TECHNISCHE UNIVERSITÄT DRESDEN

Faculty of Computer Science Institute of Systems Architecture, Operating Systems Group

EINFÜHRUNGSPRAKTIKUM: STRATEGIESPIELE

HANNES WEISBACH

ORGANISATION

- **individuelles** Arbeiten, keine Gruppen
- plagiieren verboten
- Code nicht veröffentlichen (GitHub)
- Pflicht: Einschreibung in Prüfung
- ihr bekommt: Quellcode-Archiv
- ihr liefert: einen **Computer-Gegner**
- E-Mail: einfprakt@os.inf.tu-dresden.de

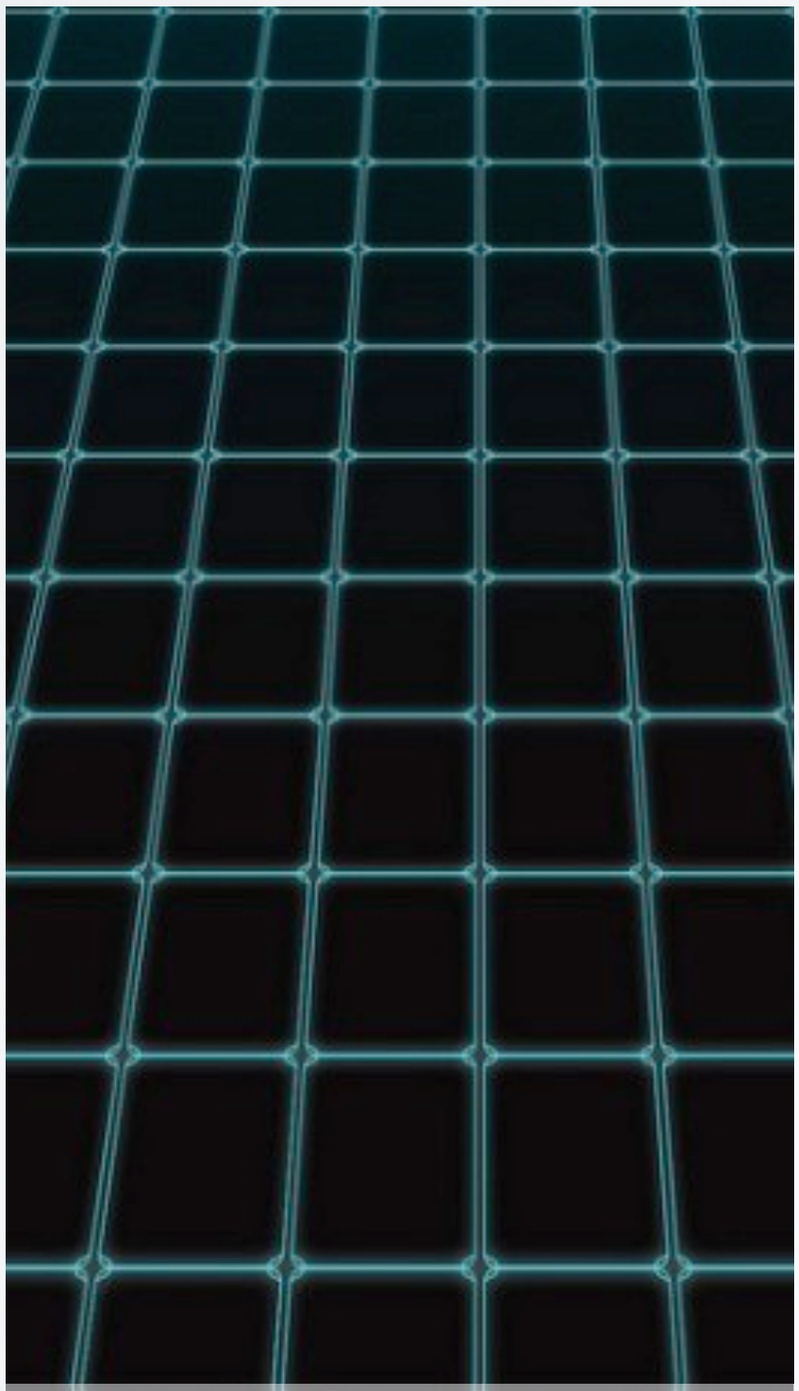
ABGABE

Mittwoch, 20.3.2019
12:00 Uhr MEZ
einfprakt@os.inf.tu-dresden.de

TECHNISCHE UNIVERSITÄT DRESDEN

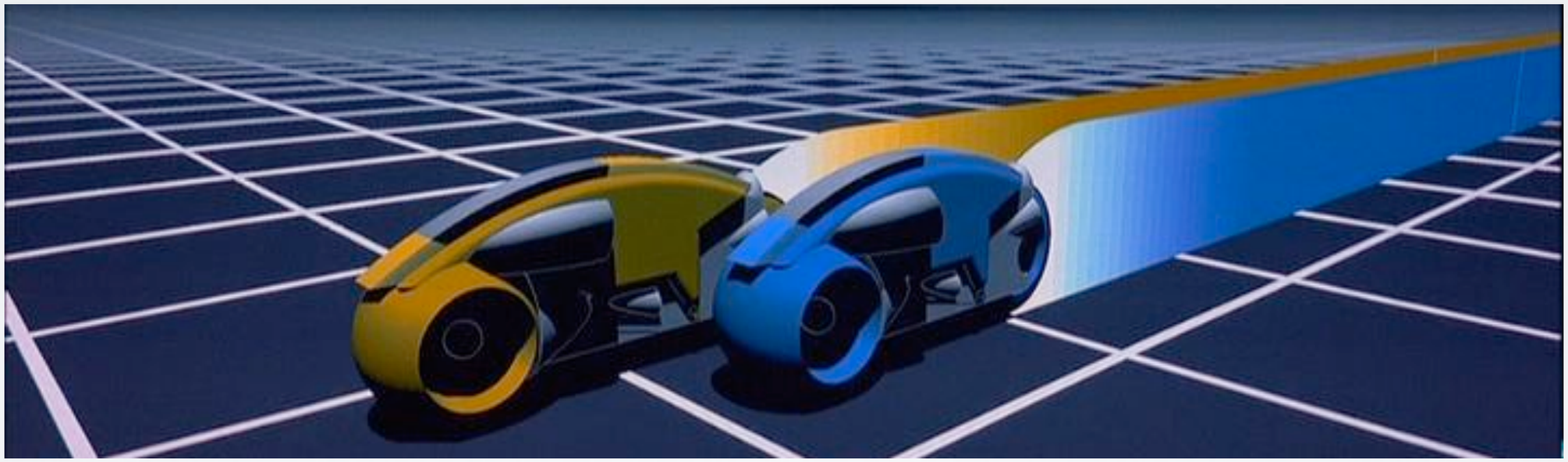
PROGRAMMIEREN

GAME GRID

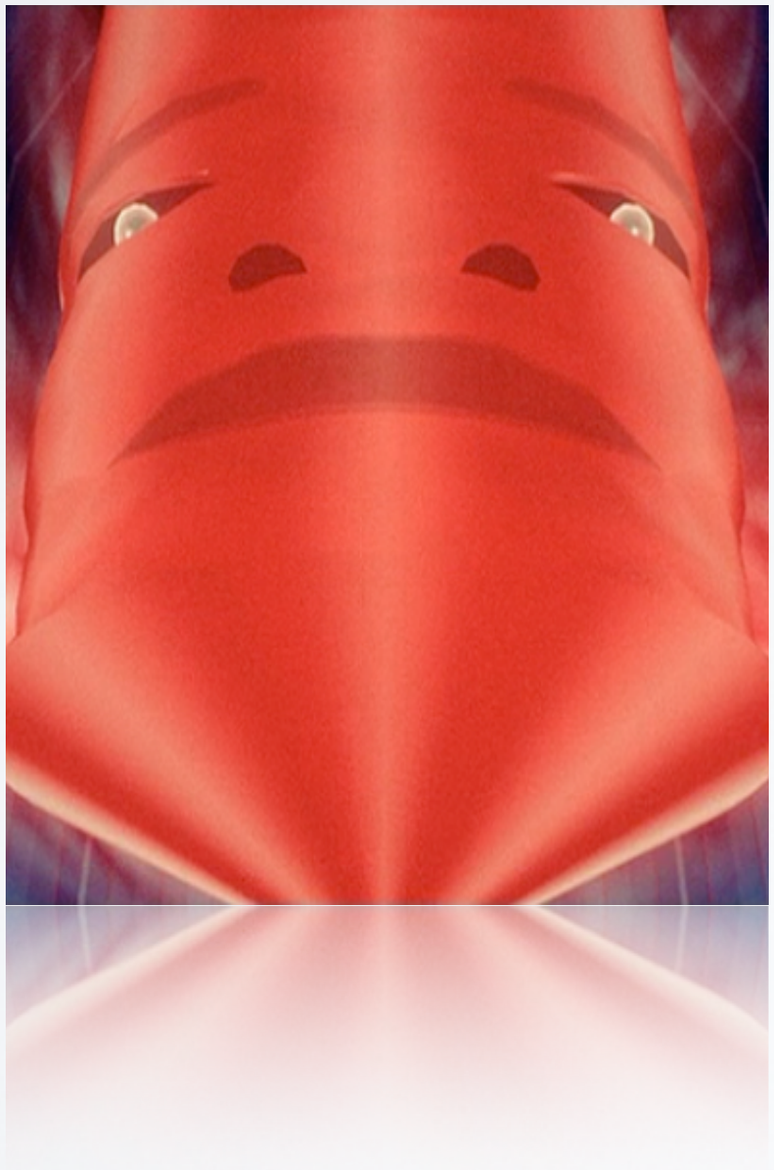


- Entwicklung unter Linux in C oder C++
- Referenz: PCs im ZIH
- Kommandozeile
- Tar-Archiv zu jedem Spiel auf Webseite
- bauen mit **`make`**
- **`README`**

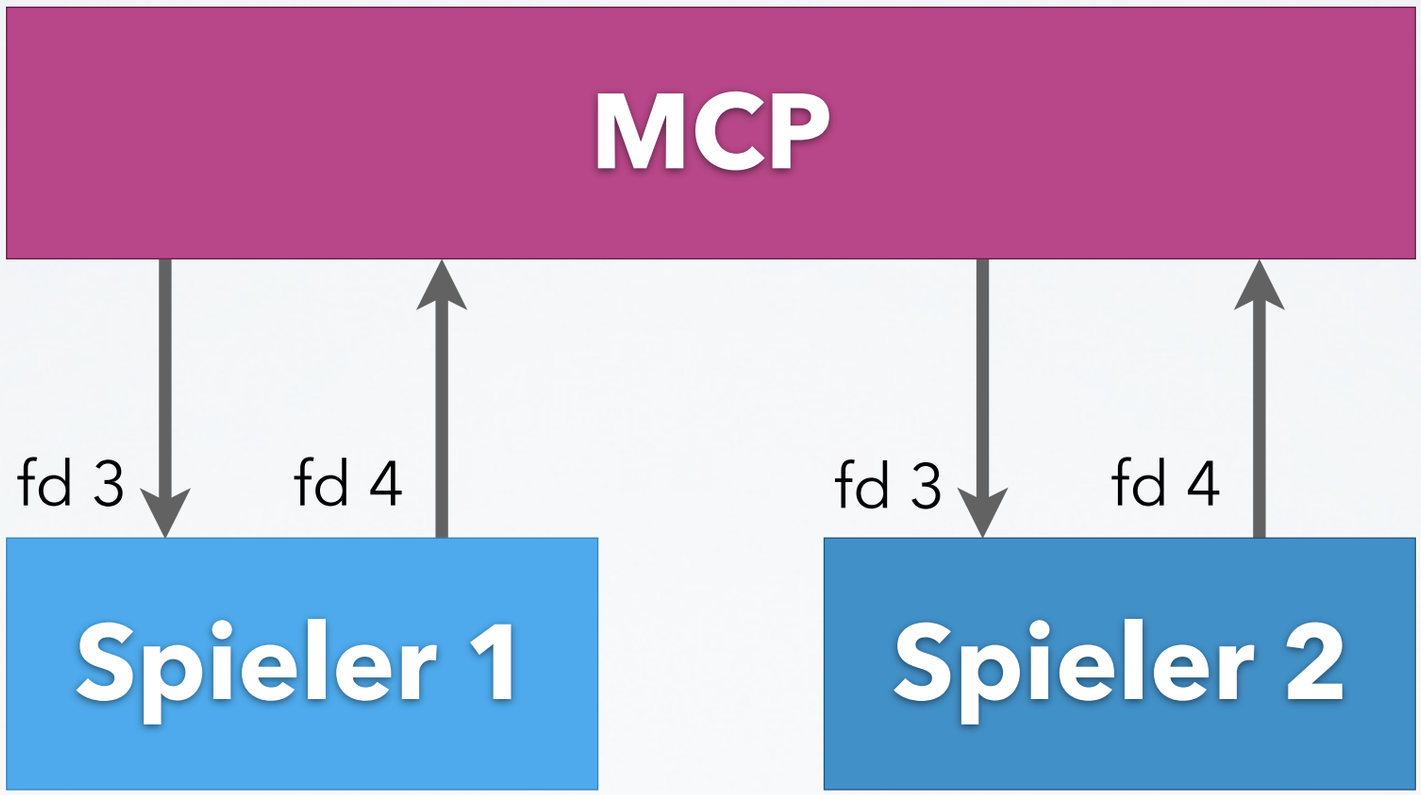
SPIELER



- interaktiver Tastatur-Spieler
- euer Computer-Gegner
  - diese Datei ist von euch auszufüllen
- Master Control Program



- verbindet zwei Spieler
  - überträgt Spielfeld
  - empfängt Zug
    - siehe Tastatur-Spieler
- überwacht Regeln und Ressourcen


MCP
fd 3
fd 4
fd 3
fd 4
Spieler 1
Spieler 2

MAKE

`make all`	baut alles
`make demo`	zwei Tastatur-Spieler
`make run`	euer Gegner und Tastatur
`make fight`	euer Gegner zweimal
`make clean`	gebaute Dateien löschen
`make help`	Anleitung

FAQ

- Linux-Installation in VirtualBox
- gezielte Übergabe eines Spielzustands an Spieler
- dynamische Arrays
- Zufallszahlen

TECHNISCHE UNIVERSITÄT DRESDEN

BESTEHEN

ANFORDERUNG

- euer Code muss **fehlerfrei kompilieren**
  - „schöner“ Code bringt Bonuspunkte
- Computer-Gegner muss **gültig spielen**
  - Regelverstöße führen zum Nicht-Bestehen
- **nicht-triviale** Strategie
  - klare Gewinnchancen nutzen
  - dem Gegner keine Steilvorlagen liefern
- Tipp: reicht **Zwischenstände** ein

CODEQUALITÄT

- **konsistente** Formatierung
- **sprechende** Bezeichner
- Gültigkeitsbereich minimieren
- **erläuternde** Kommentare
- **klare** Code-Struktur
- kein Spaghetti- oder Lasagne-Code
- **DRY:** Don’t Repeat Yourself

REGELN

- Spieler und MCP kommunizieren über UNIX-Pipes
  - wie das geht seht ihr im Tastatur-Spieler
  - ihr dürft von dort abschreiben
- MCP-Code darf nicht benutzt werden
  - ist deshalb auch teilweise in Assembler
- erlaubt: C-Bibliothek, C++/STL, pthreads
  - andere Bibliotheken auf Anfrage

TURNIER

Freitag, 29.3.2018
per Email

TURNIERREGELN

- **`make fight`**
  - testet eure Spieler gegeneinander
- 60 Sekunden Bedenkzeit, 1GB Speicher
  - nach Ablauf der Bedenkzeit wird Signal **`SIGXCPU`** zugestellt
  - nach einer weiteren Sekunde **`SIGKILL`**
- Threads sind erlaubt, **`fork()`** nicht
- wartender Gegner bekommt **`SIGSTOP`**

ABLAUF

- konzentriert euch zuerst darauf, **gültige Züge** zu finden
- Donnerstag/Freitag: Termin zur individuellen **Besprechung** eures Designs
- danach Zeit für **fortgeschrittene** Ideen
- **Mittwoch: Abgabe bis 12:00 Uhr**
- nutzt E-Mail bei Fragen

TECHNISCHE UNIVERSITÄT DRESDEN

DOS & DON'TS

```
if((buffer[j-3]=='b' && buffer[j-6]=='-' && j!=4 && j!
=8 && j!=12&&j!=16&&j!=20&&j!=24&&j!=28&&j!=32) ||
(buffer[j-4]=='b'&&buffer[j-8]=='-'&&j!=1&&j!=5&&j!
=9&&j!=13&&j!=17&&j!=21&&j!=25&&j!=29) ||
(buffer[j+1]=='W'&&buffer[j+5]=='b'&&buffer[j+10]=='-'&
&j!=4 && j!=8 && j!=12&&j!=16&&j!=20&&j!=24&&j!=28&&j!
=32) ||
(buffer[j+1]=='W'&&buffer[j+4]=='b'&&buffer[j+8]=='-'&&
j!=1&&j!=5&&j!=9&&j!=13&&j!=17&&j!=21&&j!=25&&j!=29))
```

```
if ( !possibilities[piece_number][5] && right_side )
{                                       // wenn kein jump gesetzt ist und die figur im rechten bereich ist
    if ( ( possibilities[piece_number][2] & choice1 ) == choice1 ) {possibilities[piece_number][3] += 20; possibilities[piece_number][4] = choice1; possibilities[piece_number][5] = 1;} else {
        if ( ( possibilities[piece_number][2] & choice2 ) == choice2 ) {possibilities[piece_number][3] += 15; possibilities[piece_number][4] = choice2; possibilities[piece_number][5] = 1;} else {
            if ( ( possibilities[piece_number][2] & choice3 ) == choice3 ) {possibilities[piece_number][3] += 15; possibilities[piece_number][4] = choice3; possibilities[piece_number][5] = 0;} else {
                if ( ( possibilities[piece_number][2] & choice4 ) == choice4 ) {possibilities[piece_number][3] += 10; possibilities[piece_number][4] = choice4; possibilities[piece_number][5] = 0;} else {
                    if ( ( possibilities[piece_number][2] & choice7 ) == choice7 ) {possibilities[piece_number][3] += 8; possibilities[piece_number][4] = choice7; possibilities[piece_number][5] = 1;} else {
                        if ( ( possibilities[piece_number][2] & choice8 ) == choice8 ) {possibilities[piece_number][3] += 5; possibilities[piece_number][4] = choice8; possibilities[piece_number][5] = 1;} else {
                            if ( ( possibilities[piece_number][2] & choice5 ) == choice5 ) {possibilities[piece_number][3] += 5; possibilities[piece_number][4] = choice5; possibilities[piece_number][5] = 0;} else {
                                if ( ( possibilities[piece_number][2] & choice6 ) == choice6 ) {possibilities[piece_number][3] += 3; possibilities[piece_number][4] = choice6; possibilities[piece_number][5] = 0;} else {
                                    if ( ( possibilities[piece_number][2] & choice9 ) == choice9 ) {possibilities[piece_number][3] -= 3; possibilities[piece_number][4] = choice9; possibilities[piece_number][5] = 0;} else {
                                        if ( ( possibilities[piece_number][2] & choice10 ) == choice10 ) {possibilities[piece_number][3] -= 5; possibilities[piece_number][4] = choice10; possibilities[piece_number][5] = 0;} else {possibilities
                                    }       // wird der bevorzugte zug gesetzt, jump entsprechend auf 1 oder 0 gesetzt und die priorität wird erhöht, bzw. erniedrigt
                                }
                            }
                        }
                    }
                }
            }
        }
    }
}

if ( !possibilities[piece_number][5] && left_side )
{                                       // wenn kein jump gesetzt ist und die figur im linken bereich ist
    if ( ( possibilities[piece_number][2] & choice2 ) == choice2 ) {possibilities[piece_number][3] += 20; possibilities[piece_number][4] = choice2; possibilities[piece_number][5] = 1;} else {
        if ( ( possibilities[piece_number][2] & choice1 ) == choice1 ) {possibilities[piece_number][3] += 15; possibilities[piece_number][4] = choice1; possibilities[piece_number][5] = 1;} else {
            if ( ( possibilities[piece_number][2] & choice4 ) == choice4 ) {possibilities[piece_number][3] += 15; possibilities[piece_number][4] = choice4; possibilities[piece_number][5] = 0;} else {
                if ( ( possibilities[piece_number][2] & choice3 ) == choice3 ) {possibilities[piece_number][3] += 10; possibilities[piece_number][4] = choice3; possibilities[piece_number][5] = 0;} else {
                    if ( ( possibilities[piece_number][2] & choice8 ) == choice8 ) {possibilities[piece_number][3] += 8; possibilities[piece_number][4] = choice8; possibilities[piece_number][5] = 1;} else {
                        if ( ( possibilities[piece_number][2] & choice7 ) == choice7 ) {possibilities[piece_number][3] += 5; possibilities[piece_number][4] = choice7; possibilities[piece_number][5] = 1;} else {
                            if ( ( possibilities[piece_number][2] & choice6 ) == choice6 ) {possibilities[piece_number][3] += 5; possibilities[piece_number][4] = choice6; possibilities[piece_number][5] = 0;} else {
                                if ( ( possibilities[piece_number][2] & choice5 ) == choice5 ) {possibilities[piece_number][3] += 3; possibilities[piece_number][4] = choice5; possibilities[piece_number][5] = 0;} else {
                                    if ( ( possibilities[piece_number][2] & choice10 ) == choice10 ) {possibilities[piece_number][3] -= 3; possibilities[piece_number][4] = choice10; possibilities[piece_number][5] = 0;} else {
                                        if ( ( possibilities[piece_number][2] & choice9 ) == choice9 ) {possibilities[piece_number][3] -= 5; possibilities[piece_number][4] = choice9; possibilities[piece_number][5] = 0;} else {possibilities[pi
                                    }       // wird der bevorzugte zug gesetzt, jump entsprechend auf 1 oder 0 gesetzt und die priorität wird erhöht, bzw. erniedrigt
                                }
                            }
                        }
                    }
                }
            }
        }
    }
}

if ( !possibilities[piece_number][5] && left_left_side )
{                                       // wenn kein jump gesetzt ist und die figur im ganz linken bereich ist
    if ( ( possibilities[piece_number][2] & choice2 ) == choice2 ) {possibilities[piece_number][3] += 20; possibilities[piece_number][4] = choice2; possibilities[piece_number][5] = 1;} else {
        if ( ( possibilities[piece_number][2] & choice4 ) == choice4 ) {possibilities[piece_number][3] += 15; possibilities[piece_number][4] = choice4; possibilities[piece_number][5] = 0;} else {
            if ( ( possibilities[piece_number][2] & choice8 ) == choice8 ) {possibilities[piece_number][3] += 15; possibilities[piece_number][4] = choice8; possibilities[piece_number][5] = 1;} else {
                if ( ( possibilities[piece_number][2] & choice6 ) == choice6 ) {possibilities[piece_number][3] += 10; possibilities[piece_number][4] = choice6; possibilities[piece_number][5] = 0;} else {
                    if ( ( possibilities[piece_number][2] & choice10 ) == choice10 ) {possibilities[piece_number][3] += 5; possibilities[piece_number][4] = choice10; possibilities[piece_number][5] = 0;} else {
                        if ( ( possibilities[piece_number][2] & choice12 ) == choice12 ) {possibilities[piece_number][3] += 5; possibilities[piece_number][4] = choice12; possibilities[piece_number][5] = 1;} else {possibilities[piece_number][4
                    }                   // wird der bevorzugte zug gesetzt, jump entsprechend auf 1 oder 0 gesetzt und die priorität wird erhöht, bzw. erniedrigt
                }
            }
        }
    }
}

if ( !possibilities[piece_number][5] && right_right_side )
{                                       // wenn kein jump gesetzt ist und die figur im ganz rechten bereich ist
    if ( ( possibilities[piece_number][2] & choice1 ) == choice1 ) {possibilities[piece_number][3] += 20; possibilities[piece_number][4] = choice1; possibilities[piece_number][5] = 1;} else {
        if ( ( possibilities[piece_number][2] & choice3 ) == choice3 ) {possibilities[piece_number][3] += 15; possibilities[piece_number][4] = choice3; possibilities[piece_number][5] = 0;} else {
            if ( ( possibilities[piece_number][2] & choice7 ) == choice7 ) {possibilities[piece_number][3] += 15; possibilities[piece_number][4] = choice7; possibilities[piece_number][5] = 1;} else {
                if ( ( possibilities[piece_number][2] & choice5 ) == choice5 ) {possibilities[piece_number][3] += 10; possibilities[piece_number][4] = choice5; possibilities[piece_number][5] = 0;} else {
                    if ( ( possibilities[piece_number][2] & choice9 ) == choice9 ) {possibilities[piece_number][3] += 5; possibilities[piece_number][4] = choice9; possibilities[piece_number][5] = 0;} else {
                        if ( ( possibilities[piece_number][2] & choice11 ) == choice11 ) {possibilities[piece_number][3] += 5; possibilities[piece_number][4] = choice11; possibilities[piece_number][5] = 1;} else {possibilities[piece_number][4
                    }                   // wird der bevorzugte zug gesetzt, jump entsprechend auf 1 oder 0 gesetzt und die priorität wird erhöht, bzw. erniedrigt
                }
            }
        }
    }
}
```

```
#include <stdio.h>

int i,j,m,l,n,k,a,c,t,s,w,v;


int main(void)
{
    /* ... */
    for (i = 0; i < 8; i++) {
        /* ... */
        for (j = 0; j < 8; j++) {
            /* ... */
            for (i = 5; i > 0; i--) {
                /* ... */
            }
        }
    }
}
```

```c
#include <stdio.h>



int main(void)
{
    /* ... */
    for (int i = 0; i < 8; i++) {
        /* ... */
        for (int j = 0; j < 8; j++) {
            /* ... */
            for (int i = 5; i > 0; i--) {
                /* ... */
            }
        }
    }
}
```

```
for (int i; i < 8; i++) {
    /* ... */
}
```

```
bool check_for_something()
{
    bool found;
    if ( /* some condition */ )
        found = true;
    return found;
}
```

```
int get_element(i)
{
    return matrix[i];
}

get_element(-1);
```

```
int get_element(vector<int> &v, i)
{
    if (v.size())
        return v[i];
    else
        return v[0];
}
```

```
void output_move()
{
    char *buffer = malloc(1024);
    snprintf(buffer, 1024, /* format string */);
    write(output_to_mcp, buffer, 1024);
}
```

```
void analyze()
{
    Move *move = new Move();
    /* analyze move */
    delete move;
}
```

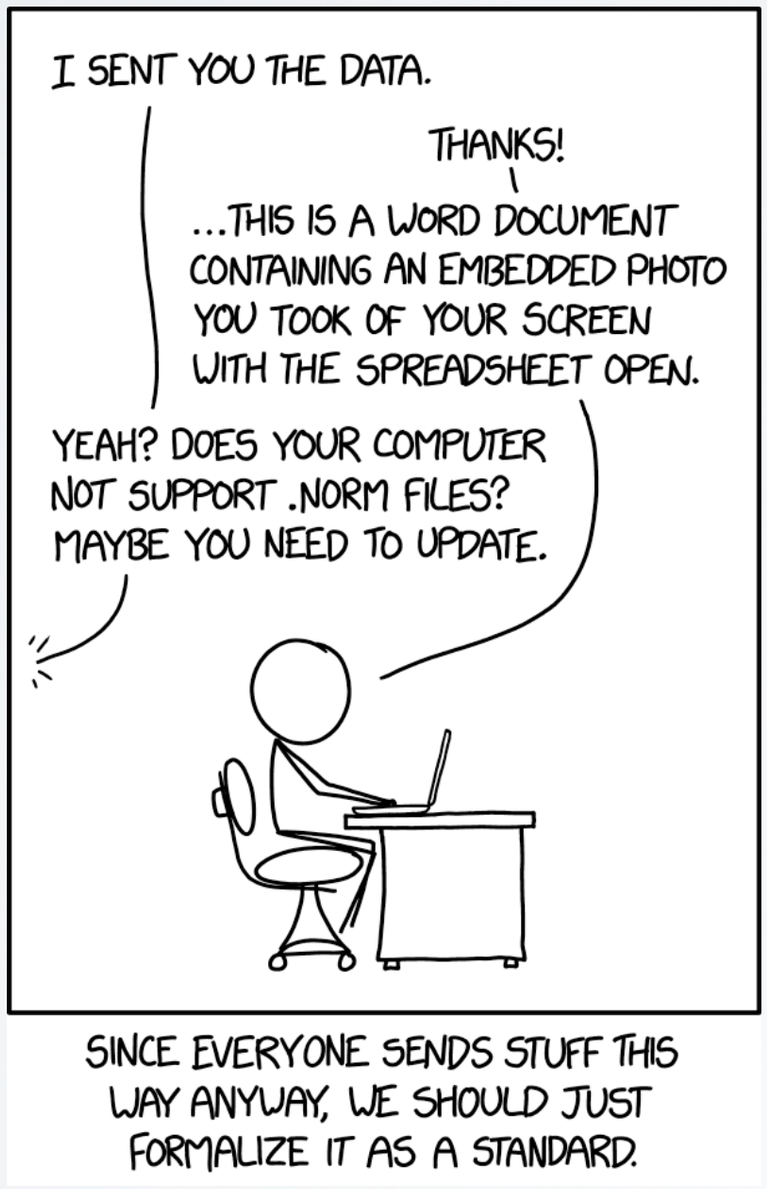
```
void analyze()
{
    Move move;
    /* analyze move */

}
```

EMAIL DOS

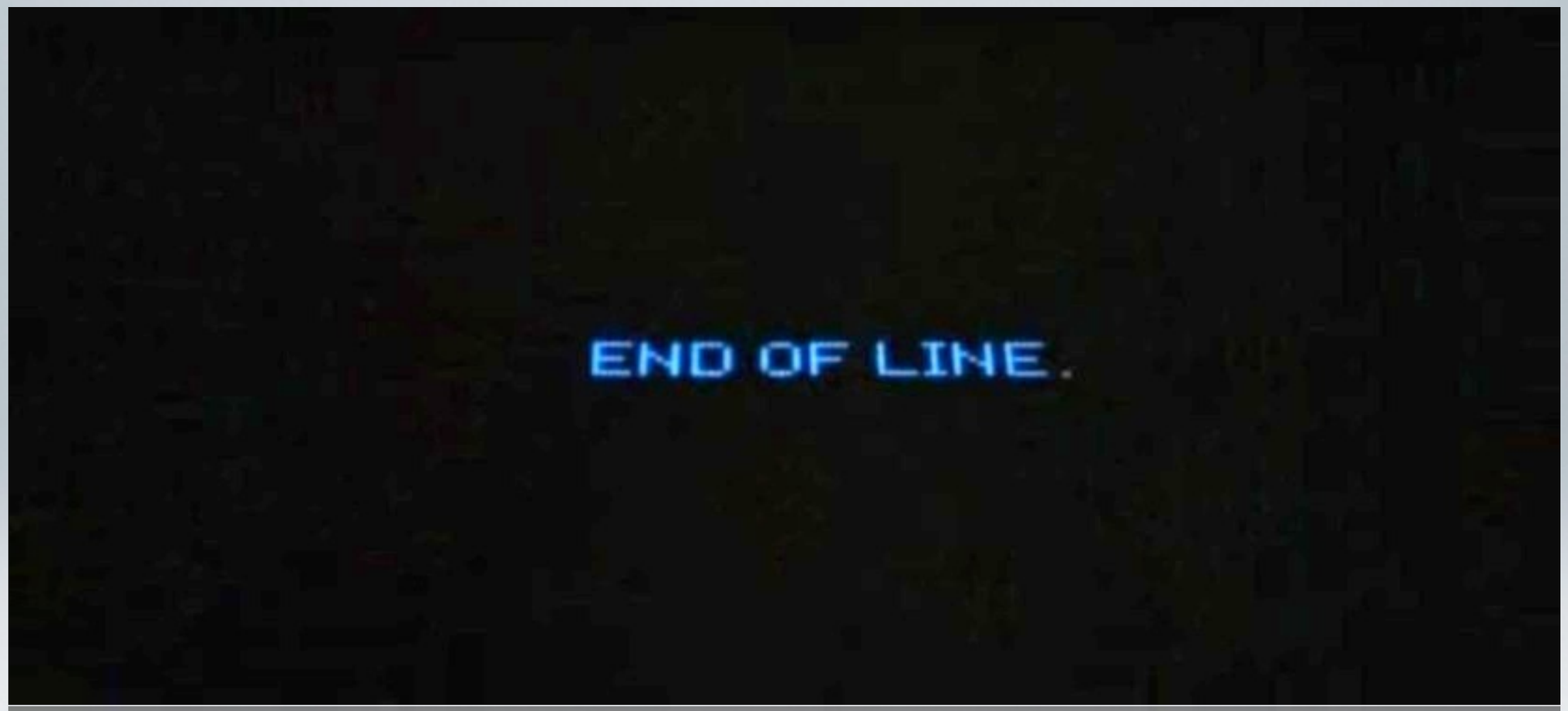
- **Immer** an **einfprakt@os.inf.tu-dresden.de** schreiben/antworten
- Spielnamen im Betreff hilft der Zuordnung
- Fehler-/Problembeschreibung
- ggf. (eigenen!) Code anhängen

EMAIL DON'TS



Quelle: https://xkcd.com/2116/

- Keine Bildschirmfotos per Email
- Keine Fotos von Bildschirmen per Email
- Pro-Tipp: Text kann man kopieren & einfügen. Auch auf/von der Shell.


END OF LINE.

GRUPPEN

Matrikel	Spiel	Student	Mitarbeiter
...00 – ...20	Dame	Tony Fiedler	Carsten Weinhold
...21 – ...46	Haliotis	Ferdinand Thiessen	Martin Küttler
...47 – ...69	Isolation	Robert Ufer	Maksym Planeta
...70 – ...85	Back-gammon	Manuel Thieme	Jan Bierbaum
...86 – ...99	Mühle	Benno Fünfstück	Hannes Weisbach