

Besprechung fängt um 16:15 Uhr an.

Besprechung der lilo¹

Tafelübung Systemprogrammierung

Andreas Schärfl <andreas.schaertl@fau.de>

SS 2021

¹<https://www.schaertl.me/sp1>

- In diesem Zoommeeting wollen wir eure Abgaben genauer angucken.
- Typischerweise machen viele Studierende ähnliche Fehler und haben ähnliche Fragen.
- Beim nächsten mal besser machen und sich freuen.
- Die Folien sind eher als Erinnerung gedacht. Diese Übung soll interaktiv sein, nicht Frontalunterricht. Liegt an euch.
- Diese Folien gibt es im Gitlab² als \LaTeX oder als PDF in meinem Home³.

²<https://gitlab.cs.fau.de/kissen/sp1-ss21-besprechung>

³<https://wwwcip.cs.fau.de/~ru64tiji/sp1/>

Formatierung

Formatierung: Mindestanforderung

- Vernünftige Einrückung und Formatierung sind nicht nur *guter Stil*, sondern *obligatorisch!*
- Wer schlechten Code schreibt quält sich selber und, noch schlimmer, andere.
- Wer seinen Editor (noch nicht) beherrscht:
 \$ clang-format -i wsort.c
 vor der Abgabe ausführen.

Formatierung: Überblick bewahren: Negativbeispiel

```
static float fmax(float p, float q)
{
    if (isnan(p)) {
        return q;
    } else if (isnan(q)) {
        return p;
    } else {
        if (p > q) {
            return p;
        } else {
            return q;
        }
    }
}
```

- Schachtelung ist sehr oft ein schlechtes Zeichen!

Formatierung: Überblick bewahren: Besser

```
static float fmax(float p, float q)
{
    if (isnan(p)) {
        return q;
    }

    if (isnan(q)) {
        return p;
    }

    if (p > q) {
        return p;
    } else {
        return q;
    }
}
```

- Es gibt Leute die euch sagen, mehrere return seien problematisch.
- Diese Leute sind tragische Opfer eines Cargokults mit Ursprung in der Assemblerprogrammierung⁴.

⁴ <https://softwareengineering.stackexchange.com/questions/118703/>

Formatierung: Überblick bewahren: Fehlerbehandlung

- Besonders bei Fehlerbehandlungen! Wenn wir am Ende sind, einfach Block zumachen und gut ist!
- Noch ein Negativbeispiel:

```
char *buf;  
if ((buf = malloc(SIZE)) == NULL) {  
    perror("malloc");  
    exit(EXIT_FAILURE);  
} else {  
    // do stuff w/ buf  
}
```

- Das else ist nicht notwendig; führt nur zu unnötiger Einrückung.

- Kommentare in eine eigene Zeile. Horizontale Auflösung ist limitiert!
- Nicht mehrere Sachen in einer Zeile *verstecken*!
- Variablennamen kann man klug wählen. `current` oder `next` sagt mehr aus als `element`.
- Für interessierte: *Linux Kernel Coding Style*⁵. Aber mit Vorsicht genießen.

⁵ <https://www.kernel.org/doc/html/latest/process/coding-style.html>

Premature Optimization

Premature Optimization

```
struct node *cur = head;

if (cur->value == value) {
    return -1;
}

struct node *prev = NULL;
while (cur != NULL) {
    if (cur->value == value) {
        return - 1;
    }

    prev = cur;
    cur = cur->next;
}

// etc.
```

Premature Optimization

- Gedachte Optimierungen machen den Code schwerer zu verstehen.
- In den meisten Fällen machen solche Optimierungen den Code höchstens langsamer. Schlimmer noch, die gesteigerte Komplexität führt zu Fehlern!

Wie geht es besser?

- *Premature optimization is the root of all evil*⁶!
- Messen! Zum Beispiel mit `perf(1)`⁷ auf Linux.

⁶<https://softwareengineering.stackexchange.com/questions/80084/>

⁷<https://fsv.tf/perf>

Besprechung

- Jetzt Besprechung einer beispielhaften Aufgabe.
- Be excellent to each other.

Gruppenarbeit

- Wer noch keine Gruppe für die Abgabe hat, jetzt gäbe es eine Möglichkeit eine Gruppe zu finden.

Fragen?