

Blatt 4.

Aufgabe 1

H - Hashtabelle , $m=13$, $h_1, h_2, h_3 : \mathbb{N}_0 \rightarrow \{0, \dots, m-1\}$
 Hashfunktionen definiert wie folgt:

- $h_1 := x \bmod m$
- $h_2 := 3x \bmod m$
- $h_3 := x+1 \bmod m$

a) Lineares Sondieren unter der Benutzung von h_1 .

1) 23

$$h_1(23) = (2+3) \bmod 13 = 5$$

0	1	2	3	4	5	6	7	8	9	10	11	12
					23							

2) 12

$$h_1(12) = (1+2) \bmod 13 = 3$$

0	1	2	3	4	5	6	7	8	9	10	11	12
			12	23								

3) 75

$$h_1(75) = (7+5) \bmod 13 = 12$$

0	1	2	3	4	5	6	7	8	9	10	11	12
			12	23								75

4) 945

$$h_1(945) = (9+4+5) \bmod 13 = 5$$

0	1	2	3	4	5	6	7	8	9	10	11	12
			12	23	945							

$$\Rightarrow h_1(945, 1) = (h_1(945) + 1) \bmod 13 = 6 \bmod 13 = 6$$

0	1	2	3	4	5	6	7	8	9	10	11	12
			12	30	23	945						75

5) 30

$$h(30) = (3+0) \bmod 13 = 3$$

Kollision
 $\Rightarrow h(30, 1) = (h(30) + 1) \bmod 13 = 4$

6) 99

0	1	2	3	4	5	6	7	8	9	10	11	12
			12	30	23	945	99					75

$$h(99) = (9+9) \bmod 13 = 5$$

Kollision

$$\Rightarrow h(99, 1) = (h(99) + 1) \bmod 13 = 6$$

Kollision

$$\Rightarrow h(99, 2) = (h(99) + 2) \bmod 13 = 7$$

7) 345

0	1	2	3	4	5	6	7	8	9	10	11	12
345			12	30	23	945	99					75

$$h(345) = (3+4+5) \bmod 13 = 12$$

Kollision

$$\Rightarrow h(345, 1) = (h(345) + 1) \bmod 13 = 0$$

b) Doppel-Hashing unter Benutzung von h_2 und h_3 .

1) 23

0	1	2	3	4	5	6	7	8	9	10	11	12
				23								

$$h(23) = (h_2(23) + 0 \cdot h_3(23)) \bmod 13 = 4$$

2) 12

0	1	2	3	4	5	6	7	8	9	10	11	12
			23						12			

$$h(23) = (h_2(12) + 1 \cdot h_3(12)) \bmod 13 = 10$$

3) 75

0	1	2	3	4	5	6	7	8	9	10	11	12
75				23					12			

$$h(75) = (h_2(75) + 2 \cdot h_3(75)) \bmod 13 = 0$$

4) 945

0	1	2	3	4	5	6	7	8	9	10	11	12
75			23	945					12			

$$h(945) = (h_2(945) + 3 \cdot h_3(945)) \bmod 13 = 6$$

5) 30

0	1	2	3	4	5	6	7	8	9	10	11	12
75			23	945	30				12			

$$h(30) = (h_2(30) + 4 \cdot h_3(30)) \bmod 13 = 6$$

6) 99

0	1	2	3	4	5	6	7	8	9	10	11	12
75			23	945	30			99	12			

$$h(99) = (h_2(99) + 5 \cdot h_3(99)) \bmod 13 = 6$$

$\xrightarrow{\text{Kollision}}$ $h(99) = (h_2(99) + 6 \cdot h_3(99)) \bmod 13 = 0$

$\xrightarrow{\text{Kollision}}$ $h(99) = (h_2(99) + 7 \cdot h_3(99)) \bmod 13 = 9$

7) 345

0	1	2	3	4	5	6	7	8	9	10	11	12
75			23	945	30	345		99	12			

$$h(345) = (h_2(345) + 8 \cdot h_3(345)) \bmod 13 = 7$$

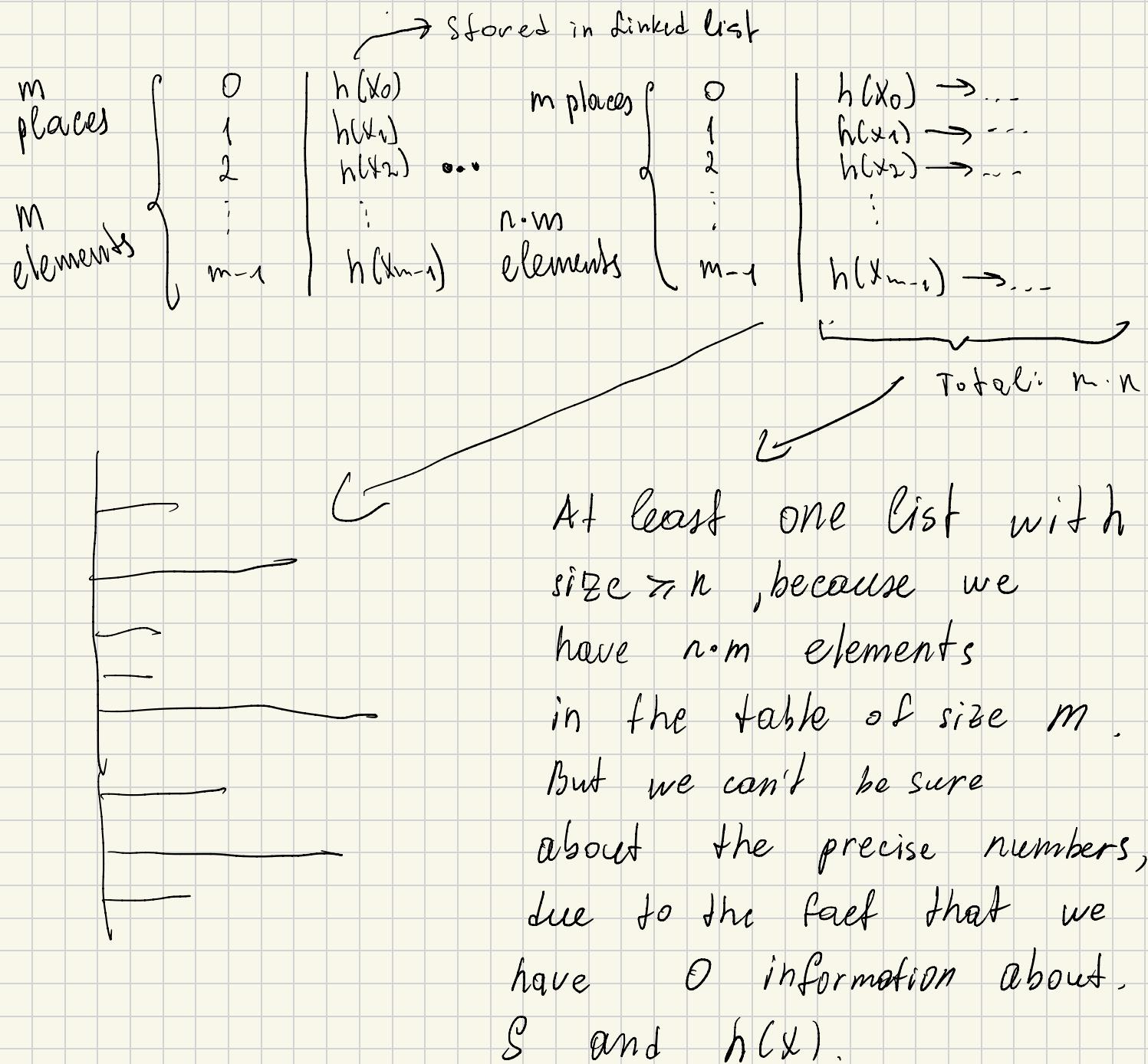
Aufgabe 2

(a) Da wir eine Hashtabelle der Länge m verwenden, aber die Menge S mit $|S| \geq y \cdot m$ Elemente haben, werden wir mindestens $ym - m$ Kollisionen haben (unter der Annahme, dass die ersten m Elemente der Tabelle vollständig gefüllt ^{haben} sind und keine Kollisionen verursacht haben). Da wir Hashing mit Chaining verwenden, werden im schlimmsten Fall, falls wir eine schlechte Hash-Funktion wählen oder falls die Menge viele sich wiederholende Elemente enthält, diese Elemente kontinuierlich zu denselben Liste hinzugefügt, was zu einer Menge Y mit $|Y| \geq y$ führt.

b) Dieses Ergebnis zeigt uns, dass wir im schlimmsten Fall eine Liste der Länge durchsuchen müssen, was $\mathcal{O}(y)$ dauern kann, wenn sich das Element ganz am Ende der Liste befindet.

Sorry if it was hard to read. Hope that the example below will help.

Z.B. $|S|=n \cdot 10$, $m=10$ $h: S \rightarrow \{0, \dots, m-1\}$.



Aufgabe 3

- a) Gesucht wird ein Element in einem Array der Länge n , das die Differenz von zwei anderen Elementen ist. Wenn sich die gesuchten Elemente ganz am Ende des Arrays befinden, müssen wir $O((n-1)(n-1)n) =$

$= O(n^3)$ Operationen durchführen, um sie zu finden,
da wir 3 verschachtelte Schleifen durchlaufen müssen.

b) Zuerst können wir alle Elemente des Arrays in
eine Hashtabelle H kopieren, was $O(n)$ Zeit benötigt.

Dannach durchlaufen wir zwei Schleifen und prüfen, ob die
Differenz $|A_{\Sigma i} - A_{\Sigma j}| \in H$.

Da wir zuerst die Hashtabelle erstellen und dann zwei
Schleifen durchlaufen, benötigen wir im schlimmsten Fall,
wenn die Elemente ganz am Ende sind, $O(n + n^2) = O(n^2)$
Zeit.

c) Wir können das Array zuerst sortieren, z.B. mit
Merge Sort, und dann in der zweiten Schleife
die binäre Suche verwenden.

Dazu benötigen wir im schlimmsten Fall

$$O(n \log n + n \cdot n \cdot \log n) = O(n^2 \log n) \text{ Zeit.}$$

$\xrightarrow{\text{Merge Sort}}$ \downarrow $\xrightarrow{\text{Erste Schleife}}$ $\xrightarrow{\text{Zweite Schleife}}$ $\xrightarrow{\text{Binary Search}}$