PJC - Projekt

Gra Hexagon

Tomasz Werner
Filip Kwiatkowski
Andrzej Abramowski
Agnieszka Szołucha

1 Opis projektu

Nie powinniśmy stosować tego samego hasła do różnych portali. Wyciek jednego z nich naraża nas na utratę danych w wielu miejscach. W związku z tym, aby zminimalizować to niebezpieczeństwo powinniśmy stosować 2FA (które nie jest częścią tego projektu) oraz różne hasła. W tym drugim mogą nam pomóc programy typu Password Manager takie jak na przykład KeePass.

Projekt polega na stworzeniu aplikacji konsolowej służącej do modyfikowania i czytania z pliku, w którym będą przechowywane hasła wraz z dodatkowymi informacjami dotyczącymi różnych portali. Będą one zaszyfrowane pojedynczym hasłem głównym.

Aplikacja winna być uruchamiana z poziomu linii poleceń (ang. command line). Bazując na komendach czytanych z klawiatury, wykonywać odpowiednie funkcje i zwracać interesujące nas informacje.

2 Wymagania funkcjonalne

Po uruchomieniu programu użytkownik ma możliwość wybrania jednego z plików znajdują- cych się w folderze programu lub ma możliwość podania bezpośredniej, absolutnej ścieżki do plik

Dane w pliku są zaszyfrowane. Sposób szyfrowania musi być autorski i w pełni rozumiany. Hasło ma stanowić nieodzowną cześć procesu szyfrowania i odszyfrowania. Trudność złamania takiego szyfru nie będzie oceniana, ale otwarcie pliku w edytorze tekstu i proste metody dedukcyjne nie powinny być wystarczające do odszyfrowania jego zawartości. To samo się tyczy modyfikacji tej zawartości.

Każda próba odszyfrowania pliku powinna zapisać timestamp takiej próby. Z uwagi na to, że sam plik nie przechowuje nigdzie informacji o poprawnym haśle, a timestamp musi być zawsze zmieniany (zarówno podczas "nieudanej" próby otwarcia i modyfikacji pliku) to timestamp będzie jedyną informacją zapisaną jawnie w całym pliku. W związku z tym musimy znaleźć inny sposób na jego ukrycie.

Jedną z możliwości byłoby rozłożyć tę informacje na różne linijki. Początkiem

linii 11. mogłoby być hhDDDD, linii 22. mmDDDD, a 33. ssDDDD, gdzie hh to godzina mm to minuta a ss to sekunda ostatniego odszyfrowania. DDDD to dalsze zaszyfrowane dane, niemiejące nic wspólnego z samym timestampem.

Każde hasło musi zawierać co najmniej:

- Nazwę (Nazwa własna tego wpisu np. "Hasło do Konta1 na Google");
- Tekst reprezentujący samo hasło;
- Kategorię.

Każde hasło dodatkowo może zawierać:

- Strona WWW/ Serwis;
- Login.

Implementacja tego elementu jest wymagana, choć nie każde hasło musi zawierać te składowe. Mają one być opcjonalne. Opis komend:
1. Wyszukaj hasła – zwraca hasła, które zawierają konkretne parametry.
2. Posortuj hasła – zwraca posortowaną listę wszystkich haseł. Ma umożliwiać posortowanie po co najmniej 2 różnych parametrach w tym samym czasie, czyli na przykład po nazwie i kategorii.
3. Dodaj hasało – dodaje nowe hasło do zaszyfrowanego pliku. Powinniśmy tu dać użytkownikowi możliwość wpisania własnego hasła i poinformować go na ile jest to bezpieczne hasło i czy nie zostało już wcześniej wykorzystane. Dodatkowo należy zaproponować mu hasło automatycznie wygenerowane dając mu jednocześnie możliwość wybrania pewnych parametrów takich jak:

- Ilość znaków;
- Czy ma zawierać wielkie i małe litery;
- Czy ma zawierać znaki specjalne.

4. Edytuj hasło – pozwala na edycje danych w istniejącym już haśle.
5. Usuń hasło – usuwa wybrane hasło lub hasła. Przed każdym usunięciem powinniśmy powiadomić o tym użytkownika szczególnie jeżeli usuwane jest więcej niż jedno hasło.
6. Dodaj kategorie – dodaje nową kategorię, którą będziemy mogli wykorzystywać przy tworzeniu nowych haseł.
7. Usuń kategorie – usuwa kategorie wraz ze wszystkimi hasłami, które do tej kategorii są przypisane.

3 Kryteria oceniania

Wymaganie	Liczba punktów	Komentarz
Odpowiednie dobieranie i wykorzystywanie narzędzi programistycznych. Poprawne rozdzielenie kodu aplikacji na wiele niezależnych od siebie modu- łów.	5	Przykładowo używanie standardowych algorytmów zgodnie z ich przeznaczeniem, używanie szablonów zamiast makr, odpowiednie dobieranie kontenerów zgodnie z ich przeznaczeniem. Podział implementacji na wiele plików, stworzenie odpowiednich przestrzeni nazw czy klas. Oczywiście w ramach rozsądku – nic na siłę.
Stosowanie się do const-correctness.	1	Oznacza to stosowanie const nie tylko wszędzie tam, gdzie można, ale też wszędzie tam, gdzie koncepcyjnie dany element nie powinien być zmieniany.
Unikanie niepotrzebnych kopii danych w programie.	1	Zgodnie z wiedzą przedstawioną na ćwiczeniach. Tyczy się to typów większych niż prymitywy – wskazanym jest kopiowanie zmiennych typu int czy double zamiast przekazywać je przez const& (oczywiście w przypadku niemodyfikowanych kopii oznaczenie ich przez const jest wciąż wymagane).
Sporządzenie dokumentacji zgodnej z uproszczonym standardem Doxygen.	4	

Klarowność komunikatów błędów przy niepoprawnym użytkowaniu aplikacji.	3	
Poprawnie działające szyfrowanie i odszyfrowanie pliku.	7	
Poprawnie działająca obsługa timestampów.	5	
Poprawna implementacja komendy nr 1.	2	
Poprawna implementacja komendy nr 2.	3	
Poprawna implementacja komendy nr 3.	5	
Poprawna implementacja komendy nr 4.	2	
Poprawna implementacja komendy nr 5.	2	
Poprawna implementacja komendy nr 6.	1	
Poprawna implementacja komendy nr 7.	3	
Przejrzystość menu i łatwość pracy z programem.	6	Program ma charakteryzować się łatwą i intuicyjną strukturą nawigacji, ja snym przekazem informacji i przejrzy stą strukturą treści.