Aufgabe 4

1 Lernziel

- Behandlung zeichenorientierter Eingabegeräte
- Synchronisation in Gerätetreibern

2 Aufgabenbeschreibung

In dieser Aufgabe bekommt CoStuBS einen Treiber für die Tastatur. Dies ist gleichzeitig die erste Eingabemöglichkeit für CoStuBS . Der Treiber an sich ist in der Klasse Keyboard.h schon fertig implementiert. Eure erste Aufgabe ist es einen Tastaturpuffer in Form eines **Ringpuffers** zu implementieren. Im zweiten Schritt sollt ihr ein Testprogramm für die Eingabe schreiben. Dazu gibt es in der Vorgabe einen kleinen **Taschenrechner**, den ihr fertig implementieren müsst. Diese Aufgabe dient vor allem dazu, mögliche Synchronisationsfehler in eurem System aufzudecken.

3 Vorbereitungsfragen

Versucht, die folgenden Fragen zu beantworten **bevor** ihr mit der Implementierung beginnt.

- Wie kommuniziert eine Tastatur mit der restlichen Hardware?
- Welche Art von Signalen kann eine Tastatur an die restliche Hardware schicken und wie wird das festgelegt?
- Wird eine Tastatureingabe immer sofort abgearbeitet?
- Was ist ein Ringpuffer und wie funktioniert dieser?
- Wozu benötige ich den Ringpuffer?

4 Implementierungshinweise

Teil A

Implementiert einen Ringpuffer, auf Basis der Klasse BoundedBuffer.h. Beachtet dabei, dass es ein synchronisierter begrenzter Puffer ist, welcher für interruptgetriebene Eingabegeräte genutzt wird. Das heißt, dass die add() Methode vom Interrupthandler aufgerufen wird und nicht blockieren darf. Die get() Methode hingegen wird von einem Prozess aufgerufen und muss diesen schlafen legen, **wenn der Puffer leer ist**. Zum Testen verändert einfach die body() Methode von Echo, sodass es zu Zeichenverlusten kommt.

Teil B

Beendet die Implementierung des Taschenrechners. Die Vorgaben dazu findet ihr im Ordner tools. Am Ende sollt ihr in der Lage sein, eine Berechnungsvorschrift einzugeben. Durch das Betätigen der Enter-Taste wird diese Eingabe analysiert und das Ergebnis berechnet. Handelt es sich um eine gültige Eingabe, so wird danach das Ergebnis ausgegeben. Andernfalls erscheint eine Fehlermeldung. Danach soll es möglich sein, eine neue Berechnung einzugeben. Ein Beispielablauf sieht so aus:

```
1+2*(3+4)
= 15
+
Error: Unexpected Symbol!
1+1
= 2
```

Zuerst solltet ihr ermöglichen, dass Tastatureingaben auf dem Bildschirm dargestellt werden. Dazu müsst ihr die body()-Methode der Klasse Calculator implementieren.

Beachtet dabei, dass ihr Sonderfälle bei den Eingaben behandeln müsst. Die read() Methode von Keyboard gibt einen Key.h zurück. Dieser kann entweder ein Buchstabe oder ein Steuerzeichen der Tastatur sein. Dies wird durch die Klasse CodeTable.h festgelegt. Ermöglicht außerdem, dass eine falsche Eingabe korrigiert werden kann. Falls ihr am unteren Bildschirmrand angelangt, muss der Inhalt des Bildschirmes nach oben geschoben werden.

Wird die Enter-Taste betätigt, soll der eingegebene Ausdruck ausgewertet werden. Dazu dienen die vorgegebenen Klassen Interpreter und Scanner. Seht euch die Vorgaben an und vervollständigt die Implementierung im Scanner. Wenn ihr Erfolg hattet, solltet ihr nun ein Ergebnis oder einen Fehlercode erhalten, den ihr darstellen könnt.

Im letzten Schritt sollt ihr eine Möglichkeit schaffen, euch den Speicherinhalt an einer Adresse ausgeben zu lassen. Der Interpreter ist in der Lage die Eingabe einer Adresse zu erkennen. Implementiert die Methode evalDump(). Schaut euch dazu an, wie die anderen Evaluationsmethoden funktionieren. Diese arbeiten analog zu der Grammatik, die ihr in der Vorgabe des Interpreters findet. Ziel ist eine Ausgabe in der Art:

```
* 0xfefe
= -426770432
```

4.1 Weitere Informationen

- Informationen zur **Tastatur**
- **Keyboard Scan Codes**

Viel Erfolg!