FRANKFURT UNIVERSITY OF APPLIED SCIENCES

Einführung in die Programmierung mit C – WS24/25

11. Übung

Dateien & Kommandozeilenargumente

Biemann Nikbakhsh Gibietz

27. Januar 2025

Aufgabe 1

Erstellen Sie ein Programm zum Ver- und Entschlüsseln von Dateien mit einem frei wählbaren Passwort.

Diese Aufgabe ist komplex und übersteigt im zeitlichen Umfang eine Klausuraufgabe, allerdings nicht inhaltlich vom Schwierigkeitsgrad. Daher ist es eine wunderbare Klausurvorbereitung, um verschiedene Themenbereich der Klausur zu üben. Die einzelnen Problemstellungen sind als Unteraufgaben gekennzeichnet. Versuchen Sie iterativ vorzugehen, indem Sie immer eine Funktionalität nach der anderen implementieren und einzeln testen. Wie bei allen Aufgaben; nutzen Sie keine Musterlösungen, sondern testen Sie Ihren eigenen Kenntnisstand darüber, dass Sie selbst versuchen auf eine eigene Lösung zu kommen. Im Anschluss können Sie Ihre Lösung(en) mit denen Ihrer Kommilitonen vergleichen.

a) Sie können sich eine geeignete (sichere?) Ver- und Entschlüsselungsmethode selbst ausdenken. Alternativ können Sie folgenden Algorithmus verwenden:

1. Beginne mit dem ersten Byte der Quelldatei und dem ersten Zeichen des Passworts. Das Byte wird um dem ASCII-Wert des ersten Passwortzeichens erhöht bzw. vermindert (je nach Ver- oder Entschlüsselung).

2. Fahren Sie mit dem zweiten Byte der Quelldatei und dem zweiten Zeichen des Passworts analog fort.

3. Verfahren Sie analog zu Punkt 1 und Punkt 2 für alle weiteren Zeichen. Ist das letzte Zeichen des Passworts erreicht, so beginnen Sie erneut beim ersten Zeichen des Passworts.

4. Die veränderten Bytes werden in die Zieldatei geschrieben.

Beispiel: Die Quelldatei beinhaltet die Bytefolge 0, 1, 2, 3, 4, 5. Das Passwort lautet „Test“ (ASCII-Werte somit 84, 101, 115, 116). Die verschlüsselte Bytefolge lautet folglich 84, 102, 117, 119, 88, 106.

b) Die Optionen, was Ihr Programm tun soll bzw. welche Dateien verwendet werden, sollen per Kommandozeilenargumente übergeben werden. Ihr Programm soll folgende Argumente entgegen nehmen können:

- --encrypt Ihr Programm geht in den Modus Verschlüsselung über.
- -e Kurzschreibweise für --encrypt
- --decrypt Ihr Programm geht in den Modus Entschlüsselung über.
- -d Kurzschreibweise für --decrypt
- --output Das darauf folgende Argument gibt die Zieldatei an.
- -o Kurzschreibweise für --output
- Dateiname der Quelldatei

Beachten Sie bitte in Ihrem Programm, dass sich die Optionen Ver- und Entschlüsselung gegenseitig ausschließen und dass das Argument --output bzw. -o eine Pflichtangabe ist. Entsprechende Probleme bei der Argumentenliste sollen dem Nutzer als Fehlermeldung mitgeteilt werden.

Wie üblich soll die Reihenfolge der Optionsargumente vom Nutzer beliebig gewählt werden können; selbstverständlich mit der Einschränkung, dass hinter -o bzw. --output der Zieldateiname folgt.

c) Das Passwort zur Ver- und Entschlüsselung soll während der Laufzeit mit einer geeigneten Eingabeaufforderung eingegeben werden. Um eine versehentlich falsche Passworteingabe zu verhindern, soll das Passwort beim Verschlüsseln zweimal eingegeben und beide Eingaben miteinander verglichen werden. Wird nicht zweimal das selbe Passwort eingegeben, so soll das Programm mit einer passenden Fehlermeldung beendet werden.

In der Bibliothek `unistd.h` wird Ihnen unter anderem für Unix-basierte Systeme die Funktion `getpass(char* eingabeaufforderung)` zur Verfügung gestellt. Mit dieser Funktion können Sie eine „blinde Texteingabe“[1] beispielsweise für eine Passworteingabe realisieren. Diese Funktion liefert einen Pointer auf einen Speicherplatz zurück, wo das Passwort hinterlegt ist. Achtung: Bei erneutem Aufruf dieser Funktion wird derselbe Speicherplatz verwendet und die alte Zeichenkette hinter diesem Pointer gelöscht.

[1] Die eingegebenen Zeichen werden nicht im Terminal angezeigt.

Um also zwei damit eingegeben Passwörter zu vergleichen, müssen Sie mindestens die erste Eingabe in einen anderen Speicherplatz kopieren. Nutzen Sie diese Funktion zur Passworteingabe, damit das benutzte Passwort aus Sicherheitsgründen *niemals* angezeigt wird.

d) Bedenken Sie bitte, dass die Quell- und somit auch Zieldatei von beliebiger Größe sein können, Ihrem Programm aber nur ein begrenzter Speicherplatz zur Verfügung steht; soll heißen: Die zu ver- oder entschlüsselnden Daten können den Umfang des zur Verfügung stehenden Speicherbereichs übersteigen. Überlegen Sie sich daher wie Sie es programmieren!

Ihr Programm muss nicht effizient sein, es reicht Effektivität.