Aufgabe 1.1: Einfachauswahl-Fragen (22 Punkte)

Bei den Multiple-Choice-Fragen in dieser Aufgabe ist jeweils nur <u>eine</u> richtige Antwort eindeutig anzukreuzen. Auf die richtige Antwort gibt es die angegebene Punktzahl.

Wollen Sie eine Multiple-Choice-Antwort korrigieren, streichen Sie bitte die falsche Antwort mit drei waagrechten Strichen durch (**) und kreuzen die richtige an.

Lesen Sie die Frage genau, bevor Sie antworten!

a)		an unterscheidet die Begriffe Programm und Prozess. Welche der fol- nden Aussagen zu diesem Thema ist richtig?	2 Punkte
		Ein Prozess kann durch mehrere Programme ausgeführt werden.	
		Mit Hilfe des Systemaufrufs <i>exec</i> wird das bestehende Programm im aktuell laufenden Prozess ersetzt.	
		Der Binder erzeugt aus einer oder mehreren Objekt-Dateien einen Proz	zess.
		Der Prozess ist der statische Teil (Rechte, Speicher, etc.), das Prograaktive Teil (Programmzähler, Register, Stack).	amm der
b)	Ma	an unterscheidet Traps und Interrupts. Welche Aussage ist richtig?	2 Punkte
		Das Betriebssystem kann Interrupts, die in ursächlichem Zusammenhang mit dem gerade laufenden Prozess stehen, nach dem Beendigungsmodell behandeln, wenn eine sinnvolle Fortführung des Prozesses nicht mehr möglich ist	
		Der Zugriff auf eine physikalische Adresse kann zu einem Trap führen	ı .
		Traps dürfen nicht nach dem Wiederaufnahmemodell behandelt werde Trap immer einen schwerwiegenden Fehler signalisiert.	n, da ein
		Ganzzahl-Rechenoperationen können nicht zu einem Trap führen.	
c)	We	elche Aussage zum Thema Threads ist richtig?	2 Punkte
		Zu jedem Kernel-Thread gehört ein eigener isolierter Adressraum.	
		Kernel-Threads können Multiprozessoren nicht ausnutzen.	
		Userlevel-Threads blockieren sich bei blockierenden Systemaufrufen g tig.	gegensei-
		Zur Umschaltung von Userlevel-Threads ist ein Adressraumwechsel elich.	erforder-

d)	We	lche Aussage zum Thema Adressraumschutz ist richtig?	2 Punkte
		Beim Adressraumschutz durch Abteilung wird der logische Adressraum in mehrere Segmente mit unterschiedlicher Semantik unterteilt.	
		Adressraumschutz durch Eingrenzung benötigt keinen verschiebende um Programmadressen an Arbeitsspeicheradressen zu binden	en Lader,
		Beim Einsatz von Segmentierung ist es möglich, dass dieselbe logische in unterschiedlichen logischen Adressräumen auf unterschiedliche psche Adressen verweist.	
		In einem segmentierten Adressraum kann zur Laufzeit kein weiterer mehr dynamisch nachgefordert werden.	Speicher
e)	We	Iche Aussage zum Thema Betriebsarten ist richtig?	2 Punkte
		Mehrzugangsbetrieb ist nur in Verbindung mit CPU- und Speicherschutz sinnvoll realisierbar.	
		Beim Stapelbetrieb können keine globalen Variablen existieren, weil a im Stapel-Segment (Stack) abgelegt sind.	lle Daten
		Mehrprogrammbetrieb ermöglicht die simultane Ausführung mehre gramme innerhalb desselben Prozesses.	erer Pro-
		Echtzeitsysteme findet man hauptsächlich auf großen Serversystemen enorme Menge an Anfragen zu bearbeiten haben.	, die eine
f)	We	Iche Aussage zum Thema Speicherverwaltung ist richtig?	2 Punkte
		Sollte bei einer Speicheranforderung mittels malloc nicht genügend Speicher vorhanden sein, wird der Prozess vom Betriebssystem beendet.	
		Ohne spezielle Unterstützung durch das Betriebssystem kann das Lautem nur eine vorab statisch festgelegte Menge an Speicher feingranulaten.	
		Speicherbereiche, die vor Beendigung des Prozesses nicht mit free fre wurden, sind bis zum Neustart des Systems unwiederbringlich verlore	
		Speicherbereiche, die im logischen Adressraum zusammenhängend si sen auch im physikalischen Hauptspeicher zusammenhängend sein.	nd, müs-

	sich außerhalb des eigenen Stack-Frames befinden.
	Bei rekursiven Funktionsaufrufen kann der Speicher des Stack-Frames in jedem Fall wiederverwendet werden, weil die gleiche Funktion aufgerufen wird.
	Ein Pufferüberlauf eines lokalen Feldes kann sicherheitskritisch sein, da er unter Umständer zur Ausführung beliebigen Codes führt.
	Der Compiler legt zur Übersetzungszeit fest, an welcher Position im Stack-Frame der main-Funktion die globalen Variablen angelegt werden.
)	tualisierung kann als Maßnahme gegen Verklemmungen genutzt 2 Punkterden. Warum?
	Im Fall einer Verklemmung können zusätzliche virtuelle Betriebsmittel neu erzeugt werden. Diese können dann eingesetzt werden, um die fehlenden physikalischen Betriebsmittel zu ersetzen.
	Durch Virtualisierung ist ein Entzug von physikalischen Betriebsmitteln möglich, obwohl dies auf der logischen Ebene unmöglich ist.
	Eine Verklemmungsauflösung ist einfacher, weil virtuelle Betriebsmittel jederzeit ohne Schaden entzogen werden können.
	Durch Virtualisierung kann man über Abbildungsvorgänge Zyklen, die auf der logischen Ebene vorhanden sind, auf der physikalischen Ebene auflösen.

j) Für lokale Variablen, Aufrufparameter, etc. einer Funktion wird bei vie-

len Prozessoren ein Stack-Frame angelegt. Welche Aussage ist richtig?

☐ Es ist nicht möglich auf lokale *automatic*-Variablen zuzugreifen, die

Juli 2014

2 Punkte

Klausur Systemprogrammierung

☐ First-Come-First-Served ist nur im Stapelbetrieb bei lang laufenden Aufträgen

☐ Virtual-Round-Robin benachteiligt E/A-intensive Prozesse zu Gunsten von

sinnvoll einsetzbar.

rechenintensiven Prozessen.

Klausur Systemprogrammierung Juli 2014

Aufgabe 1.2: Mehrfachauswahl-Fragen (8 Punkte)

Bei den Multiple-Choice-Fragen in dieser Aufgabe sind jeweils m Aussagen angegeben, n ($0 \le n \le m$) Aussagen davon sind richtig. Kreuzen Sie **alle richtigen** Aussagen an. Jede korrekte Antwort in einer Teilaufgabe gibt einen halben Punkt, jede falsche Antwort einen halben Minuspunkt. Eine Teilaufgabe wird minimal mit 0 Punkten gewertet, d. h. falsche Antworten wirken sich nicht auf andere Teilaufgaben aus.

Wollen Sie eine falsch angekreuzte Antwort korrigieren, streichen Sie bitte das Kreuz mit drei waagrechten Strichen durch (囊).

Lesen	Sie die Frage genau, bevor Sie antworten!
a) We	elche der folgenden Aussagen zum Thema Prozesszustände sind rich- ?
0	Greift ein laufender Prozess lesend auf eine Datei zu und der entsprechende Datenblock ist nicht im Hauptspeicher vorhanden, dann wird der Prozess in den Zustand <i>bereit</i> überführt.
0	Bei kooperativem Scheduling ist kein direkter Übergang vom Zustand <i>laufend</i> in den Zustand <i>bereit</i> möglich.
О	Es können sich maximal genauso viele Prozesse gleichzeitig im Zustand laufend befinden, wie Prozessorkerne vorhanden sind.
0	Die V-Operation eines Semaphors kann bewirken, dass ein Prozess vom Zustand <i>blockiert</i> in den Zustand <i>bereit</i> überführt wird.
0	Im Rahmen der mittelfristigen Einplanung kann ein Prozess von Zustand <i>laufend</i> in den Zustand <i>schwebend laufend</i> wechseln.
0	Ein Prozess kann nur durch seine eigene Aktivität vom Zustand <i>laufend</i> in den Zustand <i>blockiert</i> überführt werden.
0	Bei Eintreffen eines Interrupts wird der aktuell laufende Prozess für die Dauer der Interrupt-Abarbeitung in den Zustand <i>blockiert</i> überführt.
0	Das Auftreten eines Seitenfehlers kann dazu führen, dass der aktuell laufende Prozess in den Zustand <i>beendet</i> überführt wird.

Klausur Systemprogrammierung Juli 2014

b)	We	lche der folgenden Aussagen zu UNIX-Dateisystemen sind richtig? 4 Punkte
	О	In einem Verzeichnis darf es mehrere Einträge mit identischem Namen geben, sofern sie auf unterschiedliche <i>Inodes</i> verweisen.
	О	Für das Löschen einer Datei sind die Rechte-Informationen im $Inode$ des Verzeichnisses, das die Datei enthält, irrelevant.
	О	Nach dem Löschen eines $Inodes$ und der dazugehörigen Datenblöcke ist garatiert, dass kein $hard\ link$ mehr darauf verweist.
	О	$\label{eq:linear_energy} \mbox{Ein } \textit{Inode} \mbox{ kann im Dateisystem nicht } \mbox{\"{u}ber mehrere Namen referenziert werden.}$
	О	Beim lesenden Zugriff auf eine Datei über einen <i>symbolic link</i> kann ein Prozess den Fehler <i>Permission denied</i> erhalten, obwohl er das Leserecht auf dem <i>symbolic link</i> besitzt.
	О	Der Inode einer Datei wird getrennt von ihrem Inhalt auf der Platte gespeichert.
	О	Ein <i>Inode</i> enthält u.a. die Anzahl der <i>symbolic links</i> , die auf ihn verweisen.
	О	Die Anzahl der <i>hard links</i> , die auf ein Verzeichnis verweisen, hängt von der Anzahl seiner Unterverzeichnisse ab.

Klausur Systemprogrammierung

Aufgabe 2: vsc (60 Punkte)

Sie dürfen diese Seite zur besseren Übersicht bei der Programmierung heraustrennen!

Schreiben Sie ein Programm vsc (video stream cache), das einen eingehenden Videostrom puffert und auf mehreren Anzeigegeräten (Framebuffer) anzeigt.

Für das Einlesen des Videostroms vom Client und für die Ausgabe auf den Anzeigegeräten ist jeweils ein eigener Thread zuständig. Der Datenaustausch zwischen den beiden Threads erfolgt mit Hilfe eines jitter-Puffers, der im Modul jitbuf zu implementieren ist.

Der jitter-Puffer hat zwei Modi: *Puffern* und *Durchzug*. Im initialen Modus *Puffern* werden die Videobilder im Puffer gespeichert, die Entnahme ist noch nicht erlaubt und die Entnahmeoperation blockiert. Im Modus *Durchzug* ist die Entnahme von Videobildern erlaubt, bei leerem Puffer blockiert die Entnahmeoperation.

Der Puffer soll bis zu 100 Videobilder (CACHE_SIZE) speichern können. Bei erstmaligem Erreichen eines Füllstands von 20 Videobildern (BUFFER_SIZE) wechselt er in den Modus *Durchzug*.

Der jitter-Puffer soll folgendermaßen arbeiten:

- Funktion int jbInit (void): Führt notwendige Initialisierungen durch. Sollte ein Fehler auftreten gibt die Funktion 1 zurück, ansonsten 0.
- Funktion void jbPut (const struct data* data): Kopiert das übergebene Videobild in den Puffer. Sobald die Grenze von 20 Videobildern erreicht wird, wird in den Modus Durchzug gewechselt.
- Funktion void jbGet(struct data* data): Kopiert ein Videobild aus dem Puffer. Blockiert solange sich der Puffer nicht im Modus Durchzug befindet.
- Funktion void jbFlush (void): Kann dazu verwendet werden, den Modus des Puffers vor Erreichen der Grenze von 20 Videobildern auf Durchzug zu schalten.

Das Programm soll folgendermaßen arbeiten:

- Nach dem Starten des Anzeige-Threads nimmt der Haupt-Thread eine Verbindung auf Port 2014 an und liest von dieser Verbindung blockweise (Bibliotheksfunktion fread) Videobilder der Größe 307200 (640x480) Bytes ein und fügt diese jeweils als data-Struktur (siehe Codegerüst. Seite 10 oben) in den Puffer ein. Wenn der Client die Verbindung beendet, signalisiert er dies dem Anzeige-Thread über eine data-Struktur, wartet auf dessen Terminierung und beendet sich dann.
- Der Anzeige-Thread startet in der Funktion void *showFrame(void);
 Diese öffnet die ersten maximal 10 Einträge im Verzeichnis /dev deren Name mit fb beginnt. Danach entnimmt der Thread laufend Videobilder aus dem Puffer und gibt diese auf jedem geöffneten Anzeigegerät aus (Bibliotheksfunktion fwrite). Der Anzeige-Thread beendet sich, nachdem er das letzte Bild ausgegeben hat.

Synchronisieren Sie den Ringpuffer mit zählenden Semaphoren.

```
Schnittstelle: SEM *semCreate(int initial_value);
     void P(SEM *sem);
     void V(SEM *sem);
     void semDestrov(SEM *sem);
```

Die Semaphor-Funktionalität ist in einem separaten Modul sem bereits implementiert.

Auf den folgenden Seiten finden Sie ein Gerüst für das beschriebene Programm. In den Kommentaren sind nur die wesentlichen Aufgaben der einzelnen zu ergänzenden Programmteile beschrieben, um Ihnen eine gewisse Leitlinie zu geben. Es ist überall sehr großzügig Platz gelassen, damit Sie auch weitere notwendige Anweisungen entsprechend Ihrer Programmierung einfügen können.

Einige wichtige Manual-Seiten liegen bei - es kann aber durchaus sein, dass Sie bei Ihrer Lösung nicht alle diese Funktionen oder gegebenenfalls auch weitere Funktionen benötigen.

/*************************************	
* Zusatziich zu Funktionsdeklarationen in Datei Jitbur.n ************************************	
#define FRAME_SIZE 640*480	
struct data {	
char frame[FRAME_SIZE];	
int eof;	
}; /*******************************	
* Datei vsc.c	

#include <dirent.h></dirent.h>	
#include <errno.h></errno.h>	
#include <pthread.h></pthread.h>	
#include <stdio.h></stdio.h>	
#include <stdlib.h></stdlib.h>	
#include <string.h></string.h>	
#include <netinet in.h=""></netinet>	
#include <sys socket.h=""></sys>	
#include <sys stat.h=""></sys>	
#include "jitbuf.h"	
exit(EXIT_FAILURE); } // weitere Includes, Konstanten // globale Variablen, Funktionsdeklarationen usw.	
// Funktion main()	
	Δ.

// Socket erstellen und Verbindungsannahme vorbereiten	
// Verbindung annehmen und Verarbeitung vorbereiten	
// Verbindung annehmen und Verarbeitung vorbereiten	
	S:

// Ende Funktion main

Klausur Systemprogrammierung

Klausur Systemprogrammierung

/****************	
* Datei jitbuf.c ************************************	
#define CACHE_SIZE 100 #define BUFFER_SIZE 20	
<pre>static volatile size_t readPos; static volatile size_t writePos; static SEM* fullSlots; static SEM* freeSlots; static SEM* waiter; static size_t itemsBuffered; static struct data cache[CACHE_SIZE];</pre>	
// Funktion jbInit	
	_

// Funktion jbPut	
// Funktion jbGet	
// Funktion jbFlush	
	J:

Aufgabe 3: (22 Punkte)

Zur Koordinierung von nebenläufigen Vorgängen, die auf gemeinsame Betriebsmittel zugreifen, unterscheidet man zwischen einseitiger und mehrseitiger Synchronisation.

a)	Beschreiben Sie die Unterschiede zwischen einseitiger und mehrseitiger Synchronisation. (2 Punkte)	
b)	Beschreiben Sie für einseitige und mehrseitige Synchronisation jeweils eine typische Situation, in der ein Synchronisationsproblem der jeweiligen Art auftritt. Wählen Sie dazu jeweils ein geeignetes Synchronisationsverfahren und skizzieren Sie in Pseudocode den Ablauf der jeweiligen Synchronisationssituation. (9 P.)	
		L

c)	Synchronisation erfolgt in sehr vielen Fällen mittels blockierender Verfahren. Welche Probleme sind mit dem Einsatz blockierender Synchronisation verbunden? (4 Punkte)
d)	In manchen Situationen kann man statt mit blockierender Synchronisation mit opti-
d)	In manchen Situationen kann man statt mit blockierender Synchronisation mit optimistischen, nichtblockierenden Synchronisationsverfahren arbeiten. Beschreiben Sie an einem kleinen Beispiel (z.B. unter Verwendung eines CAS-Befehls), wie solch ein Verfahren vom Prinzip her arbeitet. Nennen Sie jeweils zwei Vor- und Nachteile solcher Verfahren. (7 Punkte)
d)	mistischen, nichtblockierenden Synchronisationsverfahren arbeiten. Beschreiben Sie an einem kleinen Beispiel (z.B. unter Verwendung eines CAS-Befehls), wie solch ein Verfahren vom Prinzip her arbeitet. Nennen Sie jeweils zwei Vor- und
d)	mistischen, nichtblockierenden Synchronisationsverfahren arbeiten. Beschreiben Sie an einem kleinen Beispiel (z.B. unter Verwendung eines CAS-Befehls), wie solch ein Verfahren vom Prinzip her arbeitet. Nennen Sie jeweils zwei Vor- und Nachteile solcher Verfahren. (7 Punkte)
d)	mistischen, nichtblockierenden Synchronisationsverfahren arbeiten. Beschreiben Sie an einem kleinen Beispiel (z.B. unter Verwendung eines CAS-Befehls), wie solch ein Verfahren vom Prinzip her arbeitet. Nennen Sie jeweils zwei Vor- und Nachteile solcher Verfahren. (7 Punkte)
d)	mistischen, nichtblockierenden Synchronisationsverfahren arbeiten. Beschreiben Sie an einem kleinen Beispiel (z.B. unter Verwendung eines CAS-Befehls), wie solch ein Verfahren vom Prinzip her arbeitet. Nennen Sie jeweils zwei Vor- und Nachteile solcher Verfahren. (7 Punkte)
d)	mistischen, nichtblockierenden Synchronisationsverfahren arbeiten. Beschreiben Sie an einem kleinen Beispiel (z.B. unter Verwendung eines CAS-Befehls), wie solch ein Verfahren vom Prinzip her arbeitet. Nennen Sie jeweils zwei Vor- und Nachteile solcher Verfahren. (7 Punkte)
d)	mistischen, nichtblockierenden Synchronisationsverfahren arbeiten. Beschreiben Sie an einem kleinen Beispiel (z.B. unter Verwendung eines CAS-Befehls), wie solch ein Verfahren vom Prinzip her arbeitet. Nennen Sie jeweils zwei Vor- und Nachteile solcher Verfahren. (7 Punkte)
d)	mistischen, nichtblockierenden Synchronisationsverfahren arbeiten. Beschreiben Sie an einem kleinen Beispiel (z.B. unter Verwendung eines CAS-Befehls), wie solch ein Verfahren vom Prinzip her arbeitet. Nennen Sie jeweils zwei Vor- und Nachteile solcher Verfahren. (7 Punkte)
d)	mistischen, nichtblockierenden Synchronisationsverfahren arbeiten. Beschreiben Sie an einem kleinen Beispiel (z.B. unter Verwendung eines CAS-Befehls), wie solch ein Verfahren vom Prinzip her arbeitet. Nennen Sie jeweils zwei Vor- und Nachteile solcher Verfahren. (7 Punkte)
d)	mistischen, nichtblockierenden Synchronisationsverfahren arbeiten. Beschreiben Sie an einem kleinen Beispiel (z.B. unter Verwendung eines CAS-Befehls), wie solch ein Verfahren vom Prinzip her arbeitet. Nennen Sie jeweils zwei Vor- und Nachteile solcher Verfahren. (7 Punkte)
d)	mistischen, nichtblockierenden Synchronisationsverfahren arbeiten. Beschreiben Sie an einem kleinen Beispiel (z.B. unter Verwendung eines CAS-Befehls), wie solch ein Verfahren vom Prinzip her arbeitet. Nennen Sie jeweils zwei Vor- und Nachteile solcher Verfahren. (7 Punkte)
d)	mistischen, nichtblockierenden Synchronisationsverfahren arbeiten. Beschreiben Sie an einem kleinen Beispiel (z.B. unter Verwendung eines CAS-Befehls), wie solch ein Verfahren vom Prinzip her arbeitet. Nennen Sie jeweils zwei Vor- und Nachteile solcher Verfahren. (7 Punkte)
d)	mistischen, nichtblockierenden Synchronisationsverfahren arbeiten. Beschreiben Sie an einem kleinen Beispiel (z.B. unter Verwendung eines CAS-Befehls), wie solch ein Verfahren vom Prinzip her arbeitet. Nennen Sie jeweils zwei Vor- und Nachteile solcher Verfahren. (7 Punkte)
d)	mistischen, nichtblockierenden Synchronisationsverfahren arbeiten. Beschreiben Sie an einem kleinen Beispiel (z.B. unter Verwendung eines CAS-Befehls), wie solch ein Verfahren vom Prinzip her arbeitet. Nennen Sie jeweils zwei Vor- und Nachteile solcher Verfahren. (7 Punkte)

Aufgabe 4: (8 Punkte)

ben Sie eine Situation		nlern (<i>page faults</i>) kommen. enfehler auftritt und was das kte)	
gelagert werden. Eine Seite ist LRU. Beschr	e mögliche Strategie zu	her verfügbar ist, muss eine ir Bestimmung der zu verdi ie kurz und erläutern Sie, wete)	rängenden
gelagert werden. Eine Seite ist LRU. Beschr	e mögliche Strategie zu reiben Sie diese Strateg	ur Bestimmung der zu verdr ie kurz und erläutern Sie, w	rängenden
gelagert werden. Eine Seite ist LRU. Beschr	e mögliche Strategie zu reiben Sie diese Strateg	ur Bestimmung der zu verdr ie kurz und erläutern Sie, w	rängenden
gelagert werden. Eine Seite ist LRU. Beschr	e mögliche Strategie zu eiben Sie diese Strateg erung bestehen. (2 Punk	ur Bestimmung der zu verdr ie kurz und erläutern Sie, w	rängenden elche Pro-
gelagert werden. Eine Seite ist LRU. Beschr	e mögliche Strategie zu eiben Sie diese Strateg erung bestehen. (2 Punk	nr Bestimmung der zu verdi ie kurz und erläutern Sie, w te)	rängenden elche Pro-
gelagert werden. Eine Seite ist LRU. Beschr bleme bei der Realisie	e mögliche Strategie zu eiben Sie diese Strateg erung bestehen. (2 Punk	ur Bestimmung der zu verdie kurz und erläutern Sie, waten	rängenden elche Pro-
gelagert werden. Eine Seite ist LRU. Beschr bleme bei der Realisie	e mögliche Strategie zu eiben Sie diese Strateg erung bestehen. (2 Punk	ur Bestimmung der zu verdie kurz und erläutern Sie, wete)	rängenden elche Pro-
gelagert werden. Eine Seite ist LRU. Beschr bleme bei der Realisie	e mögliche Strategie zu eiben Sie diese Strateg erung bestehen. (2 Punk	ur Bestimmung der zu verdie kurz und erläutern Sie, waten	rängenden elche Pro-
gelagert werden. Eine Seite ist LRU. Beschr bleme bei der Realisie	e mögliche Strategie zu eiben Sie diese Strateg erung bestehen. (2 Punk	ur Bestimmung der zu verdie kurz und erläutern Sie, wete)	rängenden elche Pro-
gelagert werden. Eine Seite ist LRU. Beschr bleme bei der Realisie	e mögliche Strategie zu eiben Sie diese Strateg erung bestehen. (2 Punk	ur Bestimmung der zu verdie kurz und erläutern Sie, waten	rängenden elche Pro-
gelagert werden. Eine Seite ist LRU. Beschr bleme bei der Realisie	e mögliche Strategie zu eiben Sie diese Strateg erung bestehen. (2 Punk	ir Bestimmung der zu verdie kurz und erläutern Sie, wete)	rängenden elche Pro-
gelagert werden. Eine Seite ist LRU. Beschr bleme bei der Realisie	e mögliche Strategie zu eiben Sie diese Strateg erung bestehen. (2 Punk	ur Bestimmung der zu verdie kurz und erläutern Sie, waten	rängenden elche Pro-
gelagert werden. Eine Seite ist LRU. Beschr bleme bei der Realisie	e mögliche Strategie zu eiben Sie diese Strateg erung bestehen. (2 Punk	ir Bestimmung der zu verdie kurz und erläutern Sie, wete)	rängenden elche Pro-
gelagert werden. Eine Seite ist LRU. Beschr bleme bei der Realisie	e mögliche Strategie zu eiben Sie diese Strateg erung bestehen. (2 Punk	ir Bestimmung der zu verdie kurz und erläutern Sie, wete)	rängenden elche Pro-

c)	Eine in der Praxis gut einsetzbare Approximation dieser Strategie ist Second Chance (oder Clock). Beschreiben Sie die Funktionalität dieser Strategie und welche Unterstützung durch die MMU dafür erforderlich ist. (4 Punkte)