Prof. Franz Schweiggert Abteilung Angewandte Informationsverarbeitung 27. Januar 2003
Christian Ehrhardt Blatt 12

Allgemeine Informatik 3 (WS 2002/2003)

Abgabetermin 10.02.2003

Vier gewinnt (20 Punkte+5 Zusatzpunkte)

Das Spiel

Das Spiel "Vier gewinnt" wird von zwei Spielern auf einem rechteckigen Spielfeld mit 6 Zeilen und 7 Spalten gespielt. Ein Spieler spielt mit weißen Spielsteinen, der andere mit Schwarzen. Ziel des Spiels ist es, 4 Steine der eigenen Farbe in einer Reihe (senkrecht, waagerecht oder diagonal) zu platzieren. Der erste Spieler, der dieses Ziel erreicht gewinnt sofort das Spiel. Dazu dürfen die einzelnen Spieler abwechselnd einen Stein der eignen Farbe auf dem Spielfeld platzieren. Einzige Einschränkung: Ein neuer Spielstein darf nur in das unterste freie Feld einer Spalte gesetzt werden.

Die Spielstände

Zu einem Spielstand gehören also die folgenden Informationen

- Das aktuelle Spielfeld
- Die Farbe des Spielers, der gerade am Zug ist.
- Ist das Spiel bereits beendet?
- Ev. weitere Informationen

Damit läßt sich feststellen, wieviel Platz ein einzelner Spielstand benötigt, wenn man ihn in einer Datei abspeichern will. Weiterhin ist es dann möglich, mehrere Spielstände von verschiedenen Spielen hintereinader in einer einzigen Datei zu speichern. Da jeder eine feste Größe hat, kann ein bestimmter Spielstand mit Hilfe einer Nummer ausgewählt werden.

Ein erstes Programm (15 Punkte)

Nach diesen Vorüberlegungen soll nun ein erstes Programm geschrieben werden, das auf der Kommandozeile die folgenden Informationen mitgeteilt bekommt:

- Die Nummer eines Spielstandes in der Datei (beginnend bei 0).
- Eine der beiden Farben, schwarz oder weiß.

Das Programm soll den durch die Nummer angegebenen Spielstand aus der Datei auslesen und ev. durch einen neuen Spielstand ersetzen. Dabei dürfen andere Spielstände in der selben Datei nicht verändert werden. Der neue Spielstand ergibt sich aus folgenden Regeln:

- Wenn der Spielstand mit der angegebenen Nummer in der Datei noch nicht existiert oder wenn das Spiel bereits beendet ist, soll ein neues Spiel begonnen werden. Das Feld ist in diesem Fall leer und die auf der Kommandozeile angegebene Farbe ist am Zug.
- Wenn bereits ein Spiel läuft und die auf der Kommandozeile angegebene Farbe ist nicht am Zug, dann soll der Spielstand in der Datei unverändert bleiben.
- Sonst soll der Spielstand angezeigt werden. Anschließend ist noch folgendes zu tun:
  - Wenn bereits eine Viererreihe existiert, dann wird ausgegeben, wer das Spiel gewonnen hat. Im neuen Spielstand wird das Spiel als beendet markiert.
  - Ansonsten wird von der Tastatur ein Zug eingelesen und ein neuer Stein in der eigenen Farbe platziert. Die eigene Farbe ist die, die auf der Kommandozeile angegeben wurde. Außerdem wird die Farbe des Spielers, der am Zug ist angepaßt.
  - Falls durch den gesetzten Stein eine neue Viererreihe entstanden ist wird ausgegeben, welche Farbe gewonnen hat, das Spiel wird aber hier noch nicht als beendet markiert.

Named pipes

Mit Hilfe des Systemcalls mknod oder mit der Funktion mkfifo können im Dateisystem sogenannte FIFOs oder named pipes angelegt werden. Eine solche FIFO hat folgende Eigenschaften:

- Alles was in diese Datei geschrieben wird, wird auch in dieser Reihenfolge wieder daraus gelesen und zwar nur genau einmal. D.h., daß die gelesenen Daten aus der Datei verschwinden, sobald sie gelesen wurden.
- Wenn es gerade nichts zu lesen gibt, aber noch jemand die Datei zum Schreiben geöffnet hat, dann blockiert read, bis es wieder Daten gibt.
- Wenn es nichts zu lesen gibt und niemand mehr die Datei zum Schreiben geöffnet hat, dann kehrt read mit dem Rückgabewert 0 zurück. Das bedeutet Dateiende.

Ein richtiges Programm (5 Punkte)

Damit das Spiel vernünftig gespielt werden kann soll in einem nächsten Schritt mit Hilfe von named pipes eine Art Signalmechanismus geschrieben werden. Damit kann dem Programm des Spielpartners mitgeteilt werden, daß er jetzt am Zug ist. Das soll folgendermaßen funktionieren:

- Zu Beginn legt das Programm jetzt eine FIFO im selben Verzeichnis wie die Datei mit den Spielständen an. Der Name der FIFO soll um Verwechslungen zu vermeiden aus der Nummer des Spiels und der Farbe des Spielers bestehen.
- Wenn ein neues Spiel begonnen wird, dann soll das sofort erledigt werden.

- Wenn das Spiel beim Programmstart bereits läuft, dann soll ein einzelnes Byte in die FIFO des Spielpartners geschrieben werden.
- Anschließend sollen wie im folgenden beschrieben Züge eingelesen werden, bis das Spiel beendet ist.
- Bevor ein Zug eingelesen wird, soll ein einzelnes Byte aus der eigenen FIFO gelesen werden.
- Dann wird der eigentliche Zug eingelesen, wie bereits im ersten Programm.
- Nachdem ein veränderter Spielstand geschrieben wurde, soll ein einzelnes Byte in die FIFO des Spielpartners geschrieben werden.
- Wenn nach dem Einlesen eines Bytes aus der FIFO festgestellt wurde, daß auf den anderen Spieler gewartet werden muß weil dieser am Zug ist (sollte normalerweise nicht vorkommen), dann soll kein neuer Zug eingelesen werden. Statt dessen soll nur ein Byte in die FIFO des Spielpartners geschrieben werden.

Da read beim Lesen aus einer leeren FIFO blockiert, wird dadurch sichergestellt, daß immer abwechselnd Züge abgegeben werden, ohne daß beide Spielparteien nachsehen müssen, ob der Gegner inzwischen einen Zug abgegeben hat.
Hinweis: Beide Prozesse sollten die FIFO zum lesen und schreiben öffnen und während der gesamten Laufzeit der Programms geöffnet halten.

Locks (5 Zusatzpunkte)

5 Zusatzpunkte gibt es, wenn das Lesen und Schreiben in der gemeinsamen Datei, die die Spielstände enthält, durch lockf synchronisiert wird.