Softwareverfikation Projekt 1

# Aufgabenstellung und Beschreibung des Round-Robin-Scheduler´s:

Wir haben mit Dafny einen Round-Robin-Scheduler (RRS) implementiert und verifiziert. Der RRS wird mit einer Warteschlange nach dem FiFo-Prinzip (First in - First out) verwaltet. In der Warteschlange befinden sich noch zu bearbeitete Prozesse. Der Ablauf des RRS ist wie folgt. Der erste Prozess wird aus der Warteschlange genommen. Er bekommt sein Quantum an Rechenzeit von der CPU. Wenn das Quantum abgelaufen ist, wird der Prozess am Ende der Warteschlange hinzugefügt. Danach wird ein neuer Prozess aus der Warteschlange genommen. Sollte ein Prozess keine Rechenzeit mehr benötigen, wird er nicht mehr in die Warteschlange eingefügt.

# Implementierung des RRS:

Unsere Implementierung war einen Prozess-Block und ein System (OS) mit einer Queue zu implementieren. Der Aufbau der einzelnen Komponenten sieht wie folgt aus.

Der Prozess (PCB\_t):

* + speichert seine Prozess-ID (pid) zur eindeutigen Wiedererkennung.
  + speichert seine benötigter Rechenzeit (duration).
  + speichert die ID vom Prozessersteller (ownerID).
  + speichert die bereits erhaltene Zeit von der CPU (usedCPU).
  + Hat einen Konstruktor „Init(pid: int, duration: int, ownerID: int)“ zur Erstellung eines Prozesses mit Initialwerten.

Das System (OS):

* + speichert eine Sequenz von Prozessen (que).
  + speichert den aktuellen Prozess (pcb).
  + speichert das Quantum (quantum).
  + hat das Prädikat „Valid()“. Es überprüft, ob es in der Sequenz keine Null-Werte sowie keine doppelten Prozess-ID´s vorkommen.
  + hat das Prädikat „inQue(pid: int)“. Es überprüft, ob sich die pid schon in der Sequenz befindet.
  + hat einen Konstruktor „Init(quantum: int)“ zum festlegen des Quantums. Es wird pcb auf null gesetzt. Die Sequenz que wird leer erstellt.
  + hat eine Methode „enQueue(prozess: PCB\_t)“ zum hinzufügen eines Prozesses.
  + Hat eine Methode „deQueue()“ zum entnehmen des ersten Elements aus der Sequenz.
  + Hat die Methode „operate()“, dass die Methode deQueue() aufruft und den Prozess in einer lokalen Variable pcb speichert. Der Variablen pcb wird bei usedCPU ein quantum hinzugefügt. Sollte der Prozess noch nicht fertig sein, wird er der Queue (que) hinzugefügt. Anschließend wird pcb auf null gesetzt.

Genauere Betrachtung einer Methode:

Betrachte die Methode operate().

Quellen:

<https://de.wikipedia.org/wiki/Round_Robin_%28Informatik%29>

<https://searchcode.com/codesearch/view/28108541/>