

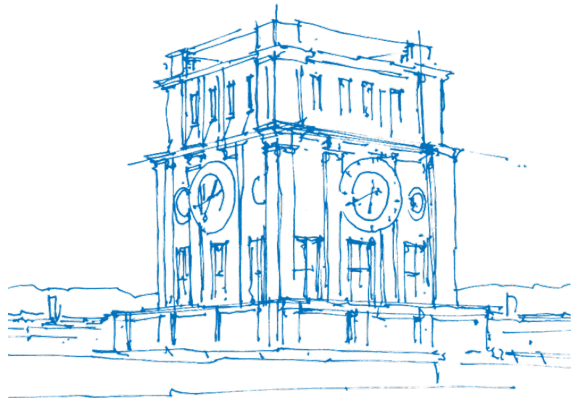
# Grundlagen: Betriebssysteme und Systemsoftware

## Tutorübung

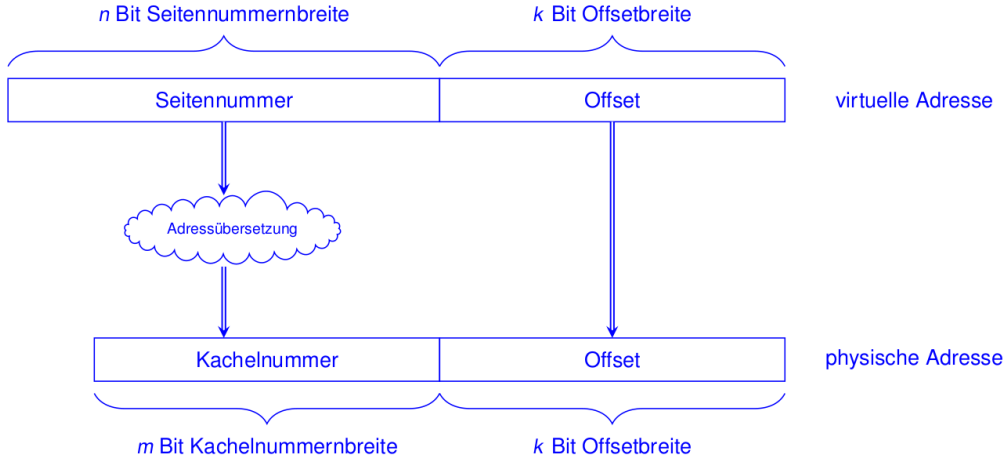
**Mario Delic**

Lehrstuhl für Connected Mobility  
School of Computation, Information and Technology  
Technische Universität München

Übungswoche 10



*TUM Uhrenturm*



# Paging Basics

- **Swapping**

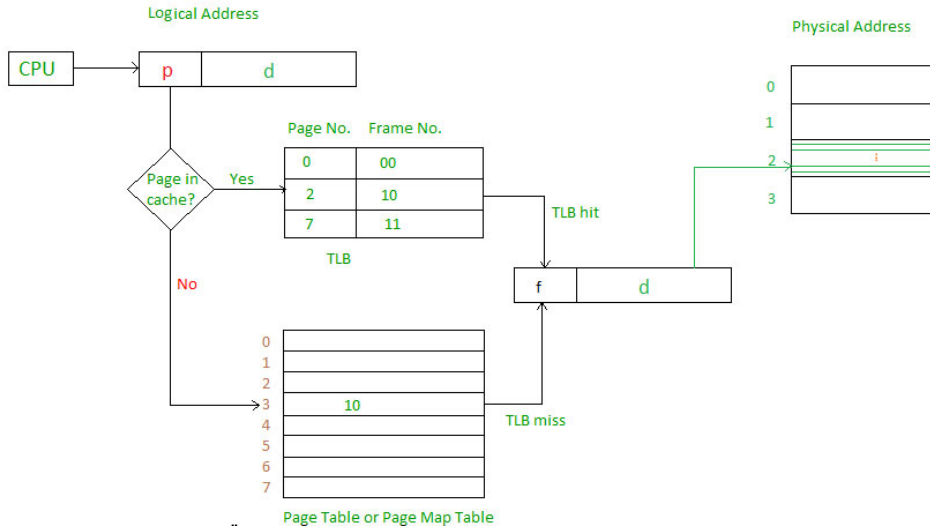
Auslagern der **gesamten** Prozessdaten auf den Hintergrundspeicher.



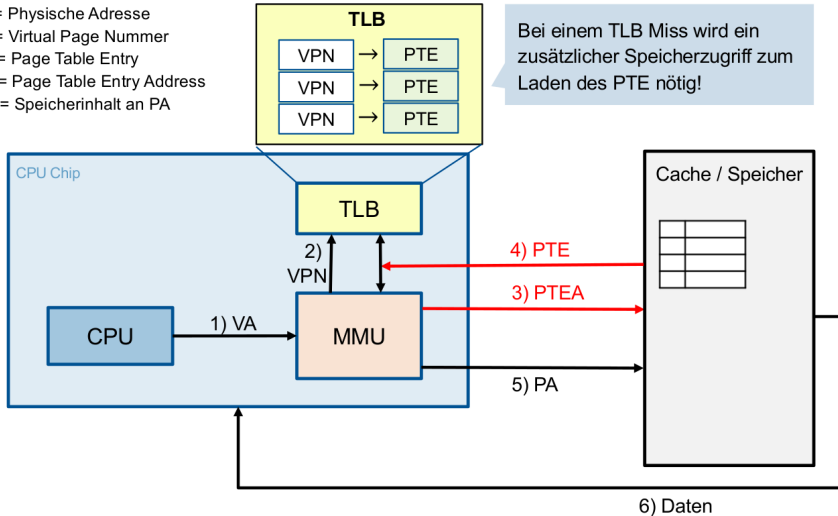
- **Paging**

**Seitenweises** Auslagern auf den Hintergrundspeicher (einzelne Prozessdaten können separat ausgelagert werden).

- **Seiten** sind **virtuell**, **Kacheln** sind **physisch**
- **Page Fault**: Wenn **Present-Bit** der Seite nicht gesetzt ist raised die MMU einen PF  
↪ Interrupt → Page Fault Handler → ggf. einlagern, PTE updaten → Interrupt-Kontext verlassen
- **Memory Management Unit (MMU)** ist für die Adressübersetzung zuständig
- MMU hat eigenen Cache: **Translation-Lookaside-Buffer (TLB)**



VA = Virtuelle Adresse  
 PA = Physische Adresse  
 VPN = Virtual Page Nummer  
 PTE = Page Table Entry  
 PTEA = Page Table Entry Address  
 Daten = Speicherinhalt an PA



# Erinnerung

- **Offset-bits ermitteln:**

Über Kachel-/Seitengröße (adressierbare Bytes):

4 KiB große Seite  $\rightarrow$  4 KiB =  $2^{12}$  Adressierbare **Bytes**  $\rightarrow$  12-bit Offset

- **Seitennummer-bits ermitteln:**

Über Anzahl an Seitentabelleneinträgen:

512 Seitentabelleneinträge =  $2^9$  Seitentabelleneinträge  $\rightarrow$  9 Bits für Seitennummer

- **Mehrstufige Seitentabellen:**

Einstufig  $v=(s,w)$ :

Seitentabelle mit Seitennummer  $s \rightarrow$  Übersetzung der  $v_a$  zu  $p_a$

Zweistufig  $v=(s_1,s_2,w)$ :

Seitentabellennummer  $s_1 \rightarrow$  Seitentabelle mit Seitennummern  $s_2 \rightarrow$  Übersetzung der  $v_a$  zu  $p_a$

# Erinnerung

i-node



- **Blockgröße ermitteln:**

Speichergröße ÷ Blockzahl → 8 MiB Speicher, 2048 Blöcke:  $2^{23}/2^{11} = 2^{12} = 4 \frac{\text{KiB}}{\text{Block}}$

- **Elemente pro Struktur:**

Strukturgröße ÷ Elementlänge → Blockgröße 4 KiB, 64-bit Adresslänge →  $2^{12}/2^3 = 2^9$   
Elemente

- **Indirect (=mehrstufige) Blöcke:**

Single-Indirect: Eine Adresse → Block mit Adressen → Datenblöcke

Double-Indirect: Eine Adresse → Block mit Adressen → Block mit Adressen → Datenblöcke

Wie viele Speicherblöcke mit 1 KiB BG, 32-bit Adr.L.

Verweis auf → Block (erste Stufe) mit  $2^{10}/2^2 = 2^8 = 256$  Verweise auf → Blöcke (zweite Stufe) mit nochmal  $2^8 = 256$  Verweise auf → Speicherblöcke = **256 \* 256**

## Aufgabe 2

### Seitenersetzungsstrategien

- **FIFO**: Als erstes eingelagerte Seite wird als erstes ersetzt.

- **LRU (Least-Recently-Used)**:

Die Seite, deren Zugriff am längsten in der Vergangenheit liegt wird ausgelagert.

↪ Speichern des Zeitpunkts  $t$  an dem letzter Zugriff erfolgte zus. zur Seitennummer.

- **NFU (Not-Frequently-Used)**:

Die aktuell am wenigsten genutzt Seite wird ausgelagert.

↪ Speichern der Häufigkeit  $\#$ , welche die Anzahl der Zugriffe seit Einlagerung darstellt.



Anfrage	$f_1$	$f_2$	$f_3$	$f_4$	Pagefaults
$s_i$ 1	1				1
3	1	2			2
5	1	3	5		3
4	1	3	5	4	4
2	2	3	5	4	5
4	2	3	5	4	5
3	2	3	5	4	5
2	2	3	5	4	5
1	2	1	5	4	6
0	2	1	0	4	7
5	2	1	0	5	8
3	3	1	0	5	9

Anfrage	$f_1, t$	$f_2, t$	$f_3, t$	$f_4, t$	Nr Pagefaults
1	<b>1,1</b>				<b>1</b>
3	1,1	<b>3,2</b>			<b>2</b>
5	1,1	3,2	<b>5,3</b>		<b>3</b>
4	1,1	3,2	5,3	<b>4,4</b>	<b>4</b>
2	<b>2,5</b>	3,2	5,3	4,4	<b>5</b>
4	2,5	3,2	5,3	<b>4,6</b>	5
3	2,5	<b>3,7</b>	5,3	4,6	5
2	<b>2,8</b>	3,7	5,3	4,6	5
1	2,8	3,7	<b>1,9</b>	4,6	<b>6</b>
0	2,8	3,7	1,9	<b>0,10</b>	<b>7</b>
5	2,8	<b>5,11</b>	1,9	0,10	<b>8</b>
3	<b>3,12</b>	5,11	1,9	0,10	<b>9</b>

Anfrage	$f_1, \#$	$f_2, \#$	$f_3, \#$	$f_4, \#$	Nr Pagefaults
1	<b>1,1</b>				<b>1</b>
3	1,1	<b>3,1</b>			<b>2</b>
5	1,1	3,1	<b>5,1</b>		<b>3</b>
4	1,1	3,1	5,1	<b>4,1</b>	<b>4</b>
2	<b>2,1</b>	3,1	5,1	4,1	<b>5</b>
4	2,1	3,1	5,1	<b>4,2</b>	5
3	2,1	<b>3,2</b>	5,1	4,2	5
2	<b>2,2</b>	3,2	5,1	4,2	5
1	2,2	3,2	<b>1,1</b>	4,2	<b>6</b>
0	2,2	3,2	<b>0,1</b>	4,2	<b>7</b>
5	2,2	3,2	<b>5,1</b>	4,2	<b>8</b>
3	2,2	<b>3,3</b>	5,1	4,2	8

# FIFO (4 Kacheln)

Anfrage	$f_1$	$f_2$	$f_3$	$f_4$	$f_5$	Nr Pagefaults
1	<b>1</b>					<b>1</b>
3	1	<b>3</b>				<b>2</b>
5	1	3	<b>5</b>			<b>3</b>
4	1	3	5	<b>4</b>		<b>4</b>
2	1	3	5	4	<b>2</b>	<b>5</b>
4	1	3	5	<b>4</b>	2	5
3	1	<b>3</b>	5	4	2	5
2	1	3	5	4	<b>2</b>	5
1	<b>1</b>	3	5	4	2	5
0	<b>0</b>	3	5	4	2	<b>6</b>
5	0	3	<b>5</b>	4	2	6
3	0	<b>3</b>	5	4	2	6

# LRU (4 Kacheln)

Anfrage	$f_1, t$	$f_2, t$	$f_3, t$	$f_4, t$	$f_5, t$	Nr Pagefaults
1	<b>1,1</b>					<b>1</b>
3	1,1	<b>3,2</b>				<b>2</b>
5	1,1	3,2	<b>5,3</b>			<b>3</b>
4	1,1	3,2	5,3	<b>4,4</b>		<b>4</b>
2	1,1	3,2	5,3	4,4	<b>2,5</b>	<b>5</b>
4	1,1	3,2	5,3	<b>4,6</b>	2,5	5
3	1,1	<b>3,7</b>	5,3	4,6	2,5	5
2	1,1	3,7	5,3	4,6	<b>2,8</b>	5
1	<b>1,9</b>	3,7	5,3	4,6	2,8	5
0	1,9	3,7	<b>0,10</b>	4,6	2,8	<b>6</b>
5	1,9	3,7	0,10	<b>5,11</b>	2,8	<b>7</b>
3	1,9	<b>3,12</b>	0,10	5,11	2,8	7

Anfrage	$f_1, \#$	$f_2, \#$	$f_3, \#$	$f_4, \#$	$f_5, \#$	Nr Pagefaults
1	<b>1,1</b>					<b>1</b>
3	1,1	<b>3,1</b>				<b>2</b>
5	1,1	3,1	<b>5,1</b>			<b>3</b>
4	1,1	3,1	5,1	<b>4,1</b>		<b>4</b>
2	1,1	3,1	5,1	4,1	<b>2,1</b>	<b>5</b>
4	1,1	3,1	5,1	<b>4,2</b>	2,1	5
3	1,1	<b>3,2</b>	5,1	4,2	2,1	5
2	1,1	3,2	5,1	4,2	<b>2,2</b>	5
1	<b>1,2</b>	3,2	5,1	4,2	2,2	5
0	1,2	3,2	<b>0,1</b>	4,2	2,2	<b>6</b>
5	1,2	3,2	<b>5,1</b>	4,2	2,2	<b>7</b>
3	1,2	<b>3,3</b>	5,1	4,2	2,2	7

# Aging

## Rechtsshift + Addition

- Aging ist eine Erweiterung der (NFU) Zählerstandberechnung
- Die Zählerstände werden nach einer Zeit  $t$  immer geupdated:
  - 1) Bitshift nach rechts
  - 2) Addieren des R-bits auf das vorderste (reingeshiftete 0) bit

Beispiel:

Seite A: Zugriffszähler  $Z_A = 13_{10} = 1101_2$ , R-bit  $R_A = 0$

Seite B: Zugriffszähler  $Z_B = 7_{10} = 0111_2$ , R-bit  $R_B = 1$

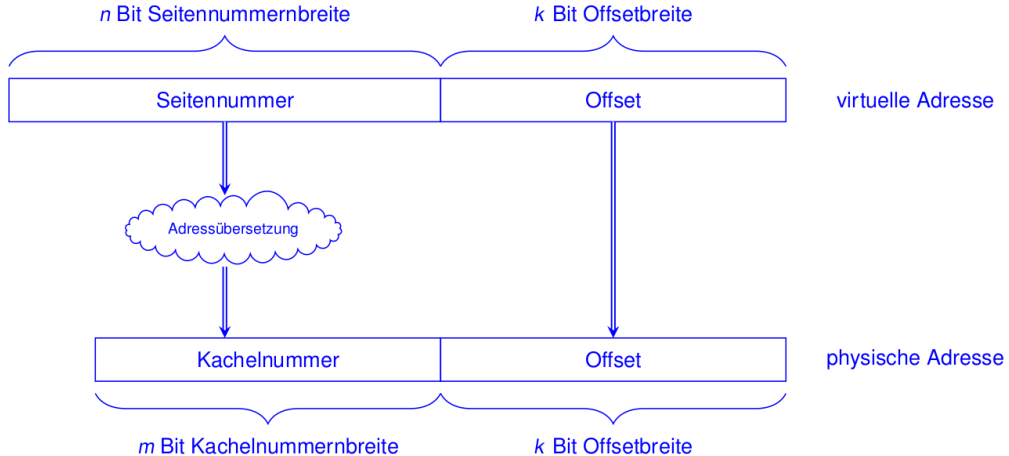
1) Rechtsshift  $\rightarrow Z_A = 0110$ ,  $Z_B = 0011$ .

2) Addition R  $\rightarrow Z_A = 0110$ ,  $Z_B = 1011$ .

$\hookrightarrow Z_A = 6$ ,  $Z_B = 11$ .

# Aufgabe 3

## Virtuelle Adressierung





## Aufgabe 3

### Virtuelle Adressierung

- 12-bit virtuelle Adresse
- 256 Byte physischer Speicher  $\rightarrow$   $\text{addr. bytes} = 256 = 2^8 \rightarrow 8\text{-bit p. A.}$
- 32 Byte Seitengröße ( $\rightarrow$  32 Byte Kachelgröße!)

$\hookrightarrow 32 \text{ addr. Bytes} \Rightarrow 2^5 \rightarrow 5\text{-bit}$

- a) Wie viele Bits der virtuellen Adresse entfallen auf die Seitennummer und Kachelnummer, wie viele auf den Offset? Wie viele Seiten können adressiert werden?

$$\text{Addr. L.} - \text{Offsetbits} = 8 - 5 = 3 \rightarrow 3 \text{ bits Kachel Nr.}$$

$$12 - 5 = 7 \Rightarrow 7\text{-bit für Seiten Nr.}$$

## Aufgabe 3

### Virtuelle Adressierung

- 12-bit virtuelle Adresse
  - 256 Byte physischer Speicher
  - 32 Byte Seitengröße (→ 32 Byte Kachelgröße!)
- a) Wie viele Bits der virtuellen Adresse entfallen auf die Seitennummer und Kachelnummer, wie viele auf den Offset? Wie viele Seiten können adressiert werden?

Bits Offset: Exponent der Seitengröße ( $\log_2(\text{Seitengröße})$ ) → 32 Byte =  $2^5$  Bytes → 5 Offsetbits

## Aufgabe 3

### Virtuelle Adressierung

- 12-bit virtuelle Adresse
- 256 Byte physischer Speicher
- 32 Byte Seitengröße ( $\rightarrow$  32 Byte Kachelgröße!)

a) Wie viele Bits entfallen auf die Seitennummer und Kachelnummer, wie viele auf den Offset? Wie viele Seiten können adressiert werden?

Bits Offset: Exponent der Seitengröße ( $\log_2(\text{Seitengröße})$ )  $\rightarrow$  32 Byte =  $2^5$  Bytes  $\rightarrow$  5 Offsetbits

Bits Seitennummer: Adresslänge - Offsetbits  $\rightarrow$  12-bit v.A.,  $2^5$  Bytes Seitengröße

Adressierbare Seiten:  $2^{\text{bits}} \rightarrow 2^{\text{v.A.}} = 2^7 = 128$

12-5=7

# Aufgabe 3b

## Virtuelle Adressierung

Seitengröße	Offsetbreite	#Einträge bei _ Bit virtueller Adressbreite:		
		16 Bit	32 Bit	64 Bit
4 KiB = 4096 Byte = $2^{12}$ Byte	12-bit	$2^4$ <small><math>16 - 12 = 4</math></small>	$2^{20}$	$2^{52}$
8 KiB = 8192 Byte = $2^{13}$ Byte	13-bit	$2^3$	$2^{19}$	$2^{51}$
16 KiB = 16384 Byte = $2^{14}$ Byte	14-bit	$2^2$	$2^{18}$	$2^{50}$

## Aufgabe 3b

### Virtuelle Adressierung

p.A  $\Rightarrow$  24-bit Addr.L.

v.A  $\Rightarrow$  32-bit Adresslänge

( $L_i L_s$ )

Seitengröße	Offsetbreite physisch	Offsetbreite virtuell	Seitennummer	Kachelnummer
1 KiB = 1024 Byte = $2^{10}$ Byte	10 bit	10 bit	22-bit	10-bit
2 KiB = 2048 Byte = $2^{11}$ Byte	11	11	21	13
4 KiB = 4096 Byte = $2^{12}$ Byte	12	12	20	12
8 KiB = 8192 Byte = $2^{13}$ Byte	13	13	19	11

- P-bit (present): Seite ist im Hauptspeicher (1) oder nicht (0)
- U/S-bit (user/supervisor): Gibt an ob user(prozesse) Zugriff auf die Seite haben (1) oder nur der BS-Kern (0)
- XD-bit (eXecute Disable): Bytes in Page können nur gelesen/geschrieben werden (1) oder auch ausgeführt werden (0)
- **R-bit (referenced)**: Auf die Seite wurde lesend/schreibend zugegriffen (1)
- **M-bit (modified)**: Der Inhalt wurde verändert und muss auf den Hintergrundspeicher zurückgeschrieben werden (1), oder ist nicht verändert und kann verworfen werden (0)

## Second-Chance (FIFO)

**Second-Chance** → R-bit == Extra-life

```
wähle Seite mittels Algorithmus;  
if (Seite zum Ersetzten ausgewählt) {  
    if (R-bit == 1) {  
        lösche R-bit (R = 0);  
        wähle nächste Seite;  
    }  
    else {  
        ersetze Seite;  
        return;  
    }  
}
```

# Second-Chance (FIFO)

## Mögliche Ausprägung

Anfrage	$f_1$ $R_1$	$f_2$ $R_2$	$f_3$ $R_3$	$f_4$ $R_4$
1	1 <sup>1</sup>			
3		3 <sup>1</sup>		
5			5 <sup>1</sup>	
4				4 <sup>1</sup>
2	2 <sup>1</sup>	3 <sup>0</sup>	5 <sup>0</sup>	4 <sup>0</sup>
4				4 <sup>1</sup>
3		3 <sup>1</sup>		
2	2 <sup>1</sup>	3 <sup>0</sup>		
1		0 <sup>1</sup>	1 <sup>1</sup>	
0				

Anfrage	$f_1$	$f_2$	$f_3$	$f_4$
1	<b>1</b>			
3	1	<b>3</b>		
5	1	3	<b>5</b>	
4	1	3	5	<b>4</b>
2	<b>2</b>	3	5	4
4	2	3	5	<b>4</b>
3	2	<b>3</b>	5	4
2	<b>2</b>	3	5	4
1	2	<b>1</b>	5	4
0	2	1	<b>0</b>	4



## Second-Chance Clock

Clock-'Element' enthält: R-bit, M-bit, Seitennummer, Kachelnummer.

Besonderheiten bei der Clock:

- 'Zirkuläre' Anordnung
- Zeiger, der hinter das zuletzt ersetzte Element zeigt.
- Bei Seitenfehler bewegt sich der Zeiger (=Suche nach einer ersetzbaren Seite)

Lese- bzw. Schreibzugriffe ohne Seitenfehler modifizieren lediglich das R-bit bzw. R- und M-bit (der Zeiger bleibt unberührt)-

Schreiben: S2

Seite: 2	
R = 1	M = 1
Kachel: 2	

Kein DF, daher wird nicht benutzt!

nur Bits updaten!

# Second-Chance Clock

schreiben: 5

Seite: 9
R = 0 M = 0
Kachel: 7

Seite: 12
R = 0 M = 0
Kachel: 5

Seite: 5
R = 1 M = 1
Kachel: 4

lesen: 59

Seite: 9
R = 1 M = 0
Kachel: 7

Seite: 12
R = 0 M = 0
Kachel: 5

Seite: 5
R = 1 M = 1
Kachel: 4

Seite: 3
R = 0 M = 0
Kachel: 3



Seite: 13
R = 1 M = 0
Kachel: 6

Seite: 3
R = 0 M = 0
Kachel: 3



Seite: 13
R = 1 M = 0
Kachel: 6

Seite: 1
R = 0 M = 1
Kachel: 1

Seite: 2
R = 1 M = 1
Kachel: 2

Seite: 0
R = 1 M = 0
Kachel: 0

Seite: 1
R = 0 M = 1
Kachel: 1

Seite: 2
R = 1 M = 1
Kachel: 2

Seite: 0
R = 1 M = 0
Kachel: 0

# Second-Chance Clock

Schreiben: 812

Seite: 9
R = 1 M = 0
Kachel: 7

Seite: 12
R = 1 M = 1
Kachel: 5

Seite: 5
R = 1 M = 1
Kachel: 4

Schreiben: 7

Seite: 9
R = 1 M = 0
Kachel: 7

Seite: 12
R = 1 M = 1
Kachel: 5

Seite: 5
R = 1 M = 1
Kachel: 4

Seite: 3
R = 0 M = 0
Kachel: 3



Seite: 13
R = 1 M = 0
Kachel: 6

Seite: 3
R = 0 M = 0
Kachel: 3



Seite: 13
R = 0 M = 0
Kachel: 6

Seite: 1
R = 0 M = 1
Kachel: 1

Seite: 2
R = 1 M = 1
Kachel: 2

Seite: 0
R = 1 M = 0
Kachel: 0

Seite: 7
R = 1 M = 1
Kachel: 1

Seite: 2
R = 0 M = 1
Kachel: 2

Seite: 0
R = 0 M = 0
Kachel: 0

# Second-Chance Clock

schreiben: 510

Seite: 9
R = 1 M = 0
Kachel: 7

Seite: 12
R = 1 M = 1
Kachel: 5

Seite: 5
R = 1 M = 1
Kachel: 4

lesen: 51

Seite: 9
R = 0 M = 0
Kachel: 7

Seite: 12
R = 0 M = 1
Kachel: 5

Seite: 5
R = 0 M = 1
Kachel: 4

Seite: 10
R = 1 M = 1
Kachel: 3

Seite: 13
R = 0 M = 0
Kachel: 6

Seite: 10
R = 1 M = 1
Kachel: 3

Seite: 1
R = 1 M = 0
Kachel: 6

Seite: 7
R = 1 M = 1
Kachel: 1

Seite: 2
R = 0 M = 1
Kachel: 2

Seite: 0
R = 0 M = 0
Kachel: 0

Seite: 7
R = 1 M = 1
Kachel: 1

Seite: 2
R = 0 M = 1
Kachel: 2

Seite: 0
R = 0 M = 0
Kachel: 0