

Übung 9 – Algorithmen II

Yaroslav Akhremtsev, Demian Hespe – yaroslav.akhremtsev@kit.edu, hespe@kit.edu

Mit Folien von Michael Axtmann

http://algo2.iti.kit.edu/AlgorithmenII_WS17.php

Institut für Theoretische Informatik - Algorithmik II

```
    result = current_weight;
    return true;
}

for( EdgeID eid = graph.edgeBegin( current ); eid != graph.edgeEnd( current ); ++eid ){
    const Edge & edge = graph.getEdge( eid );
    COUNTING( statistic_data.inc( DijkstraStatisticData::TOUCHED_EDGES ); )
    if( edge.forward ){
        COUNTING( statistic_data.inc( DijkstraStatisticData::RELAXED_EDGES ); )
        weight new_weight = edge.weight + current_weight;
        GUARANTEE( new_weight >= current_weight, std::runtime_error, "Weight overflow detected." );
        if( !priority_queue.isReached( edge.target ) ){
            COUNTING( statistic_data.inc( DijkstraStatisticData::SUCCESSFULLY_RELAXED_EDGES ); )
            COUNTING( statistic_data.inc( DijkstraStatisticData::REACHED_NODES ); )
            priority_queue.push( edge.target, new_weight );
        } else {
            if( priority_queue.getCurrentKey( edge.target ) > new_weight ){
                COUNTING( statistic_data.inc( DijkstraStatisticData::SUCCESSFULLY_RELAXED_NODES ); )
                priority_queue.decreaseKey( edge.target, new_weight );
            }
        }
    }
}
```

Spezielle *Priority Queues*

monoton, ganzzahlig

Bedingungen

- positive, **ganzzahlige** Schlüssel
- neue/geänderte Schlüssel $\geq$ minimaler Schlüssel *min*
- **maximales** Schlüsselinkrement *C*
(bezogen auf minimalen Schlüssel *min*)

Spezielle *Priority Queues*

monoton, ganzzahlig

Warum das Ganze?

- spezialisierte Datenstruktur → schneller
- Dijkstras Algorithmus

Idee

- speichere Schlüssel in *Buckets* statt *Bäumen*

Spezielle *Priority Queues*

Dijkstras Algorithmus

Laufzeit Dijkstras Algorithmus

■ $T_{Dijkstra} = O(m \cdot T_{decreaseKey} + n \cdot (T_{deleteMin} + T_{insert}))$

amortisierte Laufzeiten

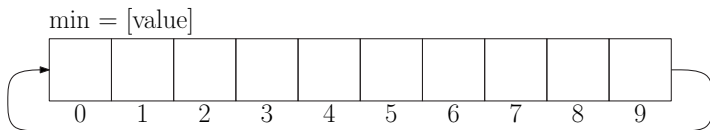
$O(\cdot)$	<i>Bin. Heap</i>	<i>Fib. Heap</i>	<i>Bkt. Queue</i>	<i>Radix Heap</i>
$T_{decreaseKey}$	$\log n$	1	1	1
T_{insert}	$\log n$	1	1	$\log C$
$T_{deleteMin}$	$\log n$	$\log n$	C	$\log C$
$T_{Dijkstra}$	$(m + n) \log n$	$m + n \log n$	$m + nC$	$m + n \log C$

Aufbau:

- zirkuläre Liste aus Buckets $B[\cdot]$
- Variable mit minimalem Schlüssel min
(der in die Datenstruktur aufgenommen werden kann)
- min : letzter deleteMin-Schlüssel, initialisiert mit 0

gegeben:

- max. Schlüsselinkrement $C \longrightarrow \# \text{ Buckets } |B| = C + 1$



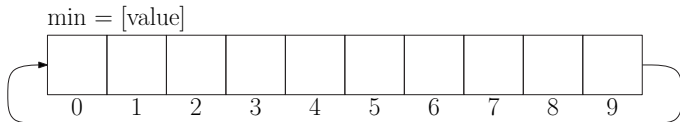
$$C = 9 \longrightarrow |B| = C + 1 = 10$$

Ablauf:

insert:	Einfügen in Bucket $B[key \bmod (C + 1)]$	$O(1)$
deleteMin:	Entfernen aus Bucket $B[\min \bmod (C + 1)]$ (bzw. aus erstem nicht-leeren Folgebucket)	$O(C)$
decreaseKey:	Verschieben von altem in neuen Bucket	$O(1)$

Beispiel:

- insert(a,5)
- insert(b,7)
- deleteMin()
- insert(c,13)
- insert(d,13)
- insert(e,11)
- decreaseKey(c,11)

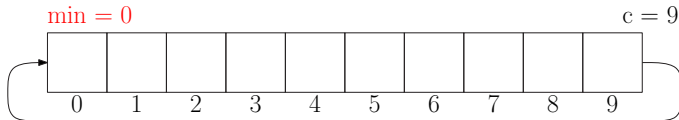


Ablauf:

insert:	Einfügen in Bucket $B[key \bmod (C + 1)]$	$O(1)$
deleteMin:	Entfernen aus Bucket $B[min \bmod (C + 1)]$ (bzw. aus erstem nicht-leeren Folgebucket)	$O(C)$
decreaseKey:	Verschieben von altem in neuen Bucket	$O(1)$

Beispiel:

- insert(a,5)
- insert(b,7)
- deleteMin()
- insert(c,13)
- insert(d,13)
- insert(e,11)
- decreaseKey(c,11)

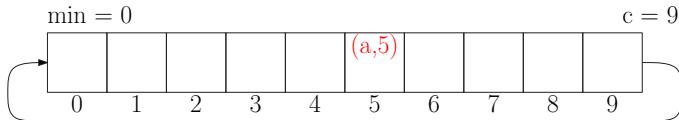


Ablauf:

insert:	Einfügen in Bucket $B[key \bmod (C + 1)]$	$O(1)$
deleteMin:	Entfernen aus Bucket $B[\min \bmod (C + 1)]$ (bzw. aus erstem nicht-leeren Folgebucket)	$O(C)$
decreaseKey:	Verschieben von altem in neuen Bucket	$O(1)$

Beispiel:

- `insert(a,5)`
- `insert(b,7)`
- `deleteMin()`
- `insert(c,13)`
- `insert(d,13)`
- `insert(e,11)`
- `decreaseKey(c,11)`

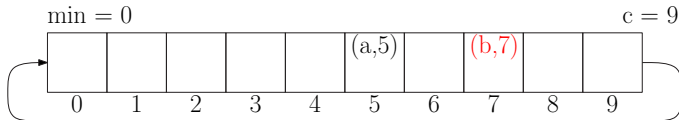


Ablauf:

insert:	Einfügen in Bucket $B[key \bmod (C + 1)]$	$O(1)$
deleteMin:	Entfernen aus Bucket $B[\min \bmod (C + 1)]$ (bzw. aus erstem nicht-leeren Folgebucket)	$O(C)$
decreaseKey:	Verschieben von altem in neuen Bucket	$O(1)$

Beispiel:

- insert(a,5)
- insert(b,7)
- deleteMin()
- insert(c,13)
- insert(d,13)
- insert(e,11)
- decreaseKey(c,11)

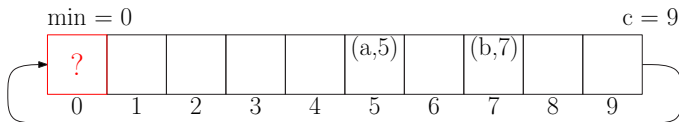


Ablauf:

insert:	Einfügen in Bucket $B[key \bmod (C + 1)]$	$O(1)$
deleteMin:	Entfernen aus Bucket $B[\min \bmod (C + 1)]$ (bzw. aus erstem nicht-leeren Folgebucket)	$O(C)$
decreaseKey:	Verschieben von altem in neuen Bucket	$O(1)$

Beispiel:

- insert(a,5)
- insert(b,7)
- deleteMin()
- insert(c,13)
- insert(d,13)
- insert(e,11)
- decreaseKey(c,11)

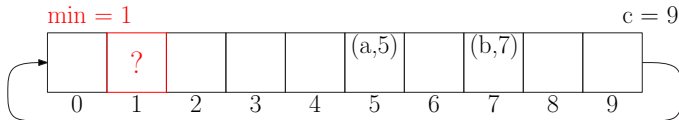


Ablauf:

insert:	Einfügen in Bucket $B[key \bmod (C + 1)]$	$O(1)$
deleteMin:	Entfernen aus Bucket $B[min \bmod (C + 1)]$ (bzw. aus erstem nicht-leeren Folgebucket)	$O(C)$
decreaseKey:	Verschieben von altem in neuen Bucket	$O(1)$

Beispiel:

- insert(a,5)
- insert(b,7)
- deleteMin()
- insert(c,13)
- insert(d,13)
- insert(e,11)
- decreaseKey(c,11)

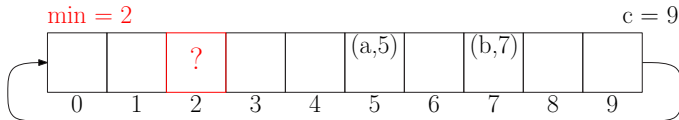


Ablauf:

insert:	Einfügen in Bucket $B[key \bmod (C + 1)]$	$O(1)$
deleteMin:	Entfernen aus Bucket $B[\min \bmod (C + 1)]$ (bzw. aus erstem nicht-leeren Folgebucket)	$O(C)$
decreaseKey:	Verschieben von altem in neuen Bucket	$O(1)$

Beispiel:

- insert(a,5)
- insert(b,7)
- deleteMin()
- insert(c,13)
- insert(d,13)
- insert(e,11)
- decreaseKey(c,11)

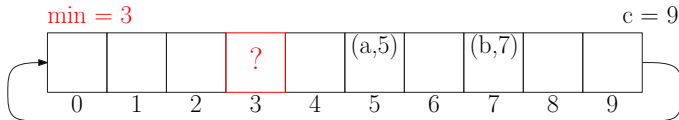


Ablauf:

insert:	Einfügen in Bucket $B[key \bmod (C + 1)]$	$O(1)$
deleteMin:	Entfernen aus Bucket $B[\min \bmod (C + 1)]$ (bzw. aus erstem nicht-leeren Folgebucket)	$O(C)$
decreaseKey:	Verschieben von altem in neuen Bucket	$O(1)$

Beispiel:

- insert(a,5)
- insert(b,7)
- deleteMin()
- insert(c,13)
- insert(d,13)
- insert(e,11)
- decreaseKey(c,11)

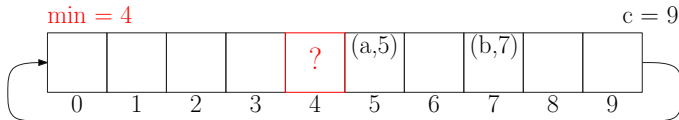


Ablauf:

insert:	Einfügen in Bucket $B[key \bmod (C + 1)]$	$O(1)$
deleteMin:	Entfernen aus Bucket $B[\min \bmod (C + 1)]$ (bzw. aus erstem nicht-leeren Folgebucket)	$O(C)$
decreaseKey:	Verschieben von altem in neuen Bucket	$O(1)$

Beispiel:

- insert(a,5)
- insert(b,7)
- deleteMin()
- insert(c,13)
- insert(d,13)
- insert(e,11)
- decreaseKey(c,11)

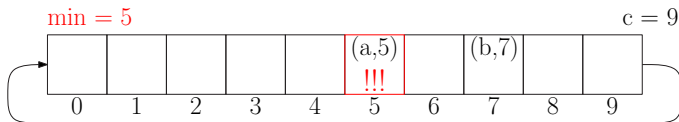


Ablauf:

insert:	Einfügen in Bucket $B[key \bmod (C + 1)]$	$O(1)$
deleteMin:	Entfernen aus Bucket $B[min \bmod (C + 1)]$ (bzw. aus erstem nicht-leeren Folgebucket)	$O(C)$
decreaseKey:	Verschieben von altem in neuen Bucket	$O(1)$

Beispiel:

- insert(a,5)
- insert(b,7)
- deleteMin()
- insert(c,13)
- insert(d,13)
- insert(e,11)
- decreaseKey(c,11)

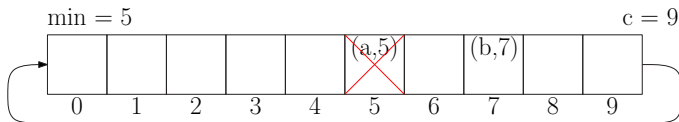


Ablauf:

insert:	Einfügen in Bucket $B[key \bmod (C + 1)]$	$O(1)$
deleteMin:	Entfernen aus Bucket $B[min \bmod (C + 1)]$ (bzw. aus erstem nicht-leeren Folgebucket)	$O(C)$
decreaseKey:	Verschieben von altem in neuen Bucket	$O(1)$

Beispiel:

- insert(a,5)
- insert(b,7)
- deleteMin()
- insert(c,13)
- insert(d,13)
- insert(e,11)
- decreaseKey(c,11)

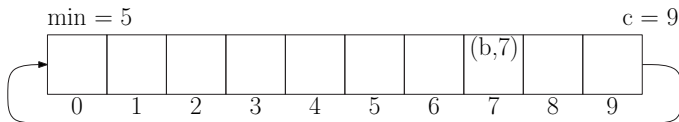


Ablauf:

insert:	Einfügen in Bucket $B[key \bmod (C + 1)]$	$O(1)$
deleteMin:	Entfernen aus Bucket $B[\min \bmod (C + 1)]$ (bzw. aus erstem nicht-leeren Folgebucket)	$O(C)$
decreaseKey:	Verschieben von altem in neuen Bucket	$O(1)$

Beispiel:

- insert(a,5)
- insert(b,7)
- deleteMin()
- insert(c,13)
- insert(d,13)
- insert(e,11)
- decreaseKey(c,11)

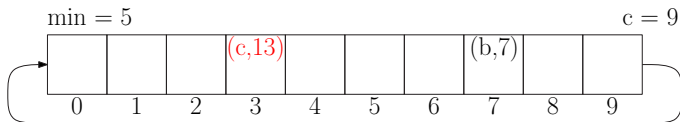


Ablauf:

insert:	Einfügen in Bucket $B[key \bmod (C + 1)]$	$O(1)$
deleteMin:	Entfernen aus Bucket $B[\min \bmod (C + 1)]$ (bzw. aus erstem nicht-leeren Folgebucket)	$O(C)$
decreaseKey:	Verschieben von altem in neuen Bucket	$O(1)$

Beispiel:

- insert(a,5)
- insert(b,7)
- deleteMin()
- insert(c,13)
- insert(d,13)
- insert(e,11)
- decreaseKey(c,11)

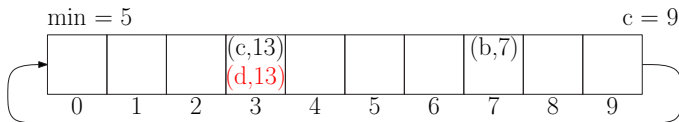


Ablauf:

insert:	Einfügen in Bucket $B[key \bmod (C + 1)]$	$O(1)$
deleteMin:	Entfernen aus Bucket $B[\min \bmod (C + 1)]$ (bzw. aus erstem nicht-leeren Folgebucket)	$O(C)$
decreaseKey:	Verschieben von altem in neuen Bucket	$O(1)$

Beispiel:

- insert(a,5)
- insert(b,7)
- deleteMin()
- insert(c,13)
- insert(d,13)
- insert(e,11)
- decreaseKey(c,11)

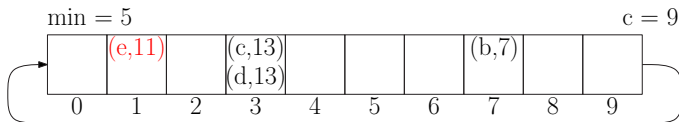


Ablauf:

insert:	Einfügen in Bucket $B[key \bmod (C + 1)]$	$O(1)$
deleteMin:	Entfernen aus Bucket $B[\min \bmod (C + 1)]$ (bzw. aus erstem nicht-leeren Folgebucket)	$O(C)$
decreaseKey:	Verschieben von altem in neuen Bucket	$O(1)$

Beispiel:

- insert(a,5)
- insert(b,7)
- deleteMin()
- insert(c,13)
- insert(d,13)
- insert(e,11)
- decreaseKey(c,11)

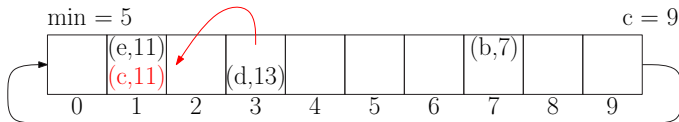


Ablauf:

insert:	Einfügen in Bucket $B[key \bmod (C + 1)]$	$O(1)$
deleteMin:	Entfernen aus Bucket $B[min \bmod (C + 1)]$ (bzw. aus erstem nicht-leeren Folgebucket)	$O(C)$
decreaseKey:	Verschieben von altem in neuen Bucket	$O(1)$

Beispiel:

- insert(a,5)
- insert(b,7)
- deleteMin()
- insert(c,13)
- insert(d,13)
- insert(e,11)
- decreaseKey(c,11)



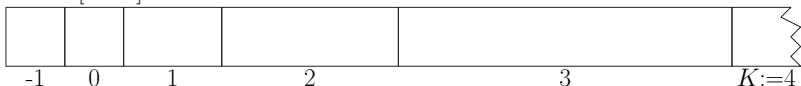
Aufbau:

- Liste aus **logarithmischer Anzahl** Buckets $B[\cdot]$
- Variable mit minimalem Schlüssel min
(der in die Datenstruktur aufgenommen werden kann)

gegeben:

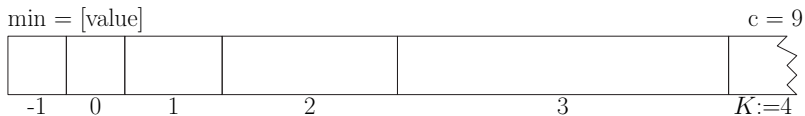
- max. Schlüsselinkrement $C \longrightarrow \# \text{ Buckets } |B| = K + 2$
 $= (\lfloor \log C \rfloor + 1) + 2$

$min = [value]$

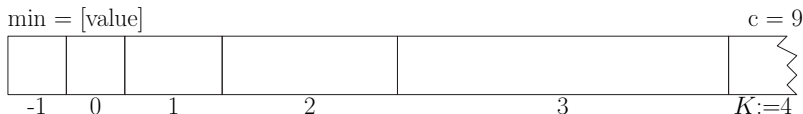


$$C = 9 \longrightarrow K + 2 = (\lfloor \log C \rfloor + 1) + 2 = (3 + 1) + 2 = 6$$

Radix Heaps

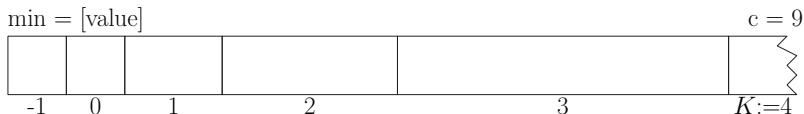


Welche Schlüssel kommen in welches Bucket?



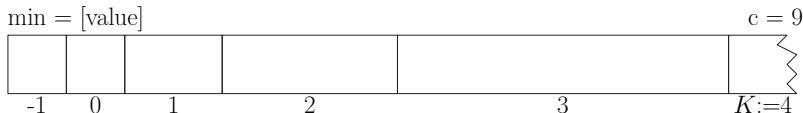
Welche Schlüssel kommen in welches Bucket?

- Bucket i : hält Elemente mit $i = \min(\text{msd}(\text{min}, \text{key}), K)$
- Bucket K ist *overflow* Bucket
 - höchstwertiges unterschiedliche Bit ist i
(bzw. -1 wenn $\text{key} = \text{min}$)
 - $\max(1, 2^i)$ verschiedene Schlüssel
 - leer, wenn Bit i von min gesetzt



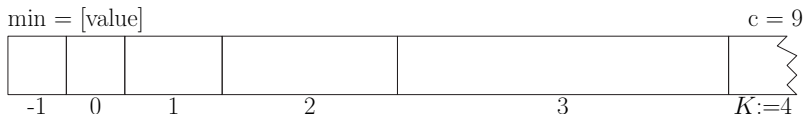
Welche Schlüssel kommen in welches Bucket?

- Bucket i : hält Elemente mit $i = \min(\text{msd}(\text{min}, \text{key}), K)$
- Bucket K ist *overflow* Bucket
 - höchstwertiges unterschiedliche Bit ist i
(bzw. -1 wenn $\text{key} = \text{min}$)
 - $\max(1, 2^i)$ verschiedene Schlüssel
 - leer, wenn Bit i von min gesetzt



Welche Schlüssel kommen in welches Bucket?

- Bucket i : hält Elemente mit $i = \min(\text{msd}(\text{min}, \text{key}), K)$
- Bucket K ist *overflow* Bucket
 - höchstwertiges unterschiedliche Bit ist i
(bzw. -1 wenn $\text{key} = \text{min}$)
 - $\max(1, 2^i)$ verschiedene Schlüssel
 - leer, wenn Bit i von min gesetzt



Welche Schlüssel kommen in welches Bucket?

Bucket i : hält Elemente mit $i = \min(\text{msd}(\text{min}, \text{key}), K)$

→ Bucket K ist *overflow* Bucket

→ höchstwertiges unterschiedliche Bit ist i
(bzw. -1 wenn $\text{key} = \text{min}$)

→ $\max(1, 2^i)$ verschiedene Schlüssel

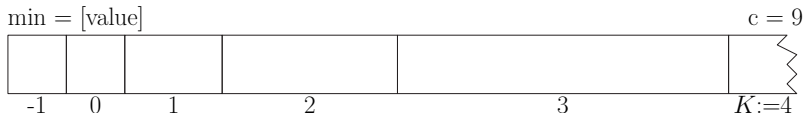
→ leer, wenn Bit i von min gesetzt

Beispiele

$\text{min} := 00001000$ (08)

$\text{msd}(00001010, 01001000) = 6$

$\text{msd}(00001000, 00001010) = 1$



Welche Schlüssel kommen in welches Bucket?

Bucket i : hält Elemente mit $i = \min(\text{msd}(\text{min}, \text{key}), K)$

→ Bucket K ist *overflow* Bucket

→ höchstwertiges unterschiedliche Bit ist i

(bzw. -1 wenn $\text{key} = \text{min}$)

→ $\max(1, 2^i)$ verschiedene Schlüssel

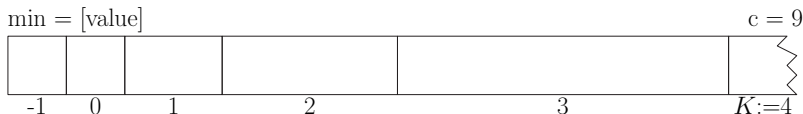
→ leer, wenn Bit i von min gesetzt

Beispiele

$\text{min} := 00001000$ (08)

$\text{msd}(00001000, 01*****) = 6 \rightarrow 2^6 = 64$ Werte

$\text{msd}(00001000, 0000101*) = 1 \rightarrow 2^1 = 2$ Werte



Welche Schlüssel kommen in welches Bucket?

Bucket i : hält Elemente mit $i = \min(\text{msd}(\text{min}, \text{key}), K)$

→ Bucket K ist *overflow* Bucket

→ höchstwertiges unterschiedliche Bit ist i

(bzw. -1 wenn $\text{key} = \text{min}$)

→ $\max(1, 2^i)$ verschiedene Schlüssel

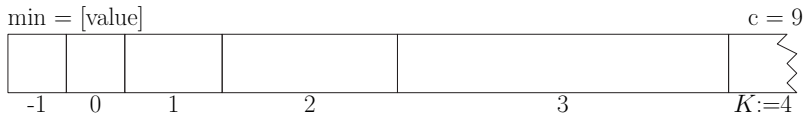
→ **leer, wenn Bit i von min gesetzt**

Beispiele

$\text{min} := 00001000$ (08)

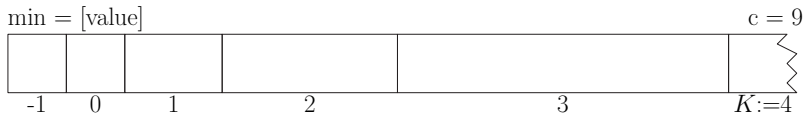
$\text{msd}(0000\textcolor{red}{1}000, 0000\textcolor{red}{0}110) = 3 \rightarrow$ kann nicht passieren, da $6 < 8$

$\text{msd}(0000\textcolor{red}{1}000, 0000\textcolor{red}{0}100) = 3 \rightarrow$ kann nicht passieren, da $4 < 8$



Welche Schlüssel kommen in welches Bucket?

	Bucket $B[\cdot]$					
<i>min</i>	-1	0	1	2	3	4
1000 (08)	8	9	10..11	12..15	-	16..8+C
1010 (10)	10	11	-	12..15	-	16..10+C
1111 (15)	15	-	-	-	-	16..15+C
1000100 (68)	68	69	70..71	-	72..79 ?	80..69+C ?



Welche Schlüssel kommen in welches Bucket?

	Bucket $B[\cdot]$					
<i>min</i>	-1	0	1	2	3	4
1000 (08)	8	9	10..11	12..15	-	16..8+C
1010 (10)	10	11	-	12..15	-	16..10+C
1111 (15)	15	-	-	-	-	16..15+C
1000100 (68)	68	69	70..71	-	72..77 !	- !

Ablauf und Laufzeiten (amortisiert):

insert:	Einfügen in Bucket $B[\min(\text{msd}(\text{min}, \text{key}), K)]$	$O(\log C)$
deleteMin:	min = minimaler Schlüssel (aus Bucket i), Elemente aus Bucket i verschieben, Entfernen aus Bucket $B[-1]$	$O(\log C)$
decreaseKey:	Verschieben von altem in neues Bucket	$O(1)$

Beispiel:

- insert(a,5)
- insert(b,7)
- deleteMin()
- insert(c,13)
- insert(d,13)
- insert(e,11)
- decreaseKey(c,11)



Ablauf und Laufzeiten (amortisiert):

insert:	Einfügen in Bucket $B[\min(\text{msd}(\text{min}, \text{key}), K)]$	$O(\log C)$
deleteMin:	min = minimaler Schlüssel (aus Bucket i), Elemente aus Bucket i verschieben, Entfernen aus Bucket $B[-1]$	$O(\log C)$
decreaseKey:	Verschieben von altem in neues Bucket	$O(1)$

Beispiel:

- insert(a,5)
- insert(b,7)
- deleteMin()
- insert(c,13)
- insert(d,13)
- insert(e,11)
- decreaseKey(c,11)

min = 0000



Ablauf und Laufzeiten (amortisiert):

insert:	Einfügen in Bucket $B[\min(\text{msd}(\text{min}, \text{key}), K)]$	$O(\log C)$
deleteMin:	min = minimaler Schlüssel (aus Bucket i), Elemente aus Bucket i verschieben, Entfernen aus Bucket $B[-1]$	$O(\log C)$
decreaseKey:	Verschieben von altem in neues Bucket	$O(1)$

Beispiel:

■ insert(a,5)

■ insert(b,7)

■ deleteMin()

■ insert(c,13)

■ insert(d,13)

■ insert(e,11)

■ decreaseKey(c,11)

min = 0000



Ablauf und Laufzeiten (amortisiert):

insert:	Einfügen in Bucket $B[\min(\text{msd}(\text{min}, \text{key}), K)]$	$O(\log C)$
deleteMin:	min = minimaler Schlüssel (aus Bucket i), Elemente aus Bucket i verschieben, Entfernen aus Bucket $B[-1]$	$O(\log C)$
decreaseKey:	Verschieben von altem in neues Bucket	$O(1)$

Beispiel:

■ insert(a,5)

■ insert(b,7)

■ deleteMin()

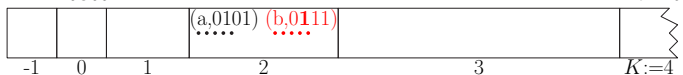
■ insert(c,13)

■ insert(d,13)

■ insert(e,11)

■ decreaseKey(c,11)

min = 0000



Ablauf und Laufzeiten (amortisiert):

insert:	Einfügen in Bucket $B[\min(\text{msd}(\text{min}, \text{key}), K)]$	$O(\log C)$
deleteMin:	min = minimaler Schlüssel (aus Bucket i), Elemente aus Bucket i verschieben, Entfernen aus Bucket $B[-1]$	$O(\log C)$
decreaseKey:	Verschieben von altem in neues Bucket	$O(1)$

Beispiel:

■ insert(a,5)

■ insert(b,7)

■ deleteMin()

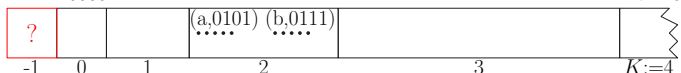
■ insert(c,13)

■ insert(d,13)

■ insert(e,11)

■ decreaseKey(c,11)

min = 0000



Ablauf und Laufzeiten (amortisiert):

insert:	Einfügen in Bucket $B[\min(\text{msd}(\text{min}, \text{key}), K)]$	$O(\log C)$
deleteMin:	min = minimaler Schlüssel (aus Bucket i), Elemente aus Bucket i verschieben, Entfernen aus Bucket $B[-1]$	$O(\log C)$
decreaseKey:	Verschieben von altem in neues Bucket	$O(1)$

Beispiel:

■ insert(a,5)

■ insert(b,7)

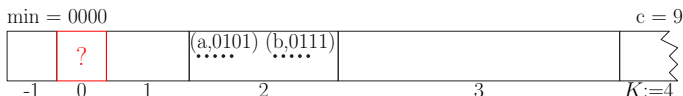
■ deleteMin()

■ insert(c,13)

■ insert(d,13)

■ insert(e,11)

■ decreaseKey(c,11)



Ablauf und Laufzeiten (amortisiert):

insert:	Einfügen in Bucket $B[\min(\text{msd}(\text{min}, \text{key}), K)]$	$O(\log C)$
deleteMin:	min = minimaler Schlüssel (aus Bucket i), Elemente aus Bucket i verschieben, Entfernen aus Bucket $B[-1]$	$O(\log C)$
decreaseKey:	Verschieben von altem in neues Bucket	$O(1)$

Beispiel:

■ insert(a,5)

■ insert(b,7)

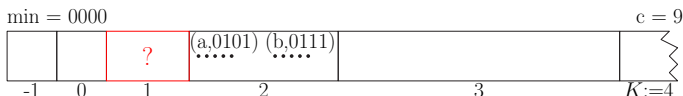
■ deleteMin()

■ insert(c,13)

■ insert(d,13)

■ insert(e,11)

■ decreaseKey(c,11)

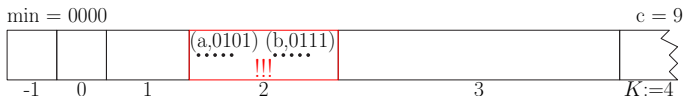


Ablauf und Laufzeiten (amortisiert):

insert:	Einfügen in Bucket $B[\min(\text{msd}(\text{min}, \text{key}), K)]$	$O(\log C)$
deleteMin:	min = minimaler Schlüssel (aus Bucket i), Elemente aus Bucket i verschieben, Entfernen aus Bucket $B[-1]$	$O(\log C)$
decreaseKey:	Verschieben von altem in neues Bucket	$O(1)$

Beispiel:

- insert(a,5)
- insert(b,7)
- deleteMin()
- insert(c,13)
- insert(d,13)
- insert(e,11)
- decreaseKey(c,11)



Ablauf und Laufzeiten (amortisiert):

insert:	Einfügen in Bucket $B[\min(\text{msd}(\text{min}, \text{key}), K)]$	$O(\log C)$
deleteMin:	min = minimaler Schlüssel (aus Bucket i), Elemente aus Bucket i verschieben, Entfernen aus Bucket $B[-1]$	$O(\log C)$
decreaseKey:	Verschieben von altem in neues Bucket	$O(1)$

Beispiel:

■ insert(a,5)

■ insert(b,7)

■ deleteMin()

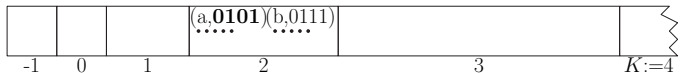
■ insert(c,13)

■ insert(d,13)

■ insert(e,11)

■ decreaseKey(c,11)

min = 0101

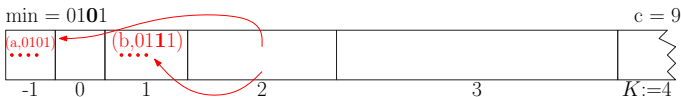


Ablauf und Laufzeiten (amortisiert):

insert:	Einfügen in Bucket $B[\min(\text{msd}(\text{min}, \text{key}), K)]$	$O(\log C)$
deleteMin:	min = minimaler Schlüssel (aus Bucket i), Elemente aus Bucket i verschieben, Entfernen aus Bucket $B[-1]$	$O(\log C)$
decreaseKey:	Verschieben von altem in neues Bucket	$O(1)$

Beispiel:

- insert(a,5)
- insert(b,7)
- deleteMin()
- insert(c,13)
- insert(d,13)
- insert(e,11)
- decreaseKey(c,11)

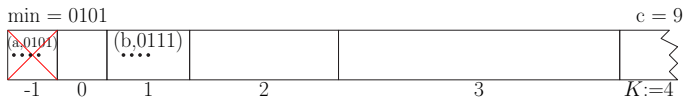


Ablauf und Laufzeiten (amortisiert):

insert:	Einfügen in Bucket $B[\min(\text{msd}(\text{min}, \text{key}), K)]$	$O(\log C)$
deleteMin:	min = minimaler Schlüssel (aus Bucket i), Elemente aus Bucket i verschieben, Entfernen aus Bucket $B[-1]$	$O(\log C)$
decreaseKey:	Verschieben von altem in neues Bucket	$O(1)$

Beispiel:

- insert(a,5)
- insert(b,7)
- deleteMin()
- insert(c,13)
- insert(d,13)
- insert(e,11)
- decreaseKey(c,11)



Ablauf und Laufzeiten (amortisiert):

insert:	Einfügen in Bucket $B[\min(\text{msd}(\text{min}, \text{key}), K)]$	$O(\log C)$
deleteMin:	min = minimaler Schlüssel (aus Bucket i), Elemente aus Bucket i verschieben, Entfernen aus Bucket $B[-1]$	$O(\log C)$
decreaseKey:	Verschieben von altem in neues Bucket	$O(1)$

Beispiel:

■ insert(a,5)

■ insert(b,7)

■ deleteMin()

■ insert(c,13)

■ insert(d,13)

■ insert(e,11)

■ decreaseKey(c,11)

min = 0101

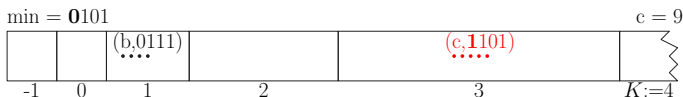


Ablauf und Laufzeiten (amortisiert):

insert:	Einfügen in Bucket $B[\min(\text{msd}(\text{min}, \text{key}), K)]$	$O(\log C)$
deleteMin:	min = minimaler Schlüssel (aus Bucket i), Elemente aus Bucket i verschieben, Entfernen aus Bucket $B[-1]$	$O(\log C)$
decreaseKey:	Verschieben von altem in neues Bucket	$O(1)$

Beispiel:

- insert(a,5)
- insert(b,7)
- deleteMin()
- insert(c,13)
- insert(d,13)
- insert(e,11)
- decreaseKey(c,11)



Ablauf und Laufzeiten (amortisiert):

insert:	Einfügen in Bucket $B[\min(\text{msd}(\text{min}, \text{key}), K)]$	$O(\log C)$
deleteMin:	min = minimaler Schlüssel (aus Bucket i), Elemente aus Bucket i verschieben, Entfernen aus Bucket $B[-1]$	$O(\log C)$
decreaseKey:	Verschieben von altem in neues Bucket	$O(1)$

Beispiel:

- insert(a,5)
- insert(b,7)
- deleteMin()
- insert(c,13)
- insert(d,13)
- insert(e,11)
- decreaseKey(c,11)

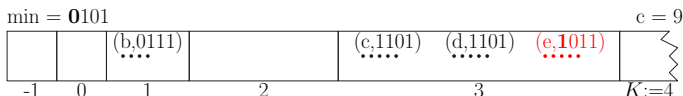


Ablauf und Laufzeiten (amortisiert):

insert:	Einfügen in Bucket $B[\min(\text{msd}(\text{min}, \text{key}), K)]$	$O(\log C)$
deleteMin:	min = minimaler Schlüssel (aus Bucket i), Elemente aus Bucket i verschieben, Entfernen aus Bucket $B[-1]$	$O(\log C)$
decreaseKey:	Verschieben von altem in neues Bucket	$O(1)$

Beispiel:

- insert(a,5)
- insert(b,7)
- deleteMin()
- insert(c,13)
- insert(d,13)
- insert(e,11)
- decreaseKey(c,11)

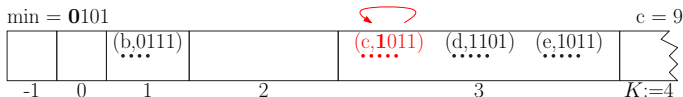


Ablauf und Laufzeiten (amortisiert):

insert:	Einfügen in Bucket $B[\min(\text{msd}(\text{min}, \text{key}), K)]$	$O(\log C)$
deleteMin:	min = minimaler Schlüssel (aus Bucket i), Elemente aus Bucket i verschieben, Entfernen aus Bucket $B[-1]$	$O(\log C)$
decreaseKey:	Verschieben von altem in neues Bucket	$O(1)$

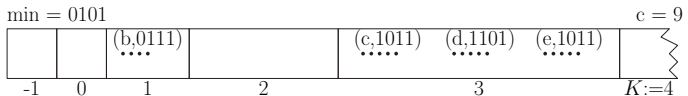
Beispiel:

- insert(a,5)
- insert(b,7)
- deleteMin()
- insert(c,13)
- insert(d,13)
- insert(e,11)
- decreaseKey(c,11)



Beispiel (fortgesetzt):

- insert(f,14)
- deleteMin()
- deleteMin()
- insert(g,20)
- deleteMin()
- insert(h,18)
- deleteMin()
- decreaseKey(g,16)
- deleteMin()
- deleteMin()
- insert(i,24)
- insert(j,22)
- decreaseKey(i,16)



Radix Heaps

Beispiel (fortgesetzt):

■ `insert(f,14)`

■ `deleteMin()`

■ `deleteMin()`

■ `insert(g,20)`

■ `deleteMin()`

■ `insert(h,18)`

■ `deleteMin()`

■ `decreaseKey(g,16)`

■ `deleteMin()`

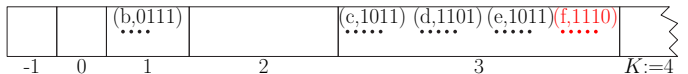
■ `deleteMin()`

■ `insert(i,24)`

■ `insert(j,22)`

■ `decreaseKey(i,16)`

min = 0101



Radix Heaps

Beispiel (fortgesetzt):

■ `insert(f,14)`

■ `deleteMin()`

■ `deleteMin()`

■ `insert(g,20)`

■ `deleteMin()` $\text{min} = 0101$

■ `insert(h,18)`

■ `deleteMin()`

■ `decreaseKey(g,16)`

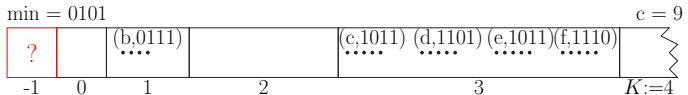
■ `deleteMin()`

■ `deleteMin()`

■ `insert(i,24)`

■ `insert(j,22)`

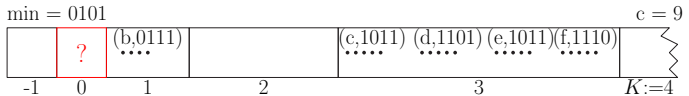
■ `decreaseKey(i,16)`



Radix Heaps

Beispiel (fortgesetzt):

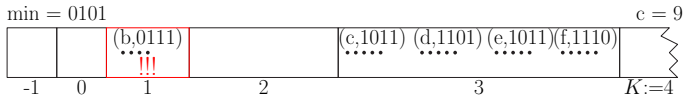
- `insert(f,14)`
- `deleteMin()`
- `deleteMin()`
- `insert(g,20)`
- `deleteMin()`
- `insert(h,18)`
- `deleteMin()`
- `decreaseKey(g,16)`
- `deleteMin()`
- `deleteMin()`
- `insert(i,24)`
- `insert(j,22)`
- `decreaseKey(i,16)`



Radix Heaps

Beispiel (fortgesetzt):

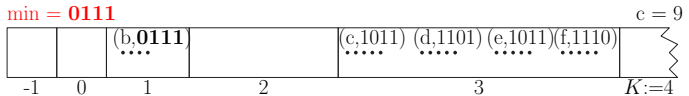
- `insert(f,14)`
- `deleteMin()`
- `deleteMin()`
- `insert(g,20)`
- `deleteMin()`
- `insert(h,18)`
- `deleteMin()`
- `decreaseKey(g,16)`
- `deleteMin()`
- `deleteMin()`
- `insert(i,24)`
- `insert(j,22)`
- `decreaseKey(i,16)`



Radix Heaps

Beispiel (fortgesetzt):

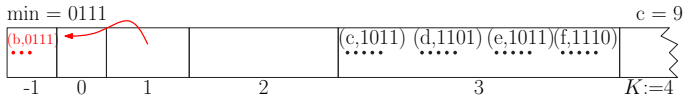
- `insert(f,14)`
- `deleteMin()`
- `deleteMin()`
- `insert(g,20)`
- `deleteMin()`
- `insert(h,18)`
- `deleteMin()`
- `decreaseKey(g,16)`
- `deleteMin()`
- `deleteMin()`
- `insert(i,24)`
- `insert(j,22)`
- `decreaseKey(i,16)`



Radix Heaps

Beispiel (fortgesetzt):

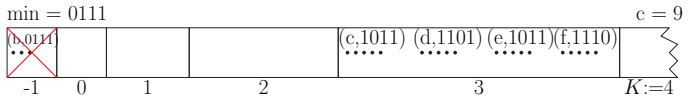
- `insert(f,14)`
- `deleteMin()`
- `deleteMin()`
- `insert(g,20)`
- `deleteMin()`
- `insert(h,18)`
- `deleteMin()`
- `decreaseKey(g,16)`
- `deleteMin()`
- `deleteMin()`
- `insert(i,24)`
- `insert(j,22)`
- `decreaseKey(i,16)`



Radix Heaps

Beispiel (fortgesetzt):

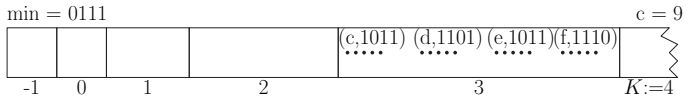
- `insert(f,14)`
- `deleteMin()`
- `deleteMin()`
- `insert(g,20)`
- `deleteMin()`
- `insert(h,18)`
- `deleteMin()`
- `decreaseKey(g,16)`
- `deleteMin()`
- `deleteMin()`
- `insert(i,24)`
- `insert(j,22)`
- `decreaseKey(i,16)`



Radix Heaps

Beispiel (fortgesetzt):

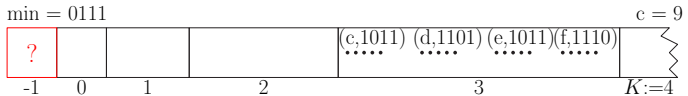
- `insert(f,14)`
- `deleteMin()`
- `deleteMin()`
- `insert(g,20)`
- `deleteMin()`
- `insert(h,18)`
- `deleteMin()`
- `decreaseKey(g,16)`
- `deleteMin()`
- `deleteMin()`
- `insert(i,24)`
- `insert(j,22)`
- `decreaseKey(i,16)`



Radix Heaps

Beispiel (fortgesetzt):

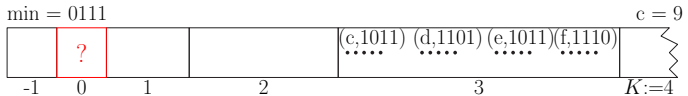
- `insert(f,14)`
- `deleteMin()`
- `deleteMin()`
- `insert(g,20)`
- `deleteMin()`
- `insert(h,18)`
- `deleteMin()`
- `decreaseKey(g,16)`
- `deleteMin()`
- `deleteMin()`
- `insert(i,24)`
- `insert(j,22)`
- `decreaseKey(i,16)`



Radix Heaps

Beispiel (fortgesetzt):

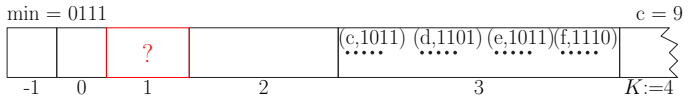
- `insert(f,14)`
- `deleteMin()`
- `deleteMin()`
- `insert(g,20)`
- `deleteMin()`
- `insert(h,18)`
- `deleteMin()`
- `decreaseKey(g,16)`
- `deleteMin()`
- `deleteMin()`
- `insert(i,24)`
- `insert(j,22)`
- `decreaseKey(i,16)`



Radix Heaps

Beispiel (fortgesetzt):

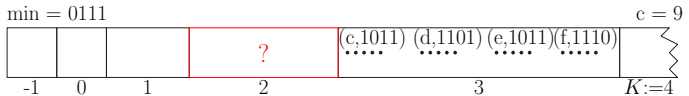
- `insert(f,14)`
- `deleteMin()`
- `deleteMin()`
- `insert(g,20)`
- `deleteMin()`
- `insert(h,18)`
- `deleteMin()`
- `decreaseKey(g,16)`
- `deleteMin()`
- `deleteMin()`
- `insert(i,24)`
- `insert(j,22)`
- `decreaseKey(i,16)`



Radix Heaps

Beispiel (fortgesetzt):

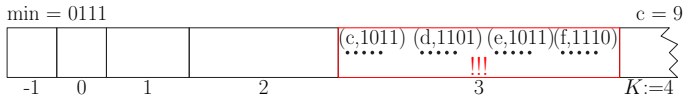
- `insert(f,14)`
- `deleteMin()`
- `deleteMin()`
- `insert(g,20)`
- `deleteMin()`
- `insert(h,18)`
- `deleteMin()`
- `decreaseKey(g,16)`
- `deleteMin()`
- `deleteMin()`
- `insert(i,24)`
- `insert(j,22)`
- `decreaseKey(i,16)`



Radix Heaps

Beispiel (fortgesetzt):

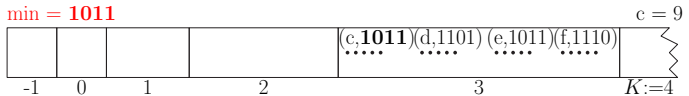
- `insert(f,14)`
- `deleteMin()`
- `deleteMin()`
- `insert(g,20)`
- `deleteMin()`
- `insert(h,18)`
- `deleteMin()`
- `decreaseKey(g,16)`
- `deleteMin()`
- `deleteMin()`
- `insert(i,24)`
- `insert(j,22)`
- `decreaseKey(i,16)`



Radix Heaps

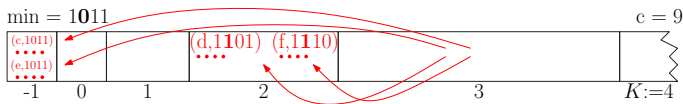
Beispiel (fortgesetzt):

- insert(f,14)
- deleteMin()
- deleteMin()
- insert(g,20)
- deleteMin()
- insert(h,18)
- deleteMin()
- decreaseKey(g,16)
- deleteMin()
- deleteMin()
- insert(i,24)
- insert(j,22)
- decreaseKey(i,16)



Beispiel (fortgesetzt):

- insert(f,14)
- deleteMin()
- deleteMin()
- insert(g,20)
- deleteMin()
- insert(h,18)
- deleteMin()
- decreaseKey(g,16)
- deleteMin()
- deleteMin()
- insert(i,24)
- insert(j,22)
- decreaseKey(i,16)



Radix Heaps

Beispiel (fortgesetzt):

■ `insert(f,14)`

■ `deleteMin()`

■ `deleteMin()`

■ `insert(g,20)`

■ `deleteMin()`

■ `insert(h,18)`

■ `deleteMin()`

■ `decreaseKey(g,16)`

■ `deleteMin()`

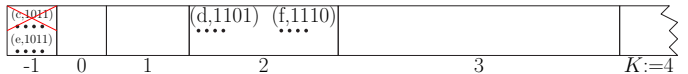
■ `deleteMin()`

■ `insert(i,24)`

■ `insert(j,22)`

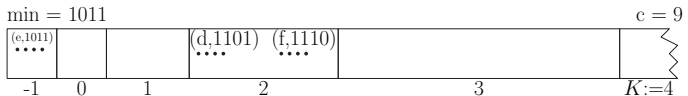
■ `decreaseKey(i,16)`

min = 1011



Beispiel (fortgesetzt):

- insert(f,14)
- deleteMin()
- deleteMin()
- insert(g,20)
- deleteMin()
- insert(h,18)
- deleteMin()
- decreaseKey(g,16)
- deleteMin()
- deleteMin()
- insert(i,24)
- insert(j,22)
- decreaseKey(i,16)



Radix Heaps

Beispiel (fortgesetzt):

■ insert(f,14)

■ deleteMin()

■ deleteMin()

■ insert(g,20)

■ deleteMin()

■ insert(h,18)

■ deleteMin()

■ decreaseKey(g,16)

■ deleteMin()

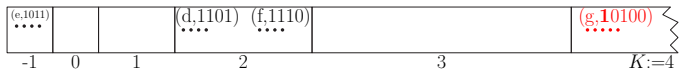
■ deleteMin()

■ insert(i,24)

■ insert(j,22)

■ decreaseKey(i,16)

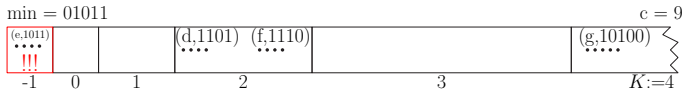
min = 01011



Radix Heaps

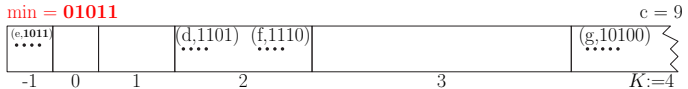
Beispiel (fortgesetzt):

- `insert(f,14)`
- `deleteMin()`
- `deleteMin()`
- `insert(g,20)`
- `deleteMin()`
- `insert(h,18)`
- `deleteMin()`
- `decreaseKey(g,16)`
- `deleteMin()`
- `deleteMin()`
- `insert(i,24)`
- `insert(j,22)`
- `decreaseKey(i,16)`



Beispiel (fortgesetzt):

- insert(f,14)
- deleteMin()
- deleteMin()
- insert(g,20)
- deleteMin()
- insert(h,18)
- deleteMin()
- decreaseKey(g,16)
- deleteMin()
- deleteMin()
- insert(i,24)
- insert(j,22)
- decreaseKey(i,16)



Radix Heaps

Beispiel (fortgesetzt):

■ insert(f,14)

■ deleteMin()

■ deleteMin()

■ insert(g,20)

■ deleteMin()

■ insert(h,18)

■ deleteMin()

■ decreaseKey(g,16)

■ deleteMin()

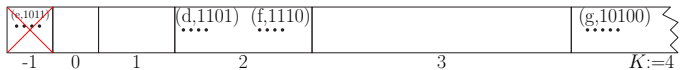
■ deleteMin()

■ insert(i,24)

■ insert(j,22)

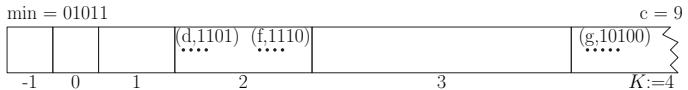
■ decreaseKey(i,16)

min = 01011



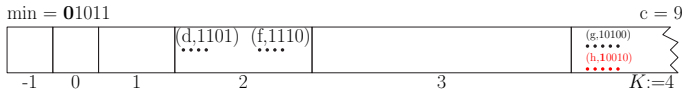
Beispiel (fortgesetzt):

- insert(f,14)
- deleteMin()
- deleteMin()
- insert(g,20)
- deleteMin()
- insert(h,18)
- deleteMin()
- decreaseKey(g,16)
- deleteMin()
- deleteMin()
- insert(i,24)
- insert(j,22)
- decreaseKey(i,16)



Beispiel (fortgesetzt):

- insert(f,14)
- deleteMin()
- deleteMin()
- insert(g,20)
- deleteMin()
- **insert(h,18)**
- deleteMin()
- decreaseKey(g,16)
- deleteMin()
- deleteMin()
- insert(i,24)
- insert(j,22)
- decreaseKey(i,16)

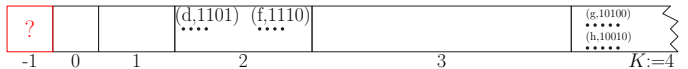


Radix Heaps

Beispiel (fortgesetzt):

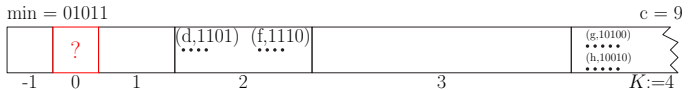
- insert(f,14)
- deleteMin()
- deleteMin()
- insert(g,20)
- deleteMin()
- insert(h,18)
- deleteMin()
- decreaseKey(g,16)
- deleteMin()
- deleteMin()
- insert(i,24)
- insert(j,22)
- decreaseKey(i,16)

min = 01011



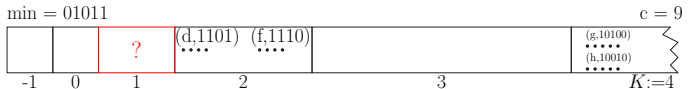
Beispiel (fortgesetzt):

- insert(f,14)
- deleteMin()
- deleteMin()
- insert(g,20)
- deleteMin()
- insert(h,18)
- **deleteMin()**
- decreaseKey(g,16)
- deleteMin()
- deleteMin()
- insert(i,24)
- insert(j,22)
- decreaseKey(i,16)



Beispiel (fortgesetzt):

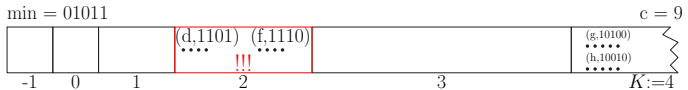
- insert(f,14)
- deleteMin()
- deleteMin()
- insert(g,20)
- deleteMin()
- insert(h,18)
- **deleteMin()**
- decreaseKey(g,16)
- deleteMin()
- deleteMin()
- insert(i,24)
- insert(j,22)
- decreaseKey(i,16)



Radix Heaps

Beispiel (fortgesetzt):

- insert(f,14)
- deleteMin()
- deleteMin()
- insert(g,20)
- deleteMin()
- insert(h,18)
- deleteMin()
- decreaseKey(g,16)
- deleteMin()
- deleteMin()
- insert(i,24)
- insert(j,22)
- decreaseKey(i,16)

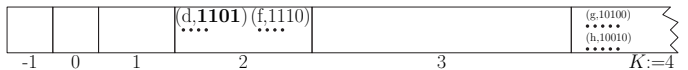


Radix Heaps

Beispiel (fortgesetzt):

- insert(f,14)
- deleteMin()
- deleteMin()
- insert(g,20)
- deleteMin()
- insert(h,18)
- deleteMin()
- decreaseKey(g,16)
- deleteMin()
- deleteMin()
- insert(i,24)
- insert(j,22)
- decreaseKey(i,16)

min = 01101



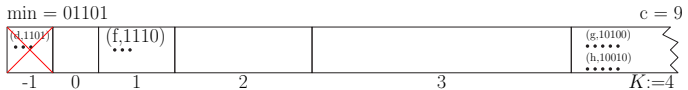
Beispiel (fortgesetzt):

- insert(f,14)
- deleteMin()
- deleteMin()
- insert(g,20)
- deleteMin()
- insert(h,18)
- **deleteMin()**
- decreaseKey(g,16)
- deleteMin()
- deleteMin()
- insert(i,24)
- insert(j,22)
- decreaseKey(i,16)



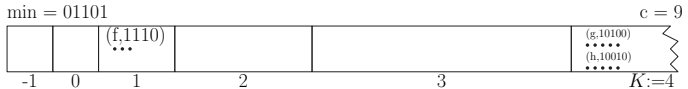
Beispiel (fortgesetzt):

- insert(f,14)
- deleteMin()
- deleteMin()
- insert(g,20)
- deleteMin()
- insert(h,18)
- **deleteMin()**
- decreaseKey(g,16)
- deleteMin()
- deleteMin()
- insert(i,24)
- insert(j,22)
- decreaseKey(i,16)



Beispiel (fortgesetzt):

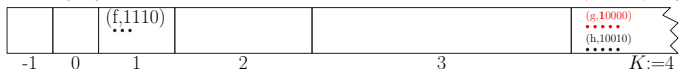
- insert(f,14)
- deleteMin()
- deleteMin()
- insert(g,20)
- deleteMin()
- insert(h,18)
- **deleteMin()**
- decreaseKey(g,16)
- deleteMin()
- deleteMin()
- insert(i,24)
- insert(j,22)
- decreaseKey(i,16)



Beispiel (fortgesetzt):

- insert(f,14)
- deleteMin()
- deleteMin()
- insert(g,20)
- deleteMin()
- insert(h,18)
- deleteMin()
- decreaseKey(g,16)
- deleteMin()
- deleteMin()
- insert(i,24)
- insert(j,22)
- decreaseKey(i,16)

min = 01101

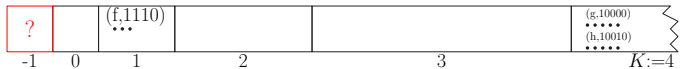


Radix Heaps

Beispiel (fortgesetzt):

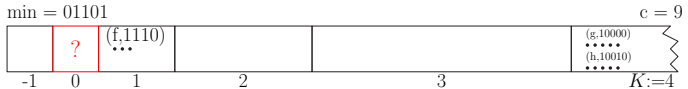
- insert(f,14)
- deleteMin()
- deleteMin()
- insert(g,20)
- deleteMin()
- insert(h,18)
- deleteMin()
- decreaseKey(g,16)
- deleteMin()
- deleteMin()
- insert(i,24)
- insert(j,22)
- decreaseKey(i,16)

min = 01101



Beispiel (fortgesetzt):

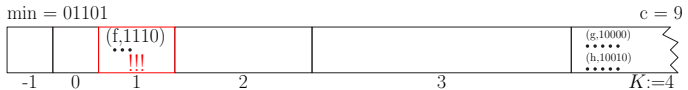
- insert(f,14)
- deleteMin()
- deleteMin()
- insert(g,20)
- deleteMin()
- insert(h,18)
- deleteMin()
- decreaseKey(g,16)
- deleteMin()
- deleteMin()
- insert(i,24)
- insert(j,22)
- decreaseKey(i,16)



Radix Heaps

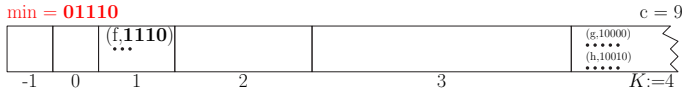
Beispiel (fortgesetzt):

- insert(f,14)
- deleteMin()
- deleteMin()
- insert(g,20)
- deleteMin()
- insert(h,18)
- deleteMin()
- decreaseKey(g,16)
- deleteMin()
- deleteMin()
- insert(i,24)
- insert(j,22)
- decreaseKey(i,16)



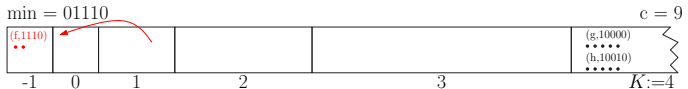
Beispiel (fortgesetzt):

- insert(f,14)
- deleteMin()
- deleteMin()
- insert(g,20)
- deleteMin()
- insert(h,18)
- deleteMin()
- decreaseKey(g,16)
- deleteMin()
- deleteMin()
- insert(i,24)
- insert(j,22)
- decreaseKey(i,16)



Beispiel (fortgesetzt):

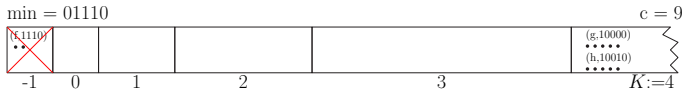
- `insert(f,14)`
- `deleteMin()`
- `deleteMin()`
- `insert(g,20)`
- `deleteMin()`
- `insert(h,18)`
- `deleteMin()`
- `decreaseKey(g,16)`
- `deleteMin()`
- `deleteMin()`
- `insert(i,24)`
- `insert(j,22)`
- `decreaseKey(i,16)`



Radix Heaps

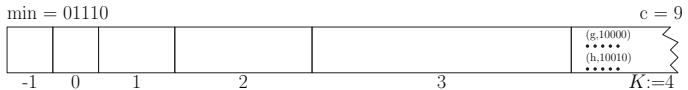
Beispiel (fortgesetzt):

- `insert(f,14)`
- `deleteMin()`
- `deleteMin()`
- `insert(g,20)`
- `deleteMin()`
- `insert(h,18)`
- `deleteMin()`
- `decreaseKey(g,16)`
- `deleteMin()`
- `deleteMin()`
- `insert(i,24)`
- `insert(j,22)`
- `decreaseKey(i,16)`



Beispiel (fortgesetzt):

- `insert(f,14)`
- `deleteMin()`
- `deleteMin()`
- `insert(g,20)`
- `deleteMin()`
- `insert(h,18)`
- `deleteMin()`
- `decreaseKey(g,16)`
- `deleteMin()`
- `deleteMin()`
- `insert(i,24)`
- `insert(j,22)`
- `decreaseKey(i,16)`



Beispiel (fortgesetzt):

- insert(f,14)
- deleteMin()
- deleteMin()
- insert(g,20)
- deleteMin()
- insert(h,18)
- deleteMin()
- decreaseKey(g,16)
- deleteMin()
- deleteMin()
- insert(i,24)
- insert(j,22)
- decreaseKey(i,16)

min = 01110



Beispiel (fortgesetzt):

- insert(f,14)
- deleteMin()
- deleteMin()
- insert(g,20)
- deleteMin()
- insert(h,18)
- deleteMin()
- decreaseKey(g,16)
- deleteMin()
- **deleteMin()**
- insert(i,24)
- insert(j,22)
- decreaseKey(i,16)

min = 01110



Radix Heaps

Beispiel (fortgesetzt):

- `insert(f,14)`
- `deleteMin()`
- `deleteMin()`
- `insert(g,20)`
- `deleteMin()`
- `insert(h,18)`
- `deleteMin()`
- `decreaseKey(g,16)`
- `deleteMin()`
- `deleteMin()`
- `insert(i,24)`
- `insert(j,22)`
- `decreaseKey(i,16)`

min = 01110



Beispiel (fortgesetzt):

- `insert(f,14)`
- `deleteMin()`
- `deleteMin()`
- `insert(g,20)`
- `deleteMin()`
- `insert(h,18)`
- `deleteMin()`
- `decreaseKey(g,16)`
- `deleteMin()`
- `deleteMin()`
- `insert(i,24)`
- `insert(j,22)`
- `decreaseKey(i,16)`

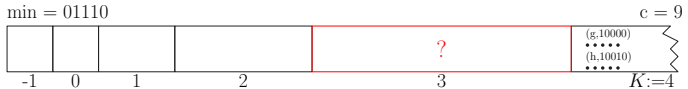
min = 01110



Radix Heaps

Beispiel (fortgesetzt):

- `insert(f,14)`
- `deleteMin()`
- `deleteMin()`
- `insert(g,20)`
- `deleteMin()`
- `insert(h,18)`
- `deleteMin()`
- `decreaseKey(g,16)`
- `deleteMin()`
- `deleteMin()`
- `insert(i,24)`
- `insert(j,22)`
- `decreaseKey(i,16)`



Beispiel (fortgesetzt):

- insert(f,14)
- deleteMin()
- deleteMin()
- insert(g,20)
- deleteMin()
- insert(h,18)
- deleteMin()
- decreaseKey(g,16)
- deleteMin()
- deleteMin()
- insert(i,24)
- insert(j,22)
- decreaseKey(i,16)

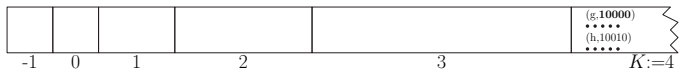
min = 01110



Beispiel (fortgesetzt):

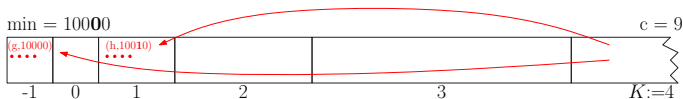
- insert(f,14)
- deleteMin()
- deleteMin()
- insert(g,20)
- deleteMin()
- insert(h,18)
- deleteMin()
- decreaseKey(g,16)
- deleteMin()
- deleteMin()
- insert(i,24)
- insert(j,22)
- decreaseKey(i,16)

min = 10000



Beispiel (fortgesetzt):

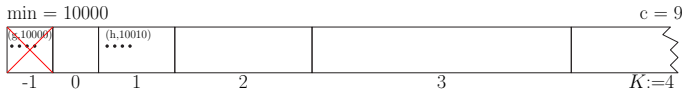
- `insert(f,14)`
- `deleteMin()`
- `deleteMin()`
- `insert(g,20)`
- `deleteMin()`
- `insert(h,18)`
- `deleteMin()`
- `decreaseKey(g,16)`
- `deleteMin()`
- `deleteMin()`
- `insert(i,24)`
- `insert(j,22)`
- `decreaseKey(i,16)`



Radix Heaps

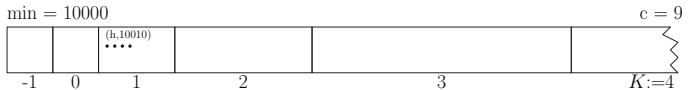
Beispiel (fortgesetzt):

- `insert(f,14)`
- `deleteMin()`
- `deleteMin()`
- `insert(g,20)`
- `deleteMin()`
- `insert(h,18)`
- `deleteMin()`
- `decreaseKey(g,16)`
- `deleteMin()`
- `deleteMin()`
- `insert(i,24)`
- `insert(j,22)`
- `decreaseKey(i,16)`



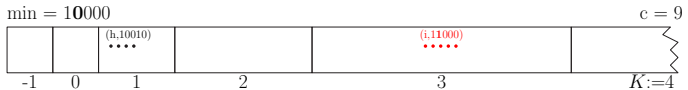
Beispiel (fortgesetzt):

- `insert(f,14)`
- `deleteMin()`
- `deleteMin()`
- `insert(g,20)`
- `deleteMin()`
- `insert(h,18)`
- `deleteMin()`
- `decreaseKey(g,16)`
- `deleteMin()`
- `deleteMin()`
- `insert(i,24)`
- `insert(j,22)`
- `decreaseKey(i,16)`



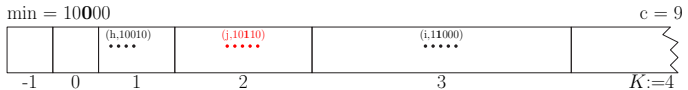
Beispiel (fortgesetzt):

- `insert(f,14)`
- `deleteMin()`
- `deleteMin()`
- `insert(g,20)`
- `deleteMin()`
- `insert(h,18)`
- `deleteMin()`
- `decreaseKey(g,16)`
- `deleteMin()`
- `deleteMin()`
- `insert(i,24)`
- `insert(j,22)`
- `decreaseKey(i,16)`



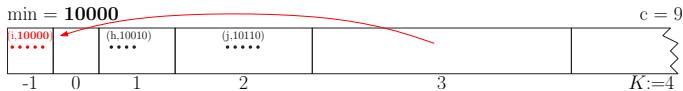
Beispiel (fortgesetzt):

- insert(f,14)
- deleteMin()
- deleteMin()
- insert(g,20)
- deleteMin()
- insert(h,18)
- deleteMin()
- decreaseKey(g,16)
- deleteMin()
- deleteMin()
- insert(i,24)
- insert(j,22)
- decreaseKey(i,16)



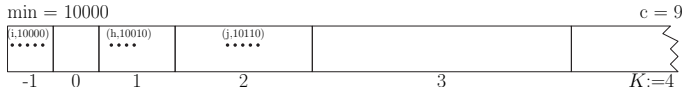
Beispiel (fortgesetzt):

- `insert(f,14)`
- `deleteMin()`
- `deleteMin()`
- `insert(g,20)`
- `deleteMin()`
- `insert(h,18)`
- `deleteMin()`
- `decreaseKey(g,16)`
- `deleteMin()`
- `deleteMin()`
- `insert(i,24)`
- `insert(j,22)`
- `decreaseKey(i,16)`



Beispiel (fortgesetzt):

- insert(f,14)
- deleteMin()
- deleteMin()
- insert(g,20)
- deleteMin()
- insert(h,18)
- deleteMin()
- decreaseKey(g,16)
- deleteMin()
- deleteMin()
- insert(i,24)
- insert(j,22)
- decreaseKey(i,16)



Warum komplizierte Formel $\min(\text{msd}(\text{min}, \text{key}), K)$?

- viel anschaulicher wäre doch

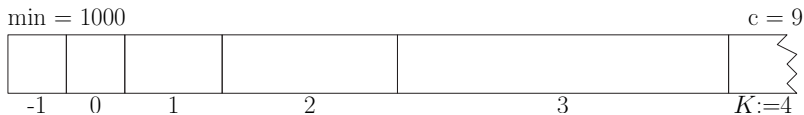
$$i = \lceil \log(\text{key} - \text{min}) \rceil$$

- okay, Anzahl Verschiebungen **erhöht sich nicht** ...
- ... aber, jede Änderung von *min* ändert **potentiell alle Buckets**
→ **schlechtere Laufzeit!**

Beispiel

$\text{min} := 01000 \text{ (08)} \xrightarrow{\text{deleteMin}} \text{min} := 01010 \text{ (10)}$

	Bucket $B[\cdot]$				
<i>min</i>	0	1	2	3	4
1000 (08)	8	9..10	11..12	13..16	17..8+C
1010 (10)	10	11..12	13..14	15..18	19..10+C



Änderung von min $\rightarrow$ Umverteilung eines Bucket genügt

$min := 01000$ (08)

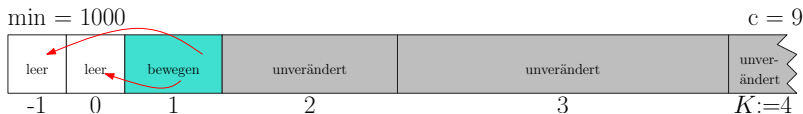
	Bucket $B[.]$					
min	-1	0	1	2	3	4
01000 (08)	8	9	10..11	12..15	-	≥ 16

$\min = 1000$
 $c = 9$


Änderung von $\min \rightarrow$ Umverteilung eines Bucket genügt

$\min := 01000$ (08) $\xrightarrow{\text{deleteMin}}$ $\min := 01001$ (09)

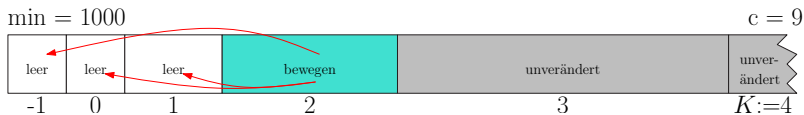
	Bucket $B[.]$					
$\min$	-1	0	1	2	3	4
01000 (08)	8	9	10..11	12..15	-	≥ 16
01001 (09)	9	-	10..11	12..15	-	≥ 16



Änderung von min $\rightarrow$ Umverteilung eines Bucket genügt

$min := 01000 (08) \xrightarrow{\text{deleteMin}} min := 0101 * (10..11)$

	Bucket $B[\cdot]$					
min	-1	0	1	2	3	4
01000 (08)	8	9	10..11	12..15	-	≥ 16
01010 (10)	10	11	-	12..15	-	≥ 16
01011 (11)	11	-	-	12..15	-	≥ 16

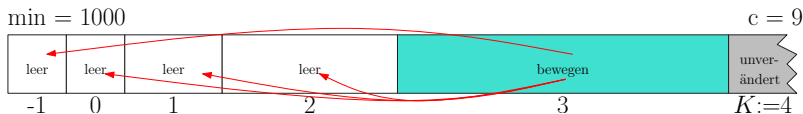


Änderung von min $\rightarrow$ Umverteilung eines Bucket genügt

$min := 01000 (08) \xrightarrow{\text{deleteMin}} min := 011 ** (12..15)$

	Bucket $B[.]$					
min	-1	0	1	2	3	4
01000 (08)	8	9	10..11	12..15	-	≥ 16

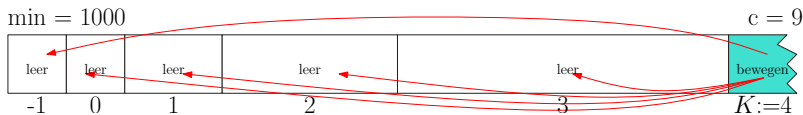
01100 (12)	12	13	14..15	-	-	≥ 16
01101 (13)	13	-	14..15	-	-	≥ 16
01110 (14)	14	15	-	-	-	≥ 16
01111 (15)	15	-	-	-	-	≥ 16



Änderung von min $\rightarrow$ Umverteilung eines Bucket genügt

$min := 01000 (08) \xrightarrow{\text{deleteMin}} min := -, \text{ Bucket 3 ist leer}$

	Bucket $B[.]$					
min	-1	0	1	2	3	4
01000 (08)	8	9	10..11	12..15	-	≥ 16



Änderung von min $\rightarrow$ Umverteilung eines Bucket genügt

$min := 01000$ (08) $\xrightarrow{\text{deleteMin}}$ $min := 1****$ (16..)

	Bucket $B[.]$					
min	-1	0	1	2	3	4
01000 (08)	8	9	10..11	12..15	-	≥ 16

10000 (16)	16	17	-	-	-	-
10001 (17)	17	-	-	-	-	-

Ende!



Feierabend!