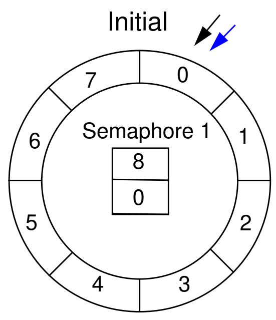
59 Überblick über die 13. Übung

- Besprechung 8. Aufgabe (Erzeuger & Verbraucher)
- Eingabeverarbeitung des Terminals
- RayTracer (Kein Prüfungsstoff)
- Film: Red's Nightmare

59.0 Besprechung 8. Aufgabe (Verbraucher)



Verwaltungsdatenstruktur

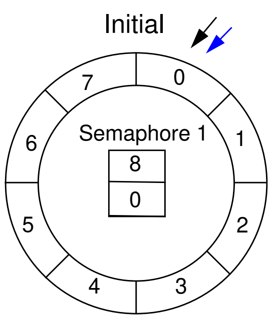
42	PID Verbraucher
0	Leseindex
0	Schreibindex
8	Puffergröße

Semaphore 2

1

- SharedMemory attachen
- Semaphoren attachen
- Semaphore 2 erniedrigen
- PID testen und schreiben
- Semaphore 2 erhöhen

59.0 Besprechung 8. Aufgabe (Erzeuger)



Verwaltungsdatenstruktur

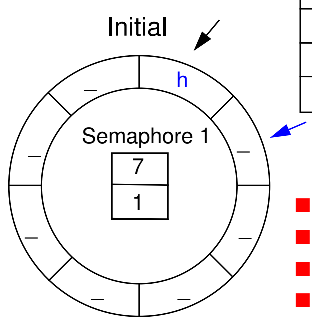
-1	PID Verbraucher
0	Leseindex
0	Schreibindex
8	Puffergröße

Semaphore 2

1

- Semaphore 1 und 2 können auch zu einer Semaphore zusammengefaßt werden
- Semaphoren erst erzeugen nachdem die Verwaltungsdaten initialisiert wurden

59.0 Besprechung 8. Aufgabe (Schreiben)



Verwaltungsdatenstruktur

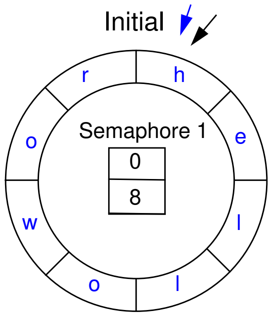
42	PID Verbraucher
0	Leseindex
1	Schreibindex
8	Puffergröße

Semaphore 2

1

- Semaphore 1 Feld 1 erniedrigen
- Zeichen schreiben
- Schreibindex erhöhen
- Semaphore 1 Feld 2 erhöhen

59.0 Besprechung 8. Aufgabe (Schreiben)



Verwaltungsdatenstruktur

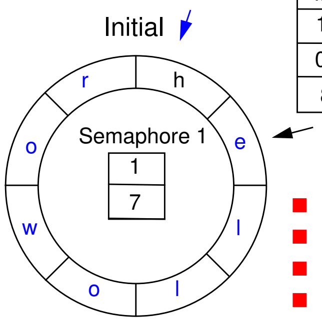
42	PID Verbraucher
0	Leseindex
0	Schreibindex
8	Puffergröße

Semaphore 2

1

- Semaphore 1 Feld 1 erniedrigen
- Zeichen schreiben
- Schreibindex erhöhen
- Semaphore 1 Feld 2 erhöhen

59.0 Besprechung 8. Aufgabe (Lesen)



Verwaltungsdatenstruktur

42	PID Verbraucher
1	Leseindex
0	Schreibindex
8	Puffergröße

Semaphore 2

1

- Semaphore 1 Feld 2 erniedrigen
- Zeichen lesen
- Leseindex erhöhen
- Semaphore 1 Feld 1 erhöhen

59.0 Warum kommen die Zeichen nicht gleich?

- Die Eingabeverarbeitung des virtuellen Terminals gibt die Eingabe nur Zeilenweise an den Erzeuger-Prozess weiter
  - ◆ wenig Kommunikation zwischen Anwendung und Terminal
  - ◆ ermöglicht die Eingabekorrektur durch das Terminal
- Lösung: Terminal auf Einzelzeicheneingabe umstellen.

59.1 Eingabeverarbeitung des Terminals

- Zeilenmodus ("canonical mode"):
  Tastatur-Eingabe wird vom Terminaltreiber vorverarbeitet
  - ➤ Pufferung von Zeilen
  - ➤ Einfache Editier-Funktion (Zeichen oder Zeile löschen)
  - ➤ Erkennen von Sonderzeichen (Signale, EOF)
- Einzelzeichenmodus ("non-canonical mode"):
  Tastatur-Eingabe wird "sofort" an das Programm weitergeleitet
  - ➤ keine Zeilenpufferung
  - ➤ keine Eingabebearbeitung möglich
- Verhalten des Terminaltreibers ist konfigurierbar

59.1 Terminal-Parameter auslesen

```
#include <termios.h>
int tcgetattr(int fildes, struct termios *termios_p);
```

- Argumente
  - ◆ **fildes**: Filedeskriptor der mit einem Terminal verknüpft ist (z.B. STDIN_FILENO)
  - ◆ **termios_p**: Puffer für Terminal-Informationen
- Rückgabewert: 0 wenn OK, -1 wenn Fehler (siehe errno-Variable)

59.1 Terminal-Parameter setzen

```
#include <termios.h>
int tcsetattr(int fildes, int optional_actions,
 const struct termios *termios_p);
```

- Argumente
  - ◆ **fildes**: Filedeskriptor der mit einem Terminal verknüpft ist
  - ◆ **optional_actions**: Modus, wann die Änderungen wirksam werden
    - **TCSANOW**: sofort
    - **TCSADRAIN**: nachdem alle Ausgaben übermittelt wurden
    - **TCSAFLUSH**: wie TCSADRAIN, jedoch werden zusätzlich alle nicht verarbeitete Eingaben verworfen
  - ◆ **termios_p**: Puffer für Terminal-Informationen
- Rückgabewert: 0 wenn mindestens ein Wert gesetzt werden konnte,
  -1 bei Fehlern (siehe errno-Variable)

59.1 Die Struktur `termios`

```
typedef unsigned int tcflag_t;
typedef unsigned char cc_t;
typedef unsigned int speed_t;

struct termios {
        tcflag_t     c_iflag;       /* input modes */
        tcflag_t     c_oflag;       /* output modes */
        tcflag_t     c_cflag;       /* control modes */
        tcflag_t     c_lflag;       /* line discipline modes */
        cc_t         c_cc[NCCS];    /* control chars */
};
```

- ◆ **c_iflag**: Eingabekontrolle (z.B. Behandlung von CR und LF)
- ◆ **c_oflag**: Ausgabekontrolle (z.B. Umsetzung des '\n'-Zeichens)
- ◆ **c_cflag**: Einstellungen der Hardware des Terminals (Baud-Rate)
- ◆ **c_lflag**: allgemeine Terminalfunktionen (z.B. canonical mode)
- ◆ **c_cc**: Kontroll-Zeichen bzw. Terminal-Variablen

59.1 Die Einzelzeicheneingabe

- Aktivieren des “non-canonical” Modus durch Löschen von **ICANON** in **c_lflag**:

```
settings.c_lflag &= ~ICANON;
```

- Eine Lese-Anforderung des Programms wird jedoch erst erfüllt wenn:
  - ◆ mindestens *MIN* Zeichen eingegeben wurden oder
  - ◆ die Zeit zwischen der Eingabe von zwei Zeichen länger ist als *TIME*.
- Der Wert *MIN* wird im Element **VMIN** des Feldes **c_cc** gespeichert, der Wert *TIME* im Element **VTIME**:

```
settings.c_cc[VMIN] = MIN;
settings.c_cc[VTIME] = TIME;
```

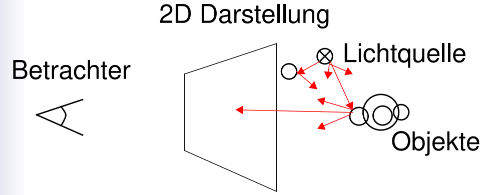
59.1 Beispiel

```
#include <termio.h>
struct termios settings;

if ( tcgetattr(STDIN_FILENO, &settings) == -1 ){
  perror("Fehler bei tcgetattr: "); exit(EXIT_FAILURE);
}

settings.c_lflag &= ~ICANON;
settings.c_cc[VMIN] = 1;
settings.c_cc[VTIME] = 0;

if ( tcsetattr(STDIN_FILENO, TCSANOW, &settings) == -1 ) {
  perror("Fehler bei tcsetattr: "); exit(EXIT_FAILURE);
}

/* ...... */
```


60 Raytracer

- Ziel: Eine möglichst realistische Darstellung einer künstlichen Szene mit einem endlichen Rechenaufwand
- Idee: verfolgen von Lichtstrahlen und Bestimmung des von der Szene ausgestrahlten Lichtes

60.0 Ausgangsmodel:

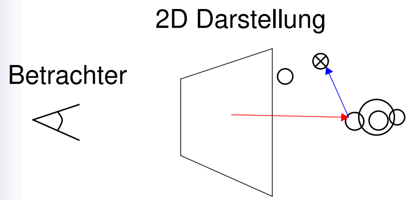


- Alle von der Lichtquelle ausgehenden Lichtstrahlen werden verfolgt
- Aus der Summe aller auf einen Oberflächenpunkt treffender Lichtstrahlen, unter Berücksichtigung des Einfallswinkel und der Oberflächenbeschaffenheit wird erneut ein ausgehender Lichtstrahl in die Szene geschickt
- Trifft ein Lichtstrahl auf die Darstellungsfläche, so wird der berechnete Farbwert angezeigt

60.0 Problem

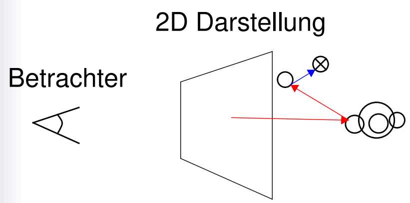
- Es gibt unendlich viele Lichtstrahlen, die meisten davon führen zu keiner Darstellung
- Erst nachdem alle Lichtstrahlen berechnet wurden, kann Aufgrund der Reflexion die Helligkeit und Farbe an einem Punkt entgültig bestimmt werden
- Lösung: Den Lichtstrahl vom Betrachter ausgehend verfolgen und die wenig bedeutenden Lichtstrahlen weglassen

60.1 Vereinfachtes Model: Farbintensität



- Von jedem Bildpunkt ausgehend wird ein Suchstrahl in die Szene geschickt
- Trifft der Suchstrahl auf ein Objekt, so wird ein Suchstrahl von dort zu allen Lichtquellen geschickt (Schattensucher).
- Erreicht der Schattensucher ohne Hindernis die Lichtquelle so wird anhand des Einfallswinkels des Lichtes der Farbwert bestimmt

60.2 Vereinfachtes Model: Spiegelung



- Zusätzlich wird vom Objekt aus ein neuer Suchstrahl in Szene geschickt und für diesen die Farbe des reflektierten Lichtes bestimmt (Was sieht der Oberflächenpunkt?)
- Dieser Vorgang wird rekursive fortgesetzt und nach einer festen Rekursionstiefe abgebrochen
- Aus der Summe des direkten und reflektierten Lichtes wird der Farbwert des Punktes bestimmt

60.3 RayTracer

- RayTracer erzeugen keine wirklich naturgetreuen Darstellungen
  - zu harte Schatten (fehlendes Streulicht)
  - Treppeneffekte an harten Farbübergängen (z.B. Schatten)
    Lösung: Oversampling
- ... sind rechenintensive.
- ... eignen sich gut als paralleles Rechenbeispiel
  - PThreads in Aufgabe 10
  - MEMSY (1993 fertiggestellte Parallelrechner vom IMMD)