SYSPROG BEISPIEL 3

Aufgabenstellung

```
SYNOPSIS:
     readin <filename>
     caesar <key>
```

Implementieren Sie zwei Programme `readin` und `caesar`, die einen einfachen (und leicht zu knackenden!) Verschlüsselungsalgorithmus anwenden.
Das Programm `readin` liest zeilenweise aus einem als Argument übergebenen Textfile, und schreibt jede Zeile nacheinander auf folgende Weise modifiziert in ein Shared Memory:

- Alle Großbuchstaben (alle Zeichen in `[A-Z]`) werden unverändert beibehalten.
- Alle Kleinbuchstaben (alle Zeichen in `[a-z]`) werden in Großbuchstaben konvertiert.
- Alle anderen Zeichen fallen weg.

Das Programm `caesar` erwartet als Argument eine Zahl zwischen 1 und 25. Es liest jede Zeile aus dem Shared Memory und ersetzt jeden Buchstaben durch den im Alphabet um `key` weiter rechts stehenden Buchstaben (wobei sich rechts neben dem Z wieder das A befinden soll). Bei einem `key` von 3 wird also z.B. A zu D, B zu E, C zu F, W zu Z, X zu A und Y zu B. Die so verschlüsselte Zeile wird dann auf `stdout` ausgegeben, und die nächste Zeile wird gelesen.
`caesar` soll die nächste Zeile erst dann lesen, wenn `readin` sie vollständig ins Shared Memory geschrieben hat und `readin` soll erst dann die nächste Zeile ins Shared Memory schreiben, wenn die vorherige schon von `caesar` gelesen wurde.
Sobald `readin` die letzte Zeile des Eingabefiles verarbeitet hat, schreibt es einen frei wählbaren String in das Shared Memory, der `caesar` anzeigt, dass alle Arbeit getan ist. Darauf sollen beide Programme terminieren.

Anleitung

Folgende vereinfachenden Einschränkungen gelten:

- Im Eingabefile stehen nur ASCII-Zeichen. Umlaute und dergleichen müssen also nicht behandelt werden.
- Keine Zeile ist länger als 1022 Zeichen (inklusive Newline).

Testen

Testen Sie Ihr Programm mit mehreren Eingabefiles. Erstellen Sie z.B. ein Testfile *t1* mit folgendem Inhalt:

```
Ich habe das dritte Sysprog-Beispiel,
das gar nicht schwer war,
schon fast fertig.
```

Rufen Sie dann die beiden Programme auf (Reihenfolge je nach Ihrer Implementierung):

```
readin t1 &
caesar 15 > t2
```

Dann haben Sie in *t2* die verschlüsselte Version von *t1*.

Mit

```
readin t2 &
caesar 11
```

können Sie diese wieder entschlüsseln, und sollten folgende Ausgabe erhalten:

```
ICHHABEDASZWEITESYSPROGBEISPIEL
DASGARNICHTSCHWERWAR
SCHONFASTFERTIG
```

Bitte beachten Sie auch die Allgemeinen Hinweise zur Beispielgruppe 3 und die Richtlinien für die Erstellung von C-Programmen.