

## 8.7.F Sonstiges/Dateiformate – Ergänzungen und Bilder

### 8.7.F.1 Aufgabe: Dateiformate mit Trennzeichen

In Unix-ähnlichen Systemen (Linux, MacOS, Android) liegen wichtige Benutzerdaten in der Textdatei mit dem Pfadnamen `/etc/passwd`. In dieser Datei ist jedem Benutzer eine Zeile (= ein Datensatz) zugeordnet. Innerhalb der Zeile sind die einzelnen Informationen (= Felder oder Komponenten) voneinander durch Doppelpunkte getrennt. Hier ist ein Auszug aus einer solchen Datei (`src/etc_passwd.txt`):

```

1 root:x:0:0:root:/root:/bin/bash
2 www-data:x:33:33:www-data:/var/www:/bin/sh
3 heinz:x:1001:100:Heinz Meier:/home/heinz:/bin/bash
4 fritz:x:1002:100:Fritz Lehmann:/home/fritz:/bin/bash
5 nobody:x:65534:65534:nobody:/nonexistent:/bin/sh
```

Die Felder sind wie folgt festgelegt:

- A) Loginname
- B) Verschlüsseltes Passwort (x bedeutet: Passwort ist an anderer Stelle abgelegt)
- C) User ID, die Nummer des Benutzers. Der Administrator (root) hat die Nummer 0.
- D) Group ID, die Nummer der Gruppe, zu der der Benutzer gehört
- E) Vollständiger Name des Benutzers
- F) Persönliches Verzeichnis des Benutzers
- G) Kommando-Interpreter für den Benutzer

fkapitelAufgabenteile

`liespw0.c` Arbeiten Sie zunächst mit der Zeichenkette:

```
char zk[]="www-data:x:33:33:www-data:/var/www:/bin/sh";
```

Entwickeln Sie mit Hilfe von `strtok()` eine Funktion `p=str_holen1(s,n)`, die Ihnen aus der Zeichenkette `s` den `n`-ten Teilstring herausholt und die Startadresse an `p` übergibt.

`liespw1.c` Leider zerstört `strtok()` die Zeichenkette unwiederbringlich. Deswegen sollen Sie nun eine neue Funktion `p=str_holen2(s,n)` schreiben. Diese Funktion soll Ihnen genauso den `n`-ten Teilstring liefern. Sie soll allerdings kein `strtok()` enthalten. Stattdessen können Sie das Trennzeichen mit `strchr()` suchen. Oder sie können von Hand den String durchsuchen oder eine andere Funktion nehmen. Damit Sie die originale Zeichenkette nicht zerstören müssen, soll das Ergebnis in eine andere Variable kopiert werden. Dazu benutzen Sie bitte die Funktion `malloc()`. Vergessen Sie nicht, das Ergebnis zu terminieren.

`liespw2.c` Lesen Sie die Datei zeilenweise ein (wobei Sie von einer Zeilenlänge ausgehen, die den Wert 800 nur im Fehlerfall überschreitet) und geben Sie sie einfach sofort wieder aus!

`liespw3.c` Nun soll der Benutzer vorher angeben können, welches Feld jeder Zeile er angezeigt haben möchte (optional: er kann mehrere Felder angeben). Danach wird jede Zeile eingelesen, sofort das richtige Feld (oder mehrere davon) geholt und ausgegeben.