SPiC-Aufgabe #6: trac

(9 Punkte, keine Gruppen)

Schreiben Sie ein Programm `trac` (**tr**anslate **a**rbitrary **c**haracters), das, ähnlich dem UNIX Kommando `tr(1)`, bestimmte Zeichen durch andere ersetzt.

Das Programm wird wie folgt aufgerufen:

```
trac <SET1> <SET2>
```

Hierbei sind `SET1` und `SET2` zwei Strings gleicher Länge. Das Programm liest nun bis zum Erreichen von End-Of-File (EOF) Zeichen von der Standardeingabe. Kommt ein gelesenes Zeichen an der Stelle i im String SET1 vor, so wird statt diesem das Zeichen an der Stelle i im String SET2 ausgegeben. Zeichen, die nicht im String `SET1` enthalten sind, werden unverändert auf die Standardausgabe ausgegeben.

Wird das Programm mit ungültigen Parametern aufgerufen, soll es sich mit einer aussagekräftigen Fehlermeldung auf dem Standardfehlerkanal und einem den Fehler signalisierenden Exitstatus beenden.

Jedes Zeichen darf nur einmal in `SET1` enthalten sein.

Fehler müssen wie immer auf `stderr` ausgegeben werden.

Verwendungsbeispiel:

```
$ ./trac ab ba
babadu                          <- Eingabe
ababdu                          <- Ausgabe

$ echo "Dies ist ein Test" > test.txt
$ cat test.txt | ./trac ie ei
Deis est ien Tist

$ cat test.txt | ./trac ie ei > test_out.txt
$ cat test_out.txt
Deis est ien Tist
```

Hinweise:

- Begründen Sie die Verwendung von allen `volatile` Variablen. Wenn für mehrere Variablen die selbe Begründung gilt, dürfen Sie diese gemeinsam begründen.
- Im Verzeichnis `/proj/i4spic/<login>/pub/aufgabe6/` befindet sich die Datei `trac`, welche eine Beispielimplementierung enthält.
- Achten Sie auf aussagekräftige Fehlermeldungen, die alle auf dem Standardfehlerkanal ausgegeben werden sollen. (`fprintf(stderr,...)(3)` / `perror(3)`)
- Testen Sie Ihr Programm auch mit `valgrind`. Dies kann bei der Suche nach Fehlern helfen. *suppressed Errors* können ignoriert werden. Weitergehende Fehlermeldungen erhalten Sie, wenn Sie `valgrind` mit den Flags `--leak-check=full --show-reachable=yes` aufrufen und das zu analysierende Binary mit Debug-Symbolen bauen.
- Ihr Programm muss mit dem folgendem Aufruf übersetzen:
  `gcc -std=c11 -pedantic -D_XOPEN_SOURCE=700 -Wall -Werror -O3 -o trac trac.c`
  Diese Konfiguration wird zur Bewertung herangezogen.
- Funktionen der `libc`, für die wir keine Fehlerbehandlung erwarten, sind online in der Linux `libc`-Doku entsprechend markiert.
- Sie können ein `Makefile` schreiben, das eine Anleitung für den Bau des Programms mit dem Tool `make` enthält. Hierfür legen Sie eine Datei `Makefile` in Ihrem Aufgabenordner (`aufgabe6/`) an. In die erste Zeile schreiben Sie:
  `CFLAGS = -std=c11 -pedantic -D_XOPEN_SOURCE=700 -Wall -Werror -O3`
  Der Bau erfolgt im Terminal durch `make trac` oder dem `make` Button in der SPiC-IDE.

Abgabezeitpunkt

alle Gruppen	10.01.2021	18:00:00