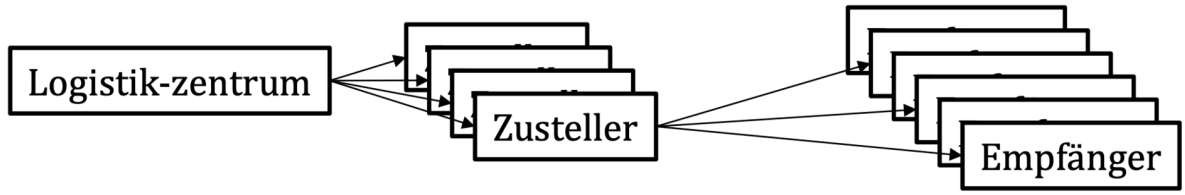
Betriebssysteme – Aufgabe 4

- **Threads / Synchronisation**
- **Systemaufrufe**

Simulieren Sie in C/C++ mithilfe von **Threads**, **Mutex** und **Semaphoren** unter Linux das folgende Problem (Brieflogistik):



1. Das Programm besteht aus einem Logistikzentrum, **m Zustellerthreads** und **n Empfängerthreads**. Jedes dieser Objekte soll durch eigene Threads simuliert werden.

2. Im Logistikzentrum lagern Briefe, die an die Empfänger zugestellt werden sollen. Das Logistikzentrum soll durch einen einzigen Thread simuliert werden (Singleton), der Briefe mit zufällig gewählter Rate **r** (Intervall mittels usleep simuliert) erzeugt und an die **Zustellerthreads** verteilt. Ein Brief ist eine Datenstruktur, bestehend mindestens aus einem beliebigen String als Inhalt sowie der ID des Empfängers.

3. Jeder der m **Zustellerthreads** besitzt eine eigene Datenstruktur mit fester Lagerkapazität für Briefe. In dieser Datenstruktur soll der Thread des Logistikzentrums Briefe ablegen. Wenn die Kapazität voll ist, signalisiert das Logistikzentrum dies dem **Zustellerthread** und dieser beginnt mit der Zustellung.

   In diesem Zeitraum können keine weiteren Briefe vom Logistikzentrum mehr akzeptiert werden. Der **Zustellerthread** besucht die Empfänger in beliebiger Reihenfolge und liefert die Briefe passend aus. Die Fahrtzeiten zwischen Empfängern soll durch Aufrufe von **sleep** oder **usleep** mit zufällig gewählter Zeitdauer simuliert werden. Nach Auslieferung aller Briefe kehrt der **Zusteller-thread** zurück zum Logistikzentrum und kann weitere Briefe in Empfang nehmen.

4. Ein **Empfängerthread** besitzt eine Datenstruktur Briefkasten, die Briefe aufnehmen kann. Dies soll in 2 Varianten möglich sein:

   a. Der Empfänger schaut periodisch in dieser Datenstruktur nach (simuliert durch **sleep** oder **usleep**).

Betriebssysteme – Aufgabe 4

b. Der **Zustellerthread** klingelt, d.h. benachrichtigt den schlafenden Empfänger, dass ein Brief im Briefkasten ist. Dadurch wird der **Empfängerthread** geweckt. Wenn der **Empfängerthread** den Empfang bestätigt hat, kann der Zusteller weiterfahren.

Anschließend entnimmt der **Empfängerthread** den Brief bzw. alle gelieferten Briefe und gibt den Inhalt (String) auf der Standardausgabe aus.

Implementieren Sie das Problem Brieflogistik mithilfe von Threads sowie Semaphor- und Mutexobjekten geeignet, so dass alle o.a. Anforderungen erfüllt sind. Es sollen die Parameter r, m und n über die Kommandozeile beim Aufruf oder eine Konfigurationsdatei einstellbar sein.

Messen Sie den Durchsatz (d.h. zugestellte Briefe pro Sekunde, Minute, o.ä.) Ihrer Implementierung für m=10 und n=100 sowie eine weitere, sinnvolle Kombination von Parametern m und n, jeweils getrennt für die V arianten 4.a und 4.b der Empfängerthreads. Dokumentieren Sie das Ergebnis, z.B. in Form eines Diagramms.

Die Aufgabe wird testiert, wenn Sie das Programm – mit den geforderten Anforderungen – lauffähig vorgezeigt und den Code sowie die Funktionsweise erklärt haben.