

# Übungsblatt 8

Abgabe bis spätestens 18.01.2024 in Stud.IP

## Allgemeiner Hinweis

Für dieses Übungsblatt benötigt ihr das in StudIP zur Verfügung gestellte Archiv ueb8-vorgabe.zip mit Vorgabedateien.

## Aufgabe 1 (5 Punkte)

Im Verzeichnis aufgabe1 im mitgelieferten Archiv findet ihr das Programm aufgabe1.cc.

- a) Erläutert in eigenen Worten die Arbeitsweise des Programms.
- b) Führt das Programm (mindestens) dreimal aus und druckt jeweils die Ausgabe des Programms in Eurer Aufgabenlösung ab. Welche vier Besonderheiten sind in der Ausgabe zu sehen? Bitte markiert jede Besonderheit mindestens einmal in der abgedruckten Ausgabe und beschreibt zusätzlich kurz, was dort zu sehen ist.
- c) Beschreibt für jede Art von Besonderheit aus c) im Detail, wodurch sie entsteht, also an genau welcher Stelle im Programmcode die Ausführung eines Threads unterbrochen wird, damit dieser Effekt auftritt. Erklärt wo nötig die internen Vorgänge, die am Programmcode nicht sichtbar sind.
- d) Schützt den/die kritische(n) Abschnitt(e) des Programms mit Mutexen. Gebt das modifizierte Programm unter dem Dateinamen aufgabe1d.cc mit ab (Makefile entsprechend abändern nicht vergessen).

## Aufgabe 2 (3 Punkte)

Im Verzeichnis aufgabe2 im mitgelieferten Archiv findet ihr das Programm aufgabe2.cc.

- a) Lest und versteht das Programm bzw. dessen Quelltext und erläutert die Funktionsweise des Programms im Hinblick auf das Thema Nebenläufigkeit.
- b) Kennzeichnet die beiden kritischen Abschnitte innerhalb des Programms. Es muss klar hervorgehen, wo ein kritischer Abschnitt beginnt und wo er endet. Die kritischen Abschnitte sollen so groß wie nötig, aber so klein wie möglich gewählt werden (mit Begründung).

### Aufgabe 3 (2 Punkte)

In den folgenden Code-Ausschnitten werden die mit „Thread-Funktion“ gekennzeichneten Funktionen regelmäßig von konkurrierenden Threads innerhalb eines Prozesses aufgerufen. Erläutert für jede Variante kurz, wo nebenläufige Zugriffe synchronisiert werden müssen, um Verfälschungen der Ergebnisse vorzubeugen (bzw. wo keine Synchronisation notwendig ist, soweit dies nicht bereits aus der Betrachtung des kritischen Abschnitts hervorgeht). Achtet darauf, den kritischen Abschnitt so klein wie möglich zu halten und keine nicht-kritischen Abschnitte als kritisch zu kennzeichnen.

Hinweis: Es sollen keine Annahmen über die hier nicht definierten Symbole getroffen werden, außer, dass sie irgendetwas sinnvolles im Zusammenhang mit dem gezeigten Code bewirken. Eine etwaige Optimierung während der Übersetzung soll nicht betrachtet werden.

a)

```
struct rational {
    long zaehler;
    unsigned long nenner;
} rat = { 1, 1 };

/* Thread-Funktion */
void mult(long z, unsigned long n) {
    if (n != 0) {
        rat.zaehler = z * rat.zaehler;
        rat.nenner = n * rat.nenner;
    }
}
```

---

b)

```
struct rational {
    long zaehler;
    unsigned long nenner;
};

unsigned long ggT(unsigned long a, unsigned long b);

/* Thread-Funktion */
rational add(rational p, rational q) {
    unsigned long teiler = ggT(p.nenner, q.nenner);
    unsigned long kgv = (p.nenner / teiler) * q.nenner;

    long n1 = p.zaehler * q.nenner/teiler;
    long n2 = q.zaehler * p.nenner/teiler;

    return { n1 + n2, kgv };
}
```

---

c)

```
extern int konto;

/* Thread-Funktion */
void faden(void *arg) {
    int value = *arg;
    for (size_t n = 0; n < 500; n++) {
        int konto_neu = konto + value;
        usleep(800);
        konto = konto_neu;
    }
}
```

---

d)

```
extern unsigned long global;

/* Thread-Funktion */
void inc_global() {
    global++;
}
```

## Abgabe

Bis 23:59 Uhr am 18.01.2024 digital in StudIP. **Es gelten die vereinbarten Scheinbedingungen (siehe StudIP).** Bitte beachtet unsere ergänzenden Hinweise ebenda.

Ladet die abgaberelevanten Dateien in DoIT hoch.

Eure Ansätze und der gewählte Lösungsweg müssen nachvollziehbar sein. Achtet insofern auf eine saubere Dokumentation im Quelltext und im abgegebenen PDF-Dokument. Benennt alle von Euch verwendeten Quellen, auch Zusammenarbeit mit anderen Gruppen und verwendete Unterlagen aus früheren Jahrgängen.

Programmieraufgaben sind im Zweifel in C++20 zu entwickeln. Die Korrektheit der Lösung bzw. deren Grenzen sind grundsätzlich in der Abgabe nachzuweisen. Dies geschieht neben der Dokumentation des Programmcodes in den Quelldateien und zusätzlich in dem abgegebenen PDF-Dokument mit der Lösungsbeschreibung durch geeignete Tests, deren Auswahl und Eignung begründet werden müssen. Als **Referenzplattform** gelten die Linux-Rechner im Rechnerpool in MZH E0.