

S2-4:

→ Multiplikation Methode

Typische Hashfunktionen

▪ Division-Rest-Methode

$$h(k) = k \bmod m$$

Hierbei ist m die Tabellengröße. m sollte möglichst eine Primzahl sein.

$$h = 17, i = 7$$

$$17 \bmod 7$$

$$\boxed{3}$$

▪ Multiplikative Methode

$$h(k) = \lfloor m * (k\phi^{-1} - \lfloor k\phi^{-1} \rfloor) \rfloor$$

- m ist die Tabellengröße

- $\phi^{-1} = (\sqrt{5} - 1)/2 \approx 0.6180339887$ ist der Kehrwert des goldenen Schnitts.

- $\lfloor x \rfloor$ runden auf die nächst kleinere ganze Zahl ab.

- Beispielsweise bilden die Werte $h(1), h(2), \dots, h(10)$ für $m = 10$ eine Permutation der Zahlen $0, 1, \dots, 9$ nämlich:
6, 2, 8, 4, 0, 7, 3, 9, 5, 1.

S2-5:

Beispiel mit Division-Rest-Methode

Beispiel

- Füge die Zahlen 7, 24, 5, 8 in eine Hashtabelle der Größe $m = 7$ ein.
- Benutze die Hashfunktion $h(k) = k \bmod m$.

7	tab[0]
8	tab[1]
1	tab[2]
24	tab[3]
	tab[4]
5	tab[5]
	tab[6]

$$7 \bmod 7 = 0$$

$$24 \bmod 7 = 3$$

$$5 \bmod 7 = 5$$

→ 7 mod 7 = 0, 14 mod 7 = 0, wohin mit der 14?

→ Adress-Kollision

S2-9:

$$\begin{aligned} h("ABER") &= ('A' * 31^3 + 'B' * 31^2 + \\ &\quad 'E' * 31^1 + 'R' * 31^0) \bmod m \\ &= [(\underline{'A'} * \underline{31} + \underline{'B'} * \underline{31} + \underline{'E'} * \underline{31} + \underline{'R'})] \bmod m \end{aligned}$$

adr

S2-11:

Hashfunktion hashCode in Java (2)

Hashfunktion für zusammengesetzte Schlüssel

- wird für einen zusammengesetzten Schlüsseltyp eine Hashfunktion benötigt, dann überschreibt man am besten für den Schlüsseltyp die hashCode-Methode.
- Dabei kann eine ähnliche Technik wie bei einem String-Schlüssel eingesetzt werden.
- Beispiel:

```
public class Datum {  
    private int tag;  
    private int mon;  
    private int jahr;  
    // ...  
  
    @Override public int hashCode() {  
        int adr = 0;  
        adr = 31*adr + tag;  
        adr = 31*adr + mon;  
        adr = 31*adr + jahr;  
        return adr;  
    }  
}
```

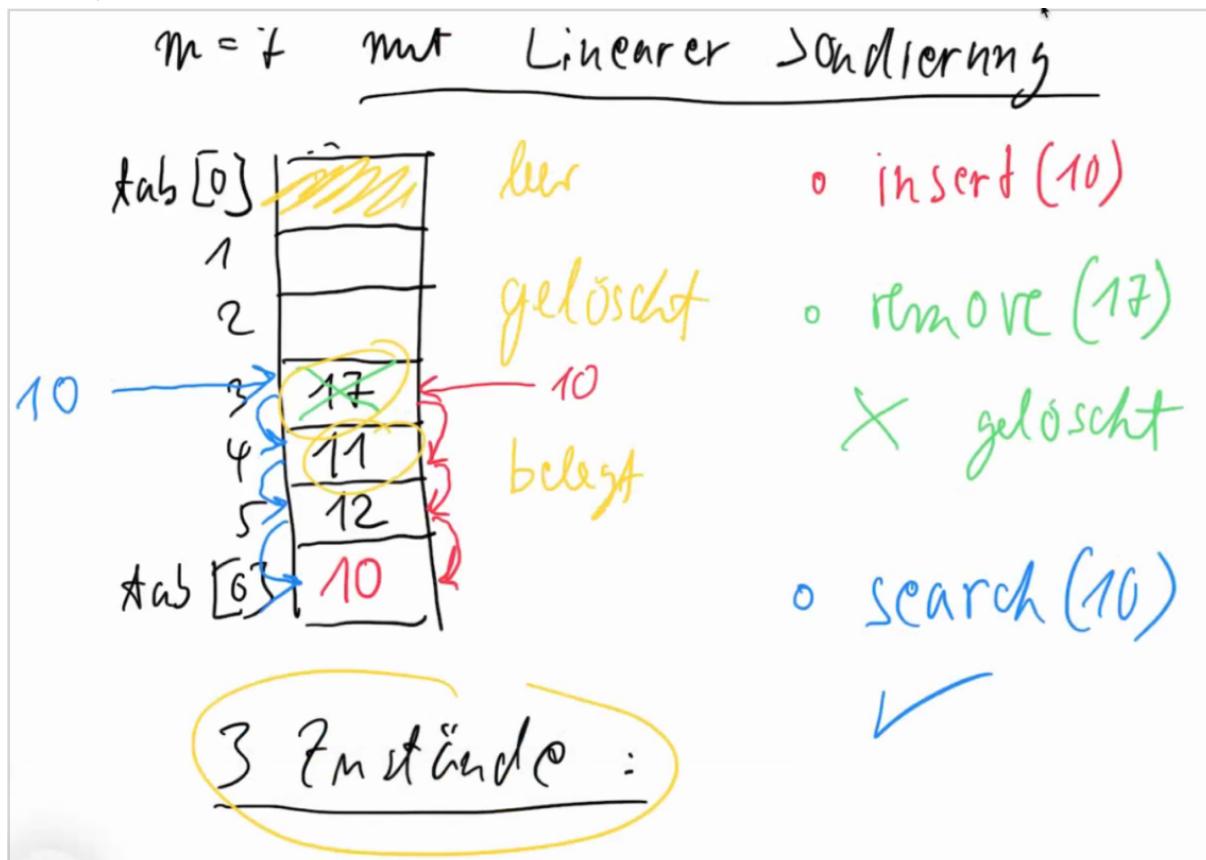
28.2.1980



$$28 * 31^2 + 2 * 31^0 + 1980 * 31^0$$

S2-22:

→ Beispiel



S2-26:

Alternierend quadratisches Sondieren

- Als Tabellengröße m wird eine Primzahl der Form $4i + 3$ gewählt.
- Die alternierend quadratische Sondierungsfolge wird definiert durch:
 $s(j,k) = \lceil j/2 \rceil^2 (-1)^j$

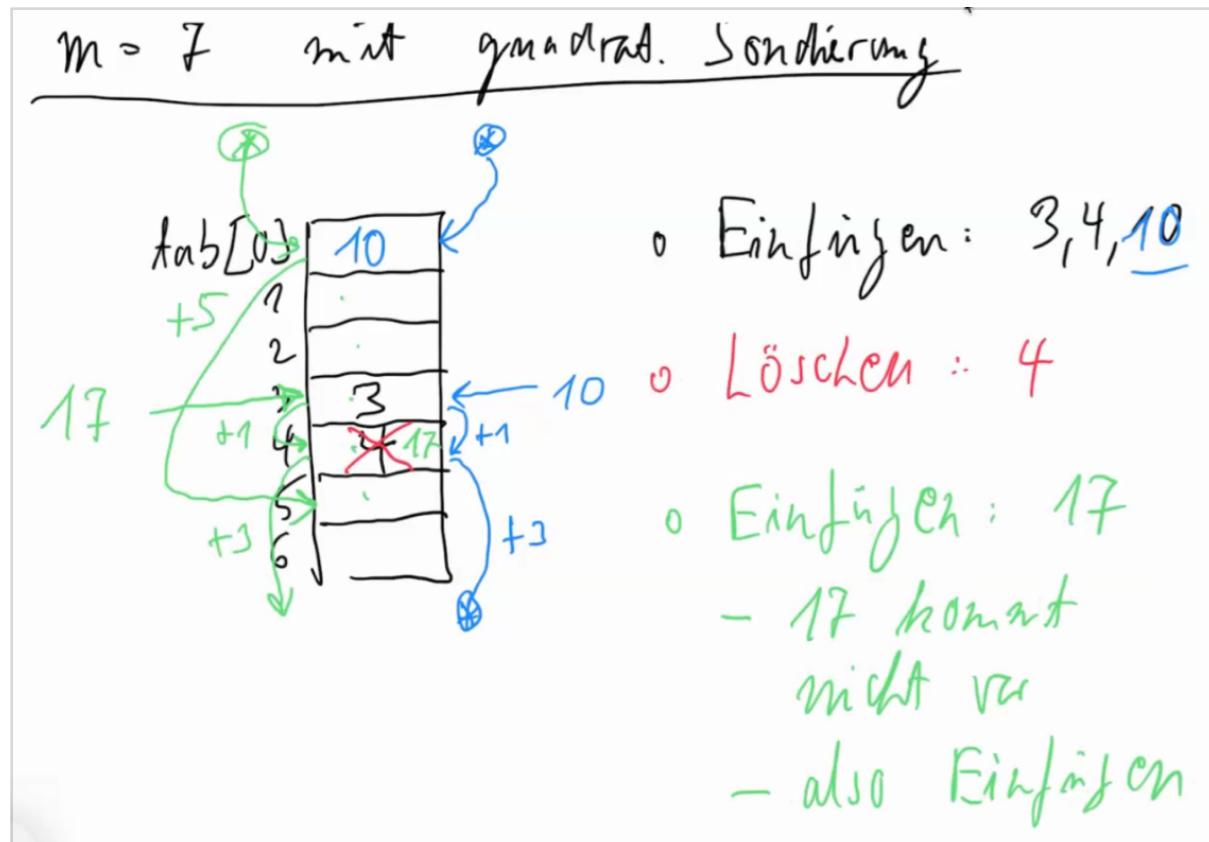
z.B. $m = 7$
 $= 4 \cdot 1 + 3$

- Die Sondierungsfolge lautet konkret:

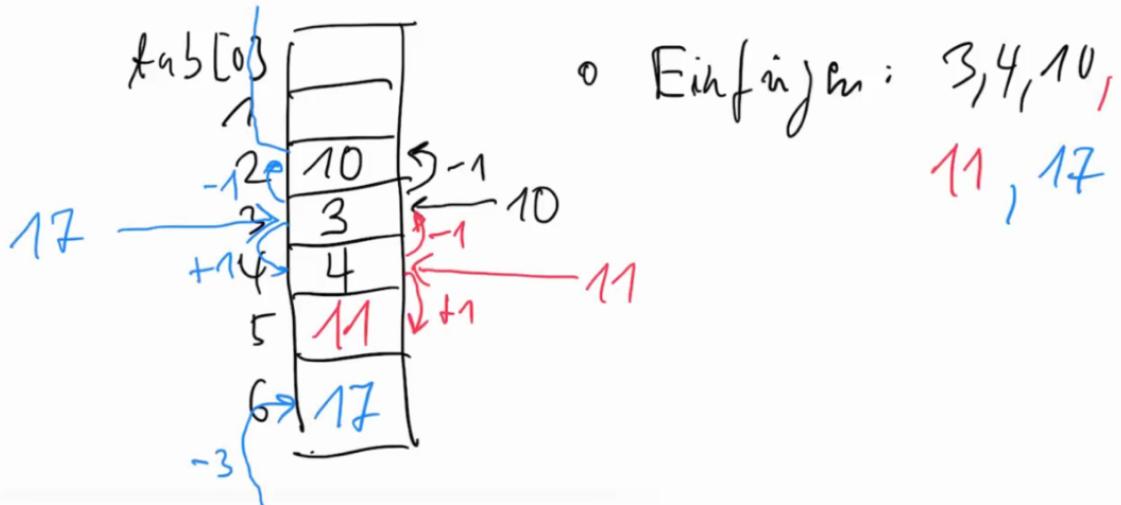
$$\begin{aligned} h(k) \\ h(k) - 1 \bmod m \\ h(k) + 1 \bmod m \\ h(k) - 4 \bmod m \\ h(k) + 4 \bmod m \\ \dots \end{aligned}$$

- Aufpassen: mod ist mathematisch definiert und unterscheidet sich für negative Zahlen vom Modulo-Operator % in Java.
- Mit alternierend quadratischem Sondieren werden alle Einträge erreicht.

Beispiele:



$m = 7$ mit alter. Quadrat. Soll.



S2-29

Analyse im schlechtesten Fall

- Alle n Einträge haben die gleiche Hashadresse.
Daher muss jede Operation eine Liste mit bis zu n Einträgen ablaufen.
- Dies ist in der Praxis so unwahrscheinlich, dass hier die Analyse im mittleren Fall wesentlich wichtiger ist.

un sinnig

Analyse im mittleren Fall

- Wichtige Maßzahlen:
 - C : Anzahl der durchschnittlich betrachteten Einträge bei erfolgreicher Suche.
 - C' : Anzahl der durchschnittlich betrachteten Einträge bei nicht erfolgreicher Suche.
- Es zeigt sich, dass C und C' nur abhängig sind vom Belegungsfaktor $\alpha = n/m$, wobei n = Anzahl der Einträge und m = Tabellengröße.
- Beachte, dass der Belegungsfaktor α bei Hashverfahren mit Verkettung größer 1 sein kann, jedoch bei offenen Hashverfahren kleiner gleich 1 sein muss.

$$\begin{aligned} m &= 100 \\ \left(\frac{1}{100}\right) \left(\frac{1}{100}\right) \cdots \left(\frac{1}{100}\right) \\ m-nal &= 10 - 2n \end{aligned}$$

Analyse im mittleren Fall

- Wichtige Maßzahlen:
 - C : Anzahl der durchschnittlich betrachteten Einträge bei erfolgreicher Suche.
 - C' : Anzahl der durchschnittlich betrachteten Einträge bei nicht erfolgreicher Suche.
- Es zeigt sich, dass C und C' nur abhängig sind vom Belegungsfaktor $\alpha = n/m$, wobei n = Anzahl der Einträge und m = Tabellengröße
- Beachte, dass der Belegungsfaktor α bei Hashverfahren mit Verkettung größer 1 sein kann, jedoch bei offenen Hashverfahren kleiner gleich 1 sein muss.

$$d = \frac{n=400}{m=100}$$

$$d = \frac{n=30}{m=100}$$

- oberes a bei linear verkettet (da mehrere einträge pro stelle geben kann, d.h kann sein $n > m$)
- unteres a bei nicht linear verkettet

S2-30

Analyse der Hashverfahren (2)

C und C' für verschiedene Hashverfahren

Erfolgr. nicht erf. folgr.

Smale

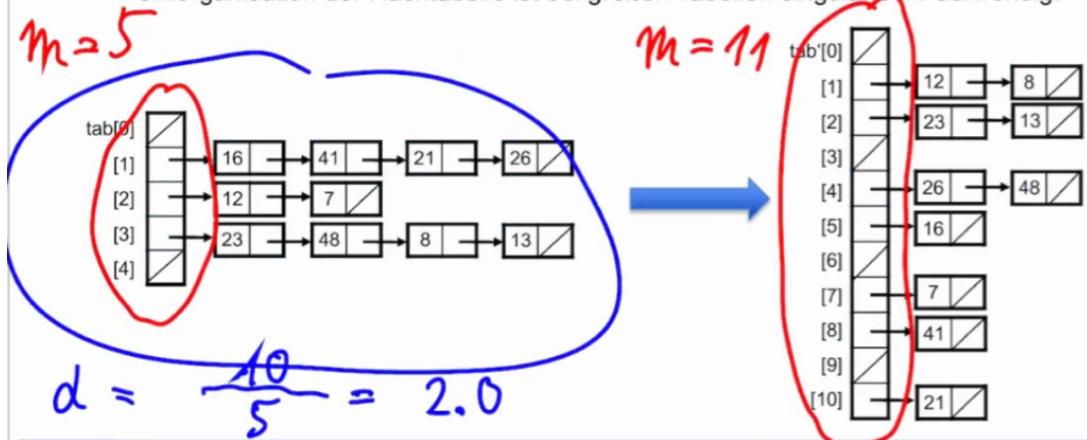
Verfahren	C	C'
Hashverfahren mit Verkettung	$1 + \frac{\alpha}{2}$	α
Offenes Hashverfahren mit linearem Sondieren	$\frac{1}{2} \left(1 + \frac{1}{1-\alpha} \right)$	$\frac{1}{2} \left(1 + \frac{1}{(1-\alpha)^2} \right)$
Offenes Hashverfahren mit quadratischem Sondieren	$1 + \ln \left(\frac{1}{1-\alpha} \right) - \frac{\alpha}{2}$	$\frac{1}{1-\alpha} - \alpha + \ln \left(\frac{1}{1-\alpha} \right)$
double hashing mit unabhängigen h u. h'	$\frac{1}{\alpha} \ln \left(\frac{1}{1-\alpha} \right)$	$\frac{1}{1-\alpha}$

- Die Angaben für C und C' setzen eine ideale Hashfunktion voraus. D.h. die Schlüssel sind gleichmäßig über die Tabelle verstreut.
- Die umfangreichen Herleitungen können in [Ottmann u. Widmayer 2002] nachgelesen werden.

S2-33

Vergrößern der Tabelle mit sofortigem Umkopieren

- Falls bestimmter Füllungsgrad überschritten wird:
 - Lege neue Tabelle tab' mit einer in etwa doppelten Größe m' an (m' sollte Primzahl sein).
 - Füge alle Einträge aus alter Tabelle tab in die neue Tabelle tab' ein.
 - Das alte Feld tab wird durch tab' ersetzt.
- Problem:
Umorganisation der Hashtabelle ist bei großen Tabellen singulär sehr aufwendig.



Prof. Dr. O. Bittel, HTWG Konstanz

Algorithmen und Datenstrukturen – Hashverfahren

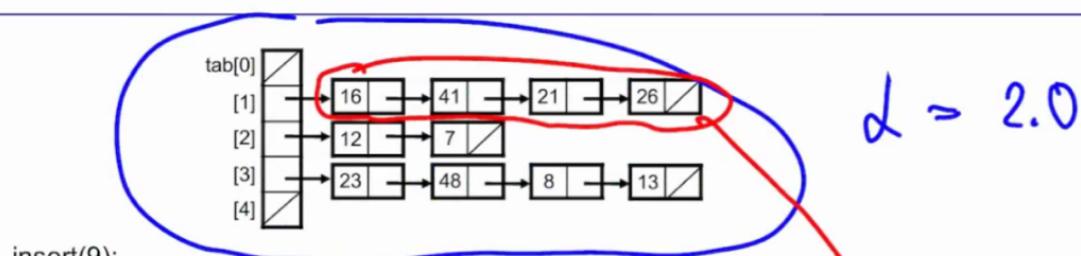
WS 20/21

2-33

→ Folie hat ein paar Fehler. 8 und 23 sind an der falschen Stelle in der neuen Tabelle

S2-35

Beispiel zu Vergrößern der Tabelle mit verzögertem Umkopieren (1)



insert(9):

- Lege neue Tabelle tab' der Größe 11 an, da Füllungsgrad von tab überschritten wird.
- Übertrage kleinsten Tabelleneintrag tab[min] mit min = 1 nach tab'.
- Füge nun 9 in alte Tabelle ein (da $h(9) = 9 \bmod 5 = 4 > \min$).

