## Tutoraufgabe 1 (Hashing mit Verkettung):

**a)** Fügen Sie die folgenden Werte in das unten stehende Array der Länge 13 unter Verwendung der Divisionsmethode ohne Sondierung (also durch Verkettung) ein:

60, 64, 63, 49, 56, 44, 68, 86, 80, 7, 13, 14, 19, 20, 32.

0:

1:

2:

3:

4:

5:

6:

7:

8:

9:

10:

11:

12:

**b)** Fügen Sie die folgenden Werte in das unten stehende Array der Länge 13 unter Verwendung der Multiplikationsmethode ($c = 0.62$) ohne Sondierung (also durch Verkettung) ein:

44, 38, 9, 17, 67, 48, 14, 83, 95, 13, 28, 42, 78, 51, 31.

0:

1:

2:

3:

4:

5:

6:

7:

8:

9:

10:

11:

12:

## Lösung:

**a)** m = 13:

0: 13

1: 14

2: 80

3: 68

4: 56

5: 44

6: 19, 32

7: 7, 20

8: 60, 86

9:

10: 49

11: 63

12: 64

**b)** m = 13, c = 0.62:

0: 13, 42

1:

2: 31

3: 44

4: 28, 78

5: 83

6:

7: 38, 9, 17, 67

8: 14, 51

9: 48

10:

11: 95

12:

## Aufgabe 2 (Hashing mit Verkettung): (1 + 1 = 2 Punkte)

**a)** Fügen Sie die folgenden Werte in das unten stehende Array der Länge 13 unter Verwendung der Divisionsmethode ohne Sondierung (also durch Verkettung) ein:

36, 74, 39, 57, 21, 53, 19, 34, 20, 58, 69, 10, 42, 61, 94.

0:

1:

2:

3:

4:

5:

6:

7:

8:

9:

10:

11:

12:

**b)** Fügen Sie die folgenden Werte in das unten stehende Array der Länge 13 unter Verwendung der Multiplikationsmethode ($c = 0.62$) ohne Sondierung (also durch Verkettung) ein:

55, 54, 37, 36, 5, 15, 26, 33, 88, 25, 11, 45, 29, 68, 22.

0:

1:

2:

3:

4:

5:

6:

7:

8:

9:

10:

11:

12:

## Lösung:

**a)** m = 13:

0: 39

1: 53

2:

3: 42, 94

4: 69

5: 57

6: 19, 58

7: 20

8: 21, 34

9: 74, 61

10: 36, 10

11:

12:

**b)** m = 13, c = 0.62:

0:

1: 55, 5, 26

2: 68

3: 15

4: 36

5: 33

6: 54, 25

7: 88

8: 22

9:

10: 11

11: 45

12: 37, 29

## Tutoraufgabe 3 (Hashing mit Kollisionsbehandlung):

**a)** Fügen Sie die folgenden Werte in das unten stehende Array der Länge 11 unter Verwendung der Divisionsmethode mit linearer Sondierung ein:

63, 2, 5, 89, 33, 74, 95, 42, 5, 86.

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | |

**b)** Fügen Sie die folgenden Werte in das unten stehende Array der Länge 13 unter Verwendung der Divisionsmethode mit quadratischer Sondierung ($c_1$= 3.0, $c_2$= 7.0 ) ein:

20, 93, 90, 42, 21, 67, 81, 72, 97, 84.

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

**c)** Fügen Sie die folgenden Werte in das unten stehende Array der Länge 11 unter Verwendung der Multiplikationsmethode (c = 0.56) mit linearer Sondierung ein:

16, 22, 54, 27, 52, 57, 97, 58, 97, 80.

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | |

**d)** Fügen Sie die folgenden Werte in das unten stehende Array der Länge 11 unter Verwendung der Multiplikationsmethode (c = 0.23) mit quadratischer Sondierung ($c_1$= 2.0, $c_2$= 5.0 ) ein:

74, 96, 95, 52, 26, 24, 6, 52, 22, 31.

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | |

### Lösung:

**a)** m = 11:

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 33 | 89 | 2 | 86 | | 5 | 5 | 95 | 63 | 74 | 42 |

**b)** m = 13, $c_1$ = 3.0, $c_2$ = 7.0:

| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 81 | 84 | 93 | 42 | 72 | | 97 | 20 | 21 | | 67 | | 90 |

**c)** m = 11, c = 0.56:

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 57 | 27 | 54 | 22 | 52 | 97 | 58 | 97 | 80 | | 16 |

**d)** m = 11, c = 0.23, $c_1$ = 2.0, $c_2$ = 5.0:

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 74 | 52 | 22 | | 6 | 24 | 26 | 96 | 31 | 95 | 52 |

## Aufgabe 4 (Hashing mit Kollisionsbehandlung): (1 + 2 + 1 + 2 = 6 Punkte)

**a)** Fügen Sie die folgenden Werte in das unten stehende Array der Länge 11 unter Verwendung der Divisionsmethode mit linearer Sondierung ein:

53, 80, 98, 41, 35, 75, 44, 17, 48, 96.

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | |

**b)** Fügen Sie die folgenden Werte in das unten stehende Array der Länge 11 unter Verwendung der Divisionsmethode mit quadratischer Sondierung ($c_1$= 3.0, $c_2$= 7.0 ) ein:

30, 6, 9, 31, 64, 97, 96.

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | |

**c)** Fügen Sie die folgenden Werte in das unten stehende Array der Länge 11 unter Verwendung der Multiplikationsmethode (c = 0.34) mit linearer Sondierung ein:

46, 45, 98, 13, 29, 46, 1, 93, 2, 90.

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | |

**d)** Fügen Sie die folgenden Werte in das unten stehende Array der Länge 13 unter Verwendung der Multiplikationsmethode (c = 0.34) mit quadratischer Sondierung ($c_1$= 0.5, $c_2$= 0.5 ) ein:

10, 25, 23, 25, 64, 69, 47, 61, 73, 72.

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

Lösung:

**a)** m = 11:

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 75 | 44 | 35 | 80 | 48 | 96 | 17 | | 41 | 53 | 98 |

**b)** m = 11, $c_1$ = 3.0, $c_2$ = 7.0:

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | 97 | | | 64 | | 6 | 96 | 30 | 9 | 31 |

**c)** m = 11, c = 0.34:

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 2 | 90 | | 45 | 98 | 13 | 1 | 46 | 46 | 29 | 93 |

**d)** m = 13, c = 0.34, $c_1$ = 0.5, $c_2$ = 0.5:

| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | 61 | 72 | | 10 | 25 | 25 | 69 | 64 | 23 | 73 | 47 |

## Tutoraufgabe 5 (Hashfunktionen):

Erläutern Sie, warum die Verwendung der folgenden Funktionen $f_1$ und $f_2$ als Hashfunktionen für eine Hashtabelle der Größe $m$ von ganzen Zahlen problematisch ist. Hierbei ist $m$ eine Primzahl.

**a)** $f_1(x) = \overline{x}$

Hierbei bezeichnet $\overline{x}$ die Quersumme von $x$.

**b)** $f_2(x) = \lfloor \frac{x}{m} \rfloor \mod m$

Lösung:

---

**a)** Die Quersumme einer ganzen Zahl wächst für beliebig große Zahlen über jede Grenze hinaus. Somit können Werte, die größer als $m$ sind, nicht für die Adressenberechnung genutzt werden. Außerdem liegen die Hashwerte vieler benachbarten Zahlen ebenfalls direkt nebeneinander, wodurch keine gute Streuung gewährleistet wird (das trifft auch auf die reine Divisionsmethode aus der Vorlesung zu).

**b)** Hier ist die nicht optimale Streuung der Divisionsmethode durch die zusätzliche Division weiter verschlechtert, sodass benachbarte Zahlen nicht nur auf benachbarte, sondern sogar auf gleiche Hashwerte abgebildet werden. Dadurch kommt es häufig zu Kollisionen.

## Aufgabe 6 (Hashfunktionen): (2 + 3 = 5 Punkte)

Erläutern Sie, warum die Verwendung der folgenden Funktionen $f_3$ und $f_4$ als Hashfunktionen für eine Hashtabelle der Größe $m$ von ganzen Zahlen problematisch ist. Hierbei ist $m$ eine Primzahl.

**a)** $f_3(x) = (x \mod m) \oplus m$

Hier bezeichnet $\oplus$ die bitweise XOR Verknüpfung zweier ganzer Zahlen.

**b)** Sei $f_{perfect}$ eine perfekte Hashfunktion für eine Hashtabelle der Größe $m$ von ganzen Zahlen (also eine Funktion, die allen Anforderungen an eine Hashfunktion in optimaler Weise gerecht wird).

$f_4(x) = f_{perfect}(x)^{m-1} \mod m$

Lösung:

---

**a)** Obwohl die $\oplus$ Operation für eine gute Streuung sorgt, kann es dazu führen, dass die resultierenden Zahlen größer sind als die Hashtabelle. Damit können die Ergebnisse dieser Hashfunktion nicht immer zur Adressenberechnung verwendet werden.

**b)** Nach dem kleinen fermatschen Satz gilt für zwei teilerfremde Zahlen $a$ und $n$, wobei $n$ eine Primzahl ist, dass $a^{n-1} \mod n = 1$. Da $f_{perfect}$ nur auf Zahlen zwischen 0 und $m-1$ abbildet und $m$ eine Primzahl ist, bildet $f_4$ alle Zahlen auf 0 oder 1 ab. Damit ist $f_4$ nicht surjektiv und hat ein extrem schlechtes Streuungsverhalten.