OPERATING SYSTEMS BEISPIEL 3

Aufgabenstellung – 2048

In dieser Aufgabe werden Sie eine Terminal Version des Spiels „2048“ (auch bekannt als „1024“ und „Threes“) implementieren.

Das Spielfeld besteht aus 16 Feldern, die in einem 4x4 Quadrat angeordnet sind. Jedes davon kann entweder leer sein oder einen Wert aus der Menge $\{2^i | i \in \mathbb{N}^+\}$ enthalten.

Benachbarte Felder, welche den selben Wert beinhalten, können kombiniert werden und verschmelzen zu einem Feld mit dem Wert der Summe beider vorherigen Felder. Zum Beispiel ergibt eine Kombination von 2 Feldern mit Wert `32` ein Feld mit Wert `64`.

Ziel des Spiels ist, Felder geschickt zu kombinieren, um Felder mit möglichst hohen Werten entstehen zu lassen. Ein Spiel ist gewonnen sobald ein Feld mit Wert `2048` entsteht (bzw. der Wert, mit dem der Server als Optionsargument gestartet wurde, siehe Abschnitt). Falls das Spielfeld voll ist und keine weiteren Kombinationsmöglichkeiten bestehen, hat der Spieler verloren.

In einer Spielrunde werden folgende Schritte durchlaufen:

1. Ein freies Feld wird zufällig ausgewählt und mit einem neuen Wert befüllt. Dabei soll der Wert `2` mit einer Wahrscheinlichkeit von 0.75 oder der Wert `4` mit einer Wahrscheinlichkeit von 0.25 erzeugt werden.
2. Der Spieler gibt eine Richtung an, in welche alle Felder mit Werten auf dem Spielfeld verschoben werden.
3. Daraufhin wird der angegebene Zug ausgeführt. Im folgenden steht r für die vom Spieler angegebene Richtung. Angefangen auf der Seite r wird nun für jedes Feld f folgende Logik angewandt:
   (a) Solange das Nachbarfeld von f in Richtung r leer ist, verschiebe f dorthin.
   (b) Falls das Nachbarfeld von f in Richtung r den selben Wert hat wie f, entferne f und verdopple den Wert vom Nachbarfeld.
4. Falls das Spielfeld durch Schritt 3 unverändert geblieben ist, setzen Sie mit Schritt 2 fort; andernfalls beginnt eine neue Spielrunde.

Bei Fragen zu den Spielregeln und für weitere Inspiration, orientieren Sie sich am einfachsten an `http://gabrielecirulli.github.io/2048/`.

Anleitung

Das Spiel soll als Client/Server Programm realisiert werden, wobei ein Server beliebig viele Clients (das heißt auch beliebig viele gleichzeitige Spiele) bedient. Kommunikation zwischen Clients und Server soll per Shared Memory erfolgen.

Server

```
USAGE: 2048-server [-p power_of_two]
    -p:     Play until 2^power_of_two is reached (default: 11)
```

Über die Option `-p` kann angegeben werden, wann das Spiel als gewonnen gilt, also bei welcher Zweierpotenz, die als Wert auf einem Feld erreicht wird. Per default beträgt dieser Wert $2^{11} = 2048$.

Spielelogik findet ausnahmslos am Server statt, während Clients lediglich für Darstellung und Benutzereingaben verantwortlich sind. Eine Nachricht zum Client soll also das aktuelle Spielfeld und Status Codes (z.B. `ST_WON`, `ST_LOST`, `ST_NOSUCHGAME`) enthalten, während eine Nachricht zum Server nur Befehle (z.B. `CMD_LEFT`, `CMD_QUIT`) enthält.

Der Server soll eine unbegrenzte Anzahl an gleichzeitigen Spielen verwalten können. Sobald ein Client ein Request für ein neues Spiel sendet, sollen lokal benötigte Ressourcen angelegt werden. Bei Beendigung eines Spiels müssen Sie diese natürlich wieder freigeben.

Clients kommunizieren mit dem Server über Shared Memory. Zugriffe darauf sollen mit Semaphoren synchronisiert werden. Stellen Sie sicher, dass nach einem Request von Client C_i die Antwort unbedingt nur C_i (und nicht einen anderen Client $C_j, j \neq i$) erreicht.

Jedem Spiel wird vom Server eine ID zugeteilt, und Clients können sich entweder zu bestehenden Spielen verbinden oder ein neues erstellen. Insbesondere werden Spiele am Server nur gelöscht falls sie entweder vom Client explizit beendet werden oder verloren sind. Ein Client soll sich also vom Spiel trennen und später wieder zum selben Spiel verbinden können. Sie dürfen annehmen, dass sich zu einem Spiel nie mehr als ein Client gleichzeitig verbindet.

Falls der Server ein `SIGTERM` oder `SIGINT` Signal erhält, sollen alle angelegten Ressourcen (lokal angelegter Speicher, Shared Memory, Semaphoren, etc) korrekt freigegeben und terminiert werden.

Client

```
USAGE: 2048-client [-n | -i <id>]
    -n:     Start a new game
    -i:     Connect to existing game with the given id
```

Der Client ist für Darstellung der Spielfelds und Behandlung von Benutzereingaben zuständig.

Die Gestaltung der Oberfläche ist nicht weiter spezifiziert, außer dass mindestens das Spielfeld, die ID des aktuellen Spiels, und eine kurze Anleitung mit den möglichen Aktionen angezeigt werden müssen.

Mögliche Aktionen sind:

- *Links/Rechts/Oben/Unten*: Angabe der Richtung eines Spielzugs. Taste `a` steht für links, `d` für rechts, `w` für oben, `s` für unten.
- *Spiel Löschen*: Das Spiel wird aufgegeben, Ressourcen werden am Server gelöscht, der Client terminiert.
- *Verbindung Trennen*: Der Client terminiert ohne Spielressourcen am Server zu löschen.

Der Client soll zusätzlich ordnungsgemäß terminieren sobald das Spiel gewonnen oder verloren ist, oder der Server (d.h. das Shared Memory oder die Semaphore) nicht mehr erreichbar ist.

Richtlinien

Bitte beachten Sie auch die *Richtlinien für die Erstellung von C-Programmen* sowie die *Allgemeinen Hinweise zur Beispielgruppe 3* auf der Übungswebsite.