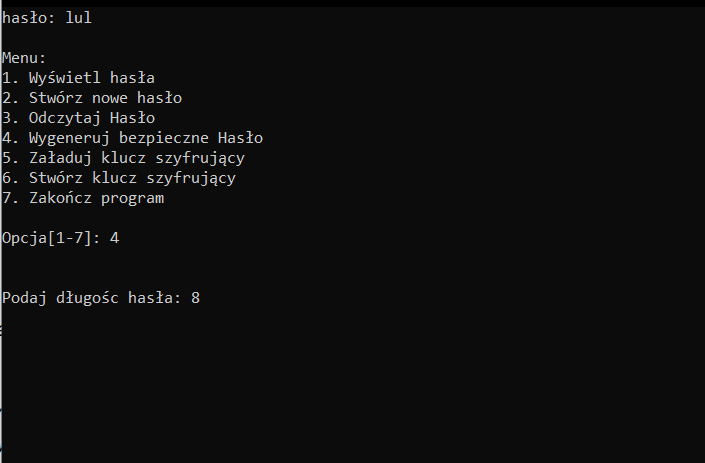
*By odczytać hasło w menu należy wybrać opcję 2. Należy mieć załadowany klucz szyfrujący (punkt II). Po wybraniu opcji program poprosi cię o podanie ścieżki do hasła w folderze passwd. Jak przy tworzeniu hasła, tu by wybrać plik należy wpisać ścieżkę do odpowiednich folderów prowadzących do pliku tekstowego z hasłem. Rekomendowane jest najpierw wyświetlić hasła (wybrać opcję 1) by wiedzieć jaką strukturę trzeba podać. Oczywiście format ścieżek zależy od systemu operacyjnego. Obok prośby o podanie ścieżki powinna być informacja jak ona ma wyglądać.*

**

**

*VI. Generowanie bezpiecznego hasła*

*By wygenerować bezpieczne hasło należy wybrać opcję 4. Po wybraniu program zapyta się ciebie o długość hasła (które musi być dłuższe niż 5 znaków). Po wybraniu wyświetli się wygenerowane hasło i menu poniżej. Takowe hasło można skopiować i użyć przy tworzeniu noewgo hasła.*

**

**5. Podsumowanie i wnioski.**

*W projekcie zrealizowałem szyfrowanie haseł, tworzenie katalogów i na ogół przyjemne korzystanie z programu. Chciałem dać możliwość odczytywania haseł poprzez interaktywne menu które polegało by na wyświelaniu i kolorowaniu jednego pliku/katalogu i przemieszczanie się po tym menu za pomocą strzełek. Za pomocą przycisku enter można by było wchodzić w poszczególne foldery jak i wybierać pliki tekstowe z hasłami. Okazało się jednak, że to niemożliwe na systemach unix’owych poniewać biblioteka która miałaby odczytywać pojedyńcze znaki z klawiatury przerabia całkowicie sposób w jaki wyświetlane są znaki przez co tekst wchodził sam na siebie i wymagałoby to przerabiania całkowicie tego kodu wraz z edytowaniem obiektu cout (nie wspominając, że na windows’ie też musiałoby to działać innaczej). Chciałem też zaimplementować bibliotekę z lepszym szyfrowaniem danych lecz niestety nie udało mi się znaleść/zrozumieć w jaki sposób można byłoby to zrobić na każdym z systemów operacyjnych.*