

BETRIEBSSYSTEME - PRAKTIKUM 4

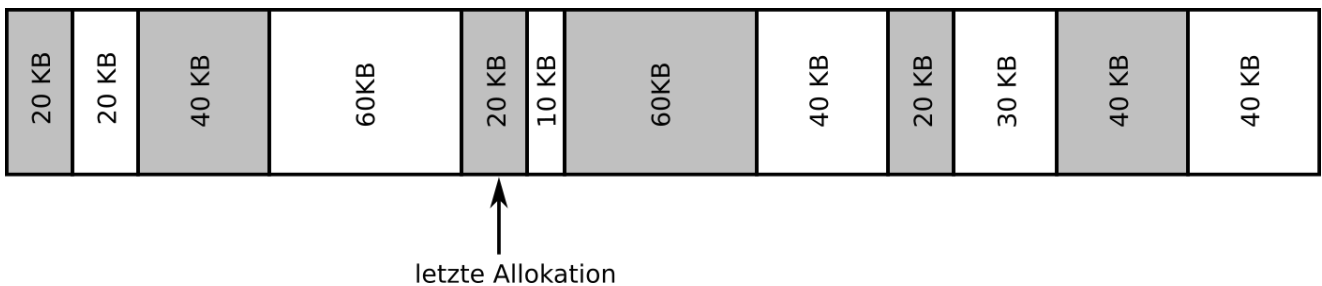
Sommersemester 2023

AUFGABE 1

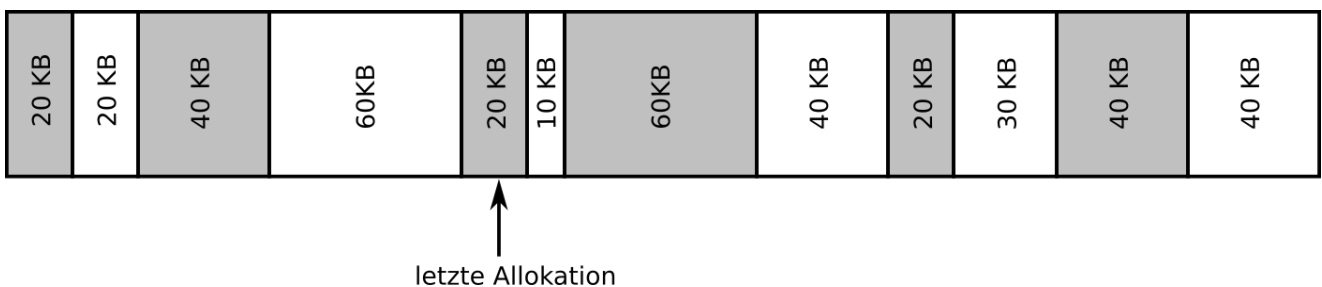
Im folgenden sind drei Diagramme für Speicherbelegung aufgelistet. Dunkle Bereiche symbolisieren belegten Speicher, helle Bereiche symbolisieren freien Speicher. Tragen Sie für alle Teilaufgaben in das jeweilige Diagramm ein, welche Speicherbereiche mit den aufgelisteten Anforderungen und der angegebenen Strategie verwendet werden

Dabei kommen die Speicheranforderungen nacheinander als (A), (B), und (C) in das Speichersystem. Wenn Anforderung (B) ankommt wurde also Speicher für (A) bereits allokiert. Wenn die Anforderung für (C) ankommt, dann wurde die Allokation von (A) und (B) bereits durchgeführt.

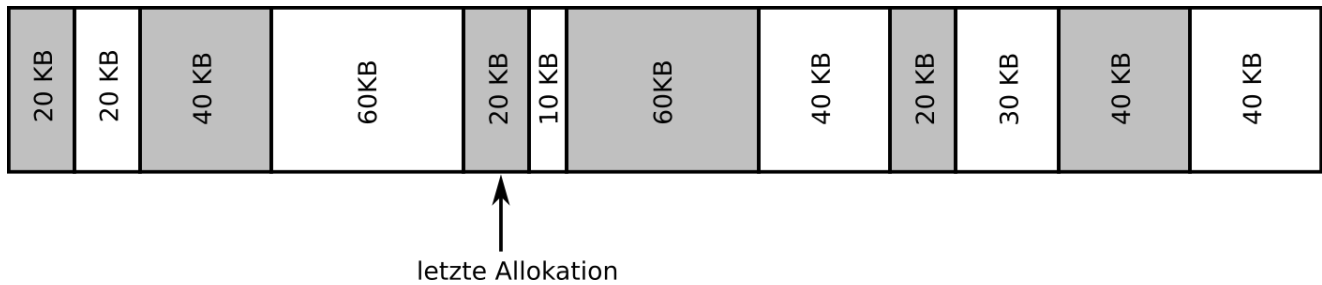
- a) Speicheranforderungen: (A) 10 KB, (B) 20 KB, (C) 40 KB
Strategie: First-fit



- b) Speicheranforderungen: (A) 20 KB, (B) 40 KB, (C) 10 KB
Strategie: Best-fit



- c) Speicheranforderungen: (A) 40 KB, (B) 10 KB, (C) 20 KB
Strategie: Next-fit



AUFGABE 2

Auf einem Rechner mit virtueller Speicherverwaltung beträgt die Seitengröße 1024 Bytes, und die Seitentabelle eines Prozesses enthält zu einem bestimmten Zeitpunkt die folgenden Einträge:

Virtuelle Seite	Present-Bit	R-Bit	M-Bit	Physikalischer Seitenrahmen
0	1	1	0	4
1	1	1	1	7
2	0	0	0	5
3	1	0	0	2
4	0	0	0	3
5	1	0	0	0

Bestimmen Sie zu den folgenden *virtuellen Adressen* jeweils die *physikalischen Adresse* und geben Sie an ob ein Seitenfehler beim Zugriff auftreten würde.

a) 1050:

b) 42:

c) 2100:

d) 5100:

e) 5200:

AUFGABE 3

Ein Rechner besitzt 4 physikalische Seitenrahmen (0..3). Die folgende Tabelle zeigt eine Folge von Zugriffen und resultierende Belegungen der Seitenrahmen. In der Spalte "Zugriff" ist jeweils die virtuelle Seitennummer zusammen mit der Zugriffsart angegeben (l = lesen, s = schreiben). In den Spalten zu den Seitenrahmen ist jeweils die virtuelle Seitennummer angegeben. Ein "r" markiert außerdem, dass das R-Bit gesetzt ist, ein "m" markiert ein gesetztes "Modified"-Bit

In den folgenden Teilaufgaben sollen Sie die Tabelle basierend auf den jeweiligen Seitenersetzungs-Strategien vervollständigen.

a) Strategie: FIFO

Zugriff	Seitenrahmen			
	0	1	2	3
0 l	0 r	-	-	-
3 s	0 r	3 r m	-	-
1 l	0 r	3 r m	1 r	-
2 s	0 r	3 r m	1 r	2 r m
0 s				
4 l				
0 l				
3 l				
1 l				

b) Strategie: Least-Recently Used

Zugriff	Seitenrahmen			
	0	1	2	3
0 l	0 r	-	-	-
3 s	0 r	3 r m	-	-
1 l	0 r	3 r m	1 r	-
2 s	0 r	3 r m	1 r	2 r m
0 s				
4 l				
0 l				
3 l				
1 l				

- c) Strategie: Erweiterter Uhrzeiger-Algorithmus (Unterstrichene Seite bedeutet, dass hier der Zeiger steht)

Zugriff	Seitenrahmen			
	0	1	2	3
0 l	<u>0</u> r	-	-	-
3 s	<u>0</u> r	3 r m	-	-
1 l	<u>0</u> r	3 r m	1 r	-
2 s	<u>0</u> r	3 r m	1 r	2 r m
0 s				
4 l				
0 l				
3 l				
1 l				

AUFGABE 4

Laden Sie sich das Rahmenprogramm *swapper.zip* herunter und kompilieren sie es. Das Rahmenprogramm enthält eine Simulation einer virtuellen Speicherverwaltung, die den Mechanismus der letzten Aufgabe implementiert. In dieser Aufgabe soll der *Erweiterte Uhrzeigeralgorithmus* in diesem Rahmen implementiert werden.

- a) Lesen Sie sich die Datei *swapper.h* durch und machen Sie sich mit den Strukturen und Makros vertraut:

Die Strukturen *virtual_page_entry* und *physical_frame_entry* enthalten die Verwaltungsstrukturen zur Verwaltung der virtuellen Seiten und der physikalischen Seitenrahmen. Das Rahmenprogramm übernimmt die Zuordnung von virtueller Seite zu physikalischem Rahmen und sorgt dafür, dass die Zuordnung immer korrekt ist. Es setzt beim Speicherzugriff auch das R-Bit und das M-Bit. Dazu existieren die Makros *SET_BIT* und *CLEAR_BIT*. Mit Hilfe von *IS_BIT_SET* kann geprüft werden, ob eines der Bits gesetzt ist.

Falls für das Rahmenprogramm keinen freien physikalischen Speicher mehr zur Verfügung hat, ruft es eine Methode auf die einen physikalischen Rahmen identifiziert der ausgelagert werden kann. Dieser wird in der Datei *swapper.c* ausgewählt:

```
46 struct physical_frame_entry* (*swap_strategy)(struct physical_frame_entry*) = &not_implemented_strategy;
47 //struct physical_frame_entry* (*swap_strategy)(struct physical_frame_entry*) = &stupid_swapper;
48 //struct physical_frame_entry* (*swap_strategy)(struct physical_frame_entry*) = &extended_clock_swap_strategy;
```

Zu Beginn ist die Strategie *not_implemented_strategy* ausgewählt. Diese führt zu einem Abbruch des Programms. Sie können die passende Strategie auswählen, indem Sie die passenden Funktionen in *swapper.c* aktivieren.

Die Strategien sind in der Datei *swap_strategies.c* implementiert. Die Funktion *stupid_swapper* zeigt Ihnen, wie eine unsinnige Implementierung einer Strategie aussieht.

- b) Aktivieren Sie die Strategie *extended_clock_swap_strategy* und implementieren Sie diese in der Datei *swap_strategies.c*. Die Speicherzugriffe des Rahmenprogramms sind die gleichen wie in der vorherigen Aufgaben. Sie sollten also am Ende die gleiche Ausgabe bekommen, wie bei Ihrer Lösung der vorherigen Aufgabe.