Aussagenlogik

In diesem Praktikum werden Sie mit bitweisen Boole'schen Operatoren arbeiten. Dabei greifen Sie auf einzelne Bits oder Bitgruppen eines größeren Datentyps zu. Außerdem überprüfen und setzen Sie einzelne Bits. Je nach Implementierung werden Sie auch die Shift-Operatoren verwenden. Darüber hinaus werden Sie Oktalzahlen einlesen, Eingaben in genau spezifiziertem Format verarbeiten, Teile daraus extrahieren und schließlich Text ausgeben. Als Anwendungsfall dienen Dateizugriffsrechte unter UNIX-artigen Betriebssystemen.

I. Hintergrund

Das Betriebssystem auf Ihrem Computer kontrolliert unter anderem den Zugriff auf Dateien: Je nachdem welche Rechte einem User-Account eingeräumt wurden, können Dateien gelesen oder geschrieben und Programme ausgeführt werden. Diese Rechte können einzeln oder in beliebiger Kombination eingeräumt oder entzogen werden: Es gibt Dateien, die Sie lesen und in die Sie Inhalt schreiben können; es gibt Dateien, die Sie nur lesen dürfen; es gibt Dateien, die nur Sie lesen dürfen; es gibt Dateien, auf die Sie überhaupt keinen Zugriff haben usw.

I.1. Wer darf zugreifen?

In UNIX-artigen Betriebssystemen – also u.a. Linux und macOS – mit den sogenannten *POSIX file system permissions* können diese Rechte für drei „Klassen" von User-Accounts festgelegt werden: Für einen einzigen User-Account (*owner user*), für eine einzige Gruppe von User-Accounts (*owner group*) sowie für alle übrigen Accounts (*all* oder *others*). Wer *owner user* und welches die *owner group* ist, kann individuell für jede Datei einzeln festgelegt werden. User-Accounts können einer beliebigen Anzahl von Gruppen zugeteilt werden.
Die Rechte werden folgendermaßen angewandt: Möchte der *owner-user*-Account auf eine Datei zugreifen, gelten die *owner-user*-Rechte. Handelt es sich nicht um den *owner-user*-Account, wird geprüft, ob der betreffende Account Mitglied der *owner group* ist. Falls ja, gelten die Rechte der *owner group*. Falls auch das nicht zutrifft, gelten für den Zugriff die Rechte für die übrigen Accounts. Mehr Informationen dazu erhalten Sie z.B. in der englischsprachigen Wikipedia[1]. Ein umfangreicheres Verfahren zur Rechtevergabe stellen sogenannte *Access Controls Lists* dar. Diese sind allerdings nicht Thema in diesem Praktikum.

I.2. Wie werden die Rechte gespeichert?

Die einzelnen Rechte werden als *Bit-Flags* gespeichert. Für jede Zugriffsklasse (*owner user*, *owner group*, *others*) gibt es die drei Rechte `r` (*read* / lesen), `w` (*write* / schreiben) und `x` (*execute* / ausführen). Insgesamt werden 9 Bit benötigt. Diese sind wie in Abbildung 1 im Speicher angeordnet. Dementsprechend lassen sich die Zugriffsrechte als dreistellige Oktalzahl angeben. Diese lässt sich in einer 16 Bit breiten Ganzzahl ablegen – selbst wenn nur 9 Bit benötigt werden. Dazu eignet sich z.B. der Datentyp `uint16_t` aus dem Header `stdint.h`[2].

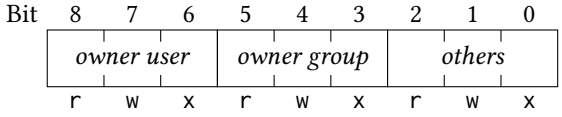

Abbildung 1: Anordnung der einzelnen Zugriffs-Bits im Speicher.

I.3. Wie werden die Rechte dargestellt

Neben der Angabe als Oktalzahl können die Rechte auch als Zeichenfolge ausgegeben werden. Dabei stehen die ersten drei Zeichen für die *owner-user*-Rechte, die nächsten drei Zeichen für die *owner-group*-Rechte, die letzten drei Zeichen für die *others*-Rechte. Die Rechte werden in der jeweiligen Klasse in der Reihenfolge `r`, `w`, `x` angegeben. Ist ein Recht eingeräumt, wird der entsprechende Buchstabe ausgegeben. Ist ein Recht nicht eingeräumt, wird statt des Buchstabens ein `-` (Bindestrich) ausgegeben. Beispiele finden Sie in Tabelle 1.

[1] `https://en.m.wikipedia.org/wiki/File-system_permissions`
[2] `https://en.cppreference.com/w/c/types/integer`

Tabelle 1: Beispiele für Rechte als Oktalzahl und als Zeichenfolge

Oktalzahl	Text	Bedeutung
755	`rwxr-xr-x`	*owner user* darf lesen, schreiben, ausführen, *owner group* darf lesen und ausführen, *others* dürfen lesen und ausführen.
750	`rwxr-x---`	*owner user* darf lesen, schreiben, ausführen, *owner group* darf lesen und ausführen, *others* haben keinen Zugriff.
644	`rw-r--r--`	*owner user* darf lesen und schreiben, *owner group* darf lesen, *others* dürfen lesen.

I.4. Wie werden die Rechte zugewiesen?

Die Zugriffsrechte werden in der Kommandozeile (unter Linux und macOS) mit dem Befehl `chmod`[3] (für ***ch**ange file **mod**es*) festgelegt. Die Dateieigenschaften *owner user* bzw. *owner group* werden in der Kommandozeile mit dem Befehl `chown`[4] (für ***ch**ange **own**ership*) festgelegt.

II. Aufgaben

1. Dateirechte ausgeben

Implementieren Sie eine Konsolenanwendung, die Eingaben der folgenden Form verarbeiten kann:

```
NUM_FILE_LINES
MODE1 FILENAME1
...
MODEn FILENAMEn
```

Dabei steht

- `NUM_FILE_LINES` für die Anzahl (`n`) der auf die erste Zeile folgenden Zeilen mit Datei-Infos,
- `MODEi` für die Zugriffsrechte der `i`-ten Datei als Oktalzahl ($i \in [1, n]$)
- `FILENAMEi` für den Namen der `i`-ten Datei – maximal 9 Nutzzeichen.

Für jeden eingegelesenen Dateinamen soll Ihr Programm die Zugriffsrechte als Text ausgeben. Beispielsweise soll diese Eingabe

```
4
050 file0.txt
022 file1.txt
351 file2.txt
063 file3.txt
```

folgende Ausgabe erzeugen:

```
---r-x--- file0.txt
----w--w- file1.txt
-wxr-x--x file2.txt
---rw--wx file3.txt
```

[3] `https://man7.org/linux/man-pages/man1/chmod.1p.html`
[4] `https://man7.org/linux/man-pages/man1/chown.1p.html`

2. Dateirechte ändern

Erweitern Sie Ihre Konsolenanwendung, sodass sie Eingaben der folgenden Form verarbeiten kann:

```
NUM_FILE_LINES
MODE1 FILENAME1
...
MODEn FILENAMEn
NUM_CHMOD_LINES
chmod CHANGE1 AFFECTED_FILE1
...
chmod CHANGEm AFFECTED_FILEm
```

Dabei steht nach wie vor

- `NUM_FILE_LINES` für die Anzahl (`n`) der auf die erste Zeile folgenden Zeilen mit Datei-Infos,
- `MODEi` für die Zugriffsrechte der `i`-ten Datei als Oktalzahl ($i \in [1, n]$)
- `FILENAMEi` für den Namen der `i`-ten Datei – maximal 9 Nutzzeichen,

und zusätzlich

- `NUM_CHMOD_LINES` für die Anzahl (`m`) der auf diese Zeile folgenden Zeilen mit Rechte-Änderungen,
- `CHANGEk` für die `k`-te Änderung von Dateirechten ($k \in [1, m]$)
- `AFFECTED_FILEk` für den Namen der von der `k`-ten Änderung betroffenen Datei.

Die Rechte-Änderungen `CHANGEk` liegen dabei in einem der folgenden Formate vor:

- `CLASS+MODE`
- `CLASS-MODE`

Die erste Variante räumt der angegebenen Klasse (`u`: *owner* ***u****ser*, `g`: *owner* ***g****roup*, `o`: ***o****thers*) das angegebene Recht (`r`, `w` oder `x`) ein. Die zweite Variante entzieht der angegeben Klasse das angegebene Recht.

Nach dem Einlesen soll Ihr Programm die aktuellen Zugriffsrechte als Text ausgeben. Dabei sollen alle angegebenen Änderungen berücksichtigt sein. Beispielsweise soll diese Eingabe

```
4
050 file0.txt
022 file1.txt
351 file2.txt
063 file3.txt
6
chmod o+r file1.txt
chmod g-x file0.txt
chmod u+r file1.txt
chmod u+r file0.txt
chmod u+x file1.txt
chmod o-w file1.txt
```

folgende Ausgabe erzeugen:

```
r--r----- file0.txt
r-x-w-r-- file1.txt
-wxr-x--x file2.txt
---rw--wx file3.txt
```