

Junioraufgabe 1: Passwörter

J1.1 Lösungsidee

Die Aufgabenstellung verlangt drei Regeln, nach denen gut aussprechbare und merkbare Passwörter erzeugt werden sollen. Diese Forderung wird hier vorsichtig umgangen; es wird vor allem ein Konstruktionssystem entwickelt, dass zur gewünschten Art von Passwörtern führt.

Die wichtigste Regel für ein gut auszusprechendes Passwort besagt, dass das Wort dem Klang der deutschen Sprache entsprechen soll. Diese Regel ist sehr abstrakt und erfordert eine detaillierte Ausarbeitung, die weitere Regeln eher implizit statt explizit mit einschließt. Bei Betrachtung eines längeren Wortes wie „Som·mer·fe·ri·en“ kann man leicht erkennen, dass ein Wort aus Silben besteht, die jeweils ähnlich aufgebaut sind und als eine Einheit ausgesprochen werden. Silben sind jeweils kurz und haben in diesem Beispiel eine der drei Grundstrukturen

- CV: Konsonant + Vokal
 - fe, ri
- VC: Vokal + Konsonant
 - en
- CVC: Konsonant + Vokal + Konsonant
 - Som, mer

Diese Silbenstrukturen kann man im weiteren Sinne als „Regeln“ im Sinne der Aufgabenstellung verstehen. Es gibt durchaus weitere Strukturen und Sonderfälle, die in der Umsetzung erläutert werden. Insbesondere wird zusätzlich eine „Silbenstruktur“ N verwendet, die für eine ein- bis zweistellige Zahl steht.

Die Idee besteht nun darin, zuerst eine zufällige Anzahl von Silben zu wählen. Für jede der Silben wird eine zufällige Struktur gewählt. Anschließend werden die Silbenstrukturen nacheinander durch zufällige Werte aus ihrem jeweiligen Bereich ersetzt, also wird z. B. für ein „Konsonant“ ein „n“ eingesetzt. Abschließend wird das generierte Passwort ausgegeben.

Ein Beispiel: Das Passwort soll aus drei Silben bestehen, als Strukturen werden CVC, N und CV gewählt. Daraus entsteht ganz zufällig das Passwort ha142da.

Doch kennt die deutsche Sprache nicht auch komplexere Silbenstrukturen wie CVCC (Kopf, Dach), CCVC (Stab, Blut) oder gar CCCCCVCC (Schnitt)? Die Frage ist nicht leicht zu beantworten, schließlich beschäftigen sich ganze Fachgebiete (Phonologie, Phonotaktik) damit. Um es kurz zusammenzufassen: Man kann selbst für einfache Fälle wie CCV nicht beliebige Buchstaben einsetzen. Wenn man als ersten Buchstaben ein *m* wählt, so kann im Deutschen kein zweiter Konsonant darauf folgen. Es gibt unzählige Regeln dafür, welche Laute innerhalb einer Silbe wie angeordnet werden können, und sie benötigen letztlich ein komplettes Wörterbuch mit erlaubten Kombinationen. Wir wollen uns daher auf die oben genannten Silbenstrukturen beschränken. Diese sind auf jeden Fall mit allen einsetzbaren Werten aussprechbar. Aber solche Wörterbuch-Methoden sind als Lösung für diese Aufgabe natürlich genausogut denkbar!

Auf das Generieren von Großbuchstaben und Sonderzeichen wird verzichtet, da sie für den Benutzer außerhalb von bekannten Begriffen eher schwierig zu merken sind.

J1.2 Umsetzung

Das zu erstellende Programm benötigt keine Benutzerinteraktion, außer zum Starten. Zuerst wird eine zufällige Anzahl an Silben im Bereich 5-7 ermittelt (**GenerateRandomValues**). Diese Grenzen wurden gewählt, da ein eigener Test gezeigt hat, dass zufällige Wörter ab 8 Silben und damit etwa 20 Buchstaben doch sehr schwer zu merken sind. Bei weniger als 4 Silben ist dass Passwort jedoch sehr kurz und daher weniger sicher.

Für jede der Silben wird anschließend eine Struktur aus den vier verfügbaren Strukturen (CV, VC, CVC, N) zufällig ermittelt (**GenerateSyllableStructure**). Hierbei steht **C** für engl. *consonant*, **V** für *vowel* und **N** für *number*.

Bis zu dieser Stelle hat der Algorithmus eine Silbenstruktur ermittelt, beispielsweise CVC-CV-CV-N-CVC. Falls benachbarte N auftreten, wird das zweite N durch CV ersetzt, damit keine Zahlenkolonnen entstehen können. Anschließend werden die Elemente der Struktur durch Phoneme aus der jeweiligen Klasse ersetzt (**GenerateLetters**). *Phoneme* sind einzelne hörbare Laute, die in der Schrift auch durch mehr als ein Zeichen dargestellt werden können. So gehören zu den Vokal-Phonemen neben a, e, i, o, u auch ie, ei, ey, au und eu. Bei den Konsonant-Phonemen gehören u. a. auch ch und sch dazu. Das c wurde absichtlich weggelassen, das es im Deutschen nicht alleine stehen kann, sondern nur in den sogenannten Clustern ch und ck, die wiederum teilweise nicht am Anfang einer Silbe stehen können. Ein N-Strukturelement wird durch eine Zahl im Bereich von 1 bis 99 ersetzt.

Nachdem alle Elemente ersetzt wurden, wird das Ergebnis dem Benutzer angezeigt.

Speicherverbrauch und Laufzeit

Der vorgestellte Algorithmus hat keine expliziten Parameter. Implizit ist lediglich die Anzahl an zu verwendenden Silben für das Passwort vorgegeben, die beispielhaft im Bereich 5-7 gewählt wurde. Speicherplatz und Laufzeit hängen jeweils linear davon ab, also $O(n)$.

J1.3 Shannon-Entropie und die Sicherheit von Passwörtern

Dieser Abschnitt beschäftigt sich etwas tiefer mit der Sicherheit und dem Maß für Sicherheit von Passwörtern. Eine solche theoretische Analyse war für die Bearbeitung der Aufgabe nicht gefordert, aber wir wollen sie interessierten Lesern nicht vorenthalten.

Schauen wir uns zunächst einmal das Passwort vom Anfang der Aufgabenstellung an. Es besteht aus 11 Zeichen, enthält Groß- und Kleinbuchstaben, Ziffern, und Sonderzeichen:

Dfq4T&o%%kM

Wie sicher ist dieses Passwort? Wenn wir davon ausgehen, dass dieses Passwort generiert wurde, indem 11 mal gleichverteilt zufällig aus der Menge $\{A, \dots, Z, a, \dots, z, 0, \dots, 9, &, %, \dots\}$, also einer Menge mit 72 Zeichen (vereinfachend 10 Sonderzeichen angenommen) gewählt wurde, dann ist dieses Passwort eines aus

$$72^{11} \approx 2,7 \cdot 10^{20}$$

möglichen und gleichverteilten Passwörtern.

Die Anzahl an möglichen (gleichverteilten!) Passwörtern ist ein gutes anschauliches Maß für die Stärke eines Passwortes. Je größer die Anzahl, desto mehr Passwörter müssen *im Durchschnitt* ausprobiert werden, um das Passwort zu erraten – und umso kleiner ist die Wahrscheinlichkeit, dass Passwort innerhalb einer bestimmten Anzahl an Versuchen zu erraten (sei es, weil man z. B. nur 3 Versuche hat, oder sei es, weil man ab einer bestimmten Anzahl einfach Jahre braucht, um alle durchzuprobieren).

Ein anderes und mathematisch etwas flexibleres Maß für die Stärke eines Passworts ist die Shannon-Entropie¹. Sie gibt einfach gesagt an, wie vielen rein zufälligen Bits die Zufälligkeit eines Passwortes entspricht. Ein Vorteil der Shannon-Entropie ist, dass sie linear in der Länge des Passwortes ist. Ein einziges zufälliges Zeichen aus der oben genannten Menge mit 72 Zeichen hat eine Shannon-Entropie $H_{\text{Zeichen}} \approx 6,16$ Bit.² Ein Passwort aus 11 zufälligen Zeichen hat also eine Shannon-Entropie von

$$H_{11 \text{ Zeichen}} \approx 67,9 \text{ Bit}$$

Was ist nun die Entropie einer unserer Silben aus der Lösungsidee? Mit einer Wahrscheinlichkeit von $\frac{1}{4}$ wählen wir je

- a) eine Ziffer zwischen 0 und 99 (100 Möglichkeiten),
- b) einen Konsonanten und einen Vokal (210 Möglichkeiten)³,
- c) einen Vokal und einen Konsonanten (wieder 210 Möglichkeiten) oder
- d) einen Konsonanten, einen Vokal und einen Konsonanten (4410 Möglichkeiten).

Damit erhalten wir für eine Silbe, die nach diesem Schema generiert wurde, eine Shannon-Entropie von:⁴

$$H_{\text{Silbe}} \approx 10,54 \text{ Bit}$$

Um ein Passwort zu erhalten, das etwa genauso sicher ist wie das ursprüngliche Passwort, müssten wir also ein Passwort mit sechs bis sieben Silben generieren:

$$H_{7 \text{ Silben}} = 7 \cdot H_{\text{Silbe}} \approx 73,8 \text{ Bit}$$

Das ist schon einigermaßen lang, aber immer noch viel leichter zu merken als 11 rein zufällige Zeichen.

Moderne Empfehlungen zum Generieren sicherer Passwörter gehen inzwischen dazu über, Passwörter nicht mehr aus Silben zu bilden, damit sie lediglich *aussprechbar* werden, sondern sie stattdessen gleich aus ganzen Wörtern zu bilden. Wichtig ist dabei aber, dass die Wörter nicht von Menschen ausgewählt, sondern wirklich zufällig aus einer großen Liste von Wörtern gewählt werden. Damit lassen sich bei gleichbleibender Entropie wesentlich leichter zu merkende Passwörter generieren. Das liegt daran, dass sich aus den Wörtern oft Geschichten bilden lassen – oftmals absurde und lustige, aber dadurch auch besonders gut zu merken! Besonders verbreitete Bekanntheit hat dieses Verfahren mit einem Cartoon von Randall Munroe erlangt,

¹[https://de.wikipedia.org/wiki/Entropie_\(Informationstheorie\)](https://de.wikipedia.org/wiki/Entropie_(Informationstheorie))

² $H_{\text{Zeichen}} = -\sum_{i=1}^{72} \frac{1}{72} \log_2 \frac{1}{72} = -\log_2 \frac{1}{72}$

³Wir gehen in dieser Berechnung von 10 Vokal-Phonemen und 21 Konsonanten aus, s. Abschnitt J1.2.

⁴
$$\begin{aligned} H_{\text{Silbe}} &= -\sum_{i=1}^{100} \frac{1}{4 \cdot 100} \log_2 \frac{1}{4 \cdot 100} - \sum_{i=1}^{210} \frac{1}{4 \cdot 210} \log_2 \frac{1}{4 \cdot 210} - \sum_{i=1}^{210} \frac{1}{4 \cdot 210} \log_2 \frac{1}{4 \cdot 210} - \sum_{i=1}^{4410} \frac{1}{4 \cdot 4410} \log_2 \frac{1}{4 \cdot 4410} \\ &= -\frac{1}{4} (\log_2 \frac{1}{4 \cdot 100} + \log_2 \frac{1}{4 \cdot 210} + \log_2 \frac{1}{4 \cdot 210} + \log_2 \frac{1}{4 \cdot 4410}) \end{aligned}$$

der in Abbildung J1.1 abgedruckt ist. Auch hier verwendet Munroe die Shannon-Entropie als wesentliches Maß für die Stärke von Passwörtern.

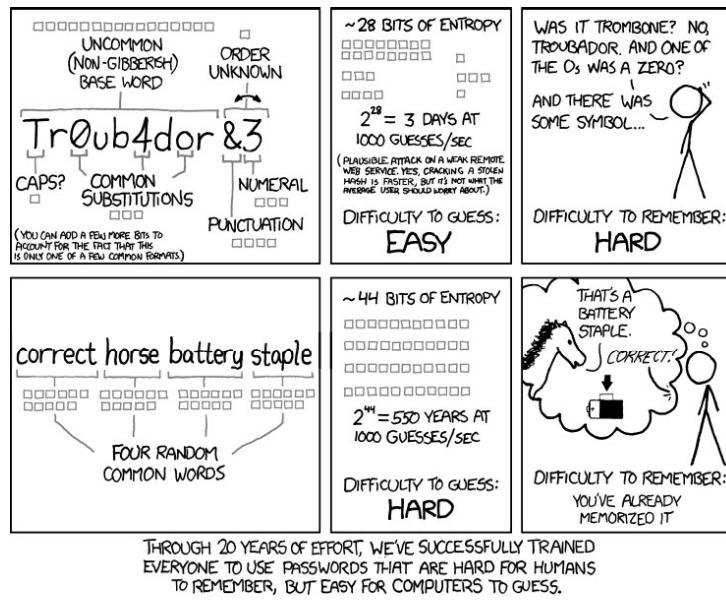


Abbildung J1.1: Cartoon von Randall Munroe zur Sicherheit von Passwörtern. Heruntergeladen von <https://xkcd.com/936/>. Titeltext für Browser: „To anyone who understands information theory and security and is in an infuriating argument with someone who does not (possibly involving mixed case), I sincerely apologize.“

J1.4 Beispiele

Programmablauf

Im Normalfall zeigt das Programm keine Zwischenschritte an. Beispielhaft wird daher ein Blick auf die interne Verarbeitung geworfen.

Im ersten Schritt wird die Anzahl an Silben (**length**) auf 5 gesetzt.

Die Silbentypen (**syllableTypes**) sind: 3, 2, 3, 1, 1

Damit ist die Silbenstruktur (**passwordStructure**): N, CVC, N, CV, CV

Nach **GenerateLetters** ergibt sich daraus: 65rod3fusei

Die Struktur N wurde ein Mal durch eine Ziffer und ein anderes Mal durch zwei Ziffern ersetzt, ebenso das letzte V durch den Doppelaut **ei**. Das erzeugte Passwort kann man durchaus aussprechen.

Ausgaben

Hier einige subjektiv lustige Programmausgaben, jeweils mit ihrer Zerlegung in Silben und der zugehörigen Silbenstruktur.

- bieariefbuschhaschu
bie-ar-ief-busch-ha-schu
CV -VC- VC-CVC -CV- CV

- titlideitujeieidquie
tit-li-dei-tu-jei-eid-quie
CVC-CV-CV -CV-CV - VC- CV
- du98mehubschir8quu
du-98-me-hub-schir-8-quu
CV-N -CV-CVC- CVC-N-C V
- nussiemfusdimienviewiel
nus-siem-fus-di-mien-vie-wiel
CVC-CV C-CVC-CV-CV C-CV -CV C
- mameworielpeg
ma-me-wo-riel-peg
CV-CV-CV-CV C-CVC
- kenschu28teyllalof
ken-schu-28-teyl-la-lof
CVC- CV-N -CV C-CV-CVC

J1.5 Bewertungskriterien

Die Bewertungskriterien vom Bewertungsbogen werden hier erläutert (Punktabzug in []).

- **[−1] Passwörter nicht gut auszusprechen**

Ob explizite Regeln oder ein Erzeugungssystem mit impliziten Regeln wie in dieser Beispieldlösung: Die Aufgabenstellung fordert auf jeden Fall, dass die Passwörter „einigermaßen gut auszusprechen und deshalb leichter zu merken sind“. Ob diese Forderung erfüllt ist, mag Geschmackssache sein. Ein rein zufällige Aneinanderreihung von Zeichen ist aber sicher nicht akzeptabel. Auch andere Verfahren, die sich ungenügend am Aussprechen orientieren, können zu Punktabzug führen. Es soll zumindest einigermaßen sinnvoll begründet sein, warum und wie das Verfahren zur Erzeugung von Passwörtern sich an der Aussprechbarkeit orientiert.

- **[−1] Passwörter zu leicht zu erraten**

Die Aufgabenstellung fordert auch, dass die Passwörter schwer zu erraten sind. Werden regelmäßig Passwörter erzeugt, die aus weniger als 10 Zeichen bestehen, ist diese Forderung nicht erfüllt. Wenn die Länge des Passworts im Programm frei einstellbar ist, ist das in Ordnung; es sei denn, in der Dokumentation werden zu kurze Passwörter präsentiert. Eine Diskussion der Passwortstärke war nicht erwartet.

- **[−1] Implementierung fehlerhaft oder unnötig aufwendig**

Die Implementierung sollte die Regeln oder das Erzeugungsverfahren wie beschrieben umsetzen. Die Generierung der Passwörter sollte nicht unnötig kompliziert realisiert werden.

- **[−1] Beispiele fehlerhaft bzw. zu wenige oder ungeeignete Beispiele**

Es sollten mindestens drei erzeugte Passwörter angegeben werden.