

Aufgabe V1: Segmente

Im Rahmen dieser Aufgabe definiert eine ausführbare Datei die drei Segmente Code, Konstanten und statische Daten. Das gleiche gilt für eine dynamisch gebundene Bibliothek. Eine Instanz im Speicher benötigt außerdem noch eine Halde und einen Stapel. Der Programmlader hält unveränderbare Segmente nur einmal im Speicher. Er stellt sie allen Instanzen zur Verfügung, die sie benötigen.

- **ProgP** verwendet **libL**
- **ProgQ** verwendet **libL** und **libM**
- **ProgR** verwendet **libM** und **libX**

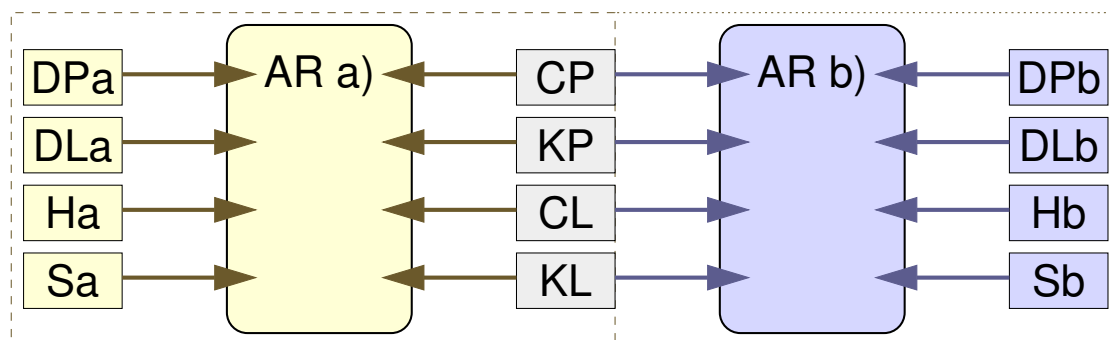
Geben sie für die folgenden Instanzen an, wie viele Adressräume, Einblendungen und Segmente der Programmlader neu erzeugt. Die Instanzen aus den vorhergehenden Teilaufgaben befinden sich jeweils schon bzw. noch im Speicher.

a) Eine Instanz von **ProgP** startet.

1 Adressraum, 8 Einblendungen, 8 Segmente. Code und Konstanten von P und L (CP KP CL KL), statische Daten von P und L (DPa DLa), Halde (Ha) und Stapel (Sa).

b) Eine zweite Instanz von **ProgP** startet.

1 Adressraum, 8 Einblendungen, 4 Segmente. Die zweite Instanz braucht ebenso viele Segmente wie die erste, deshalb gibt es gleich viele Einblendungen. Code und Konstanten von P und L sind bereits geladen (CP KP CL KL). Für die statischen Daten von P und L (DPb DLb), die Halde (Hb) und den Stapel (Sb) braucht die Instanz eigene Segmente.



weiter auf der nächsten Seite...

c) Eine Instanz von **ProgQ** startet.

1 Adressraum, 11 Einblendungen, 9 Segmente. **ProgQ** verwendet zwei dynamisch gebundene Bibliotheken und braucht deshalb drei Einblendungen mehr als **ProgP**. Code und Konstanten von **libL** (CL KL) befinden sich bereits im Speicher, die restlichen Segmente muss der Programmlader neu anlegen.

d) Eine Instanz von **ProgR** startet.

1 Adressraum, 11 Einblendungen, 9 Segmente. Die Situation ist die gleiche wie bei **ProgQ**: von den beiden dynamisch gebundenen Bibliotheken befindet sich eine bereits im Speicher, die andere noch nicht.

