Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Prüfung **(Lösung)**

Informatik II (D-BAUG)

Felix Friedrich, David Sidler

ETH Zürich, 14.8.2017.

Name, Vorname:	.....
Legi-Nummer:	.....

Ich bestätige mit meiner Unterschrift, dass ich diese Prüfung unter regulären Bedingungen ablegen konnte, und dass ich die allgemeinen Richtlinien gelesen und verstanden habe.

I confirm with my signature that I was able to take this exam under regular conditions and that I have read and understood the general guidelines.

Unterschrift:

Allgemeine Richtlinien:

1. Dauer der Prüfung: 120 Minuten.
2. Erlaubte Unterlagen: Wörterbuch (für gesprochene Sprachen). 4 A4 Seiten handgeschrieben oder $\geq$ 11pt Schriftgrösse.
3. Benützen Sie einen Kugelschreiber (blau oder schwarz) und keinen Bleistift. Bitte schreiben Sie leserlich. Nur lesbare Resultate werden bewertet.
4. Lösungen sind direkt auf das Aufgabenblatt in die dafür vorgesehenen Boxen zu schreiben (und direkt darunter, falls mehr Platz benötigt wird). Ungültige Lösungen sind deutlich durchzustreichen! Korrekturen bei Multiple-Choice Aufgaben bitte unmissverständlich anbringen!
5. Es gibt keine Negativpunkte für falsche Antworten.
6. Störungen durch irgendjemanden oder irgendetwas melden Sie bitte sofort der Aufsichtsperson.
7. Wir sammeln die Prüfung zum Schluss ein. Wichtig: stellen Sie unbedingt selbst sicher, dass Ihre Prüfung von einem Assistenten eingezogen wird. Stecken Sie keine Prüfung ein und lassen Sie Ihre Prüfung nicht einfach am Platz liegen. Dasselbe gilt, wenn Sie früher abgeben wollen: bitte melden Sie sich lautlos, und wir holen die Prüfung ab. Vorzeitige Abgaben sind nur bis 15 Minuten vor Prüfungsende möglich.
8. Wenn Sie zur Toilette müssen, melden Sie dies einer Aufsichtsperson durch Handzeichen.
9. Wir beantworten keine inhaltlichen Fragen während der Prüfung. Kommentare zur Aufgabe schreiben Sie bitte auf das Aufgabenblatt.

General guidelines:

1. *Exam duration: 120 minutes.*
2. *Permitted examination aids: dictionary (for spoken languages). 4 A4 pages hand written or $\geq$ 11pt font size*
3. *Use a pen (black or blue), not a pencil. Please write legibly. We will only consider solutions that we can read.*
4. *Solutions must be written directly onto the exam sheets in the provided boxes (and directly below, if more space is needed). Invalid solutions need to be crossed out clearly. Provide corrections to answers of multiple choice questions without any ambiguity!*
5. *There are no negative points for false answers.*
6. *If you feel disturbed by anyone or anything, let the supervisor of the exam know immediately.*
7. *We collect the exams at the end. Important: you must ensure that your exam has been collected by an assistant. Do not take any exam with you and do not leave your exam behind on your desk. The same applies when you want to finish early: please contact us silently and we will collect the exam. Handing in your exam ahead of time is only possible until 15 minutes before the exam ends.*
8. *If you need to go to the toilet, raise your hand and wait for a supervisor.*
9. *We will not answer any content-related questions during the exam. Please write comments referring to the tasks on the exam sheets.*

Question:	1	2	3	4	5	6	7	Total
Points:	7	4	6	6	5	6	8	42
Score:								

Generelle Anmerkung / *General Remark*

Verwenden Sie die Notation, Algorithmen und Datenstrukturen aus der Vorlesung. Falls Sie andere Methoden verwenden, müssen Sie diese kurz so erklären, dass Ihre Ergebnisse nachvollziehbar sind.

Use notation, algorithms and data structures from the course. If you use different methods, you need to explain them such that your results are reproducible.

Aufgabe 1: Verschiedenes (7P)

1) In dieser Aufgabe sollen nur Ergebnisse angegeben werden. Sie können sie direkt bei den Einzelteilen notieren.

   In this task only results have to be provided. You can note them down in the parts.

2) Als Ordnung verwenden wir für Buchstaben die alphabetische Reihenfolge, für Zahlen die aufsteigende Anordnung gemäss ihrer Grösse.

   As the order of characters we take the lexicographic order and for numbers we take the increasing order according to their sizes.

/1P (a) Gegeben seinen Daten der Länge n. Wie viele der folgenden Operationen führt der Algorithmus Sortieren durch Auswahl im schlechtesten Fall durch?

Given Data with length n. How many of the following operations are executed by the algorithm Selection Sort in the worst case?

Anzahl Vergleiche/*Number Comparisons* : ◯ $\Theta(n)$ ◯ $\Theta(n \log n)$ √ $\Theta(n^2)$

Anzahl Vertauschungen/*Number Swaps* : √ $\Theta(n)$ ◯ $\Theta(n \log n)$ ◯ $\Theta(n^2)$

/1P (b) Führen Sie auf dem folgenden Array zwei weitere Iterationen des Algorithmus Quicksort aus. Als Pivot wird jeweils das erste Element des (Sub-)Arrays genommen. Die Lösung zu dieser Aufgabe hängt von der gewählten Quicksort-Variante ab und ist nicht eindeutig. Unterstreichen Sie die Pivots.

Execute on the following array two further iterations of the algorithm Quicksort. The first element of the (sub-)array serves as the pivot. The solution to this task depends on the Quicksort variant chosen and is not unique. Underline the pivots.

8	7	9	15	3	6	10	2	5	13
5	7	2	6	3	**8**	10	15	9	13
3	2	**5**	6	7	**8**	9	**10**	15	13
2	**3**	**5**	**6**	7	**8**	**9**	**10**	**13**	15

(c) Im folgenden ist eine Hashtabelle dargestellt. Es wird offenes Hashing mit der Hashfunktion $h(k) = k \bmod 11$ verwendet. Kollisionen werden mit linearem Sondieren aufgelöst. Die Sondierung läuft immer nach links. Vervollständigen Sie die angegebene Einfügereihenfolge so, dass die Belegung entsteht, wenn man mit einer zu Anfang leeren Hashtabelle startet. Wie viele Kollisionen sind insgesamt aufgetreten?

In the following a hash table is displayed. Open hashing with the hash function $h(k) = h \bmod 11$ is used. Collisions are resolved using linear probing. Probing always goes left. Complete the given insertion order such that the hash table as given would be produced from an initially empty hash table. How many collisions took place overal?

/2P

0	1	2	3	4	5	6	7	8	9	10
		18	3	16	5	7	29			

Einfügereihenfolge/*Insertion Order*: 3, 5, 16, 29, 7, 18

Anzahl Kollisionen/*Number collisions*: ◯ 3 ◯ 4 ◯ 5 ◯ 6 √ 7

Gegeben seien nun folgende Daten, eine Sequenz von Zahlen

Now consider the following data, a sequence of numbers.

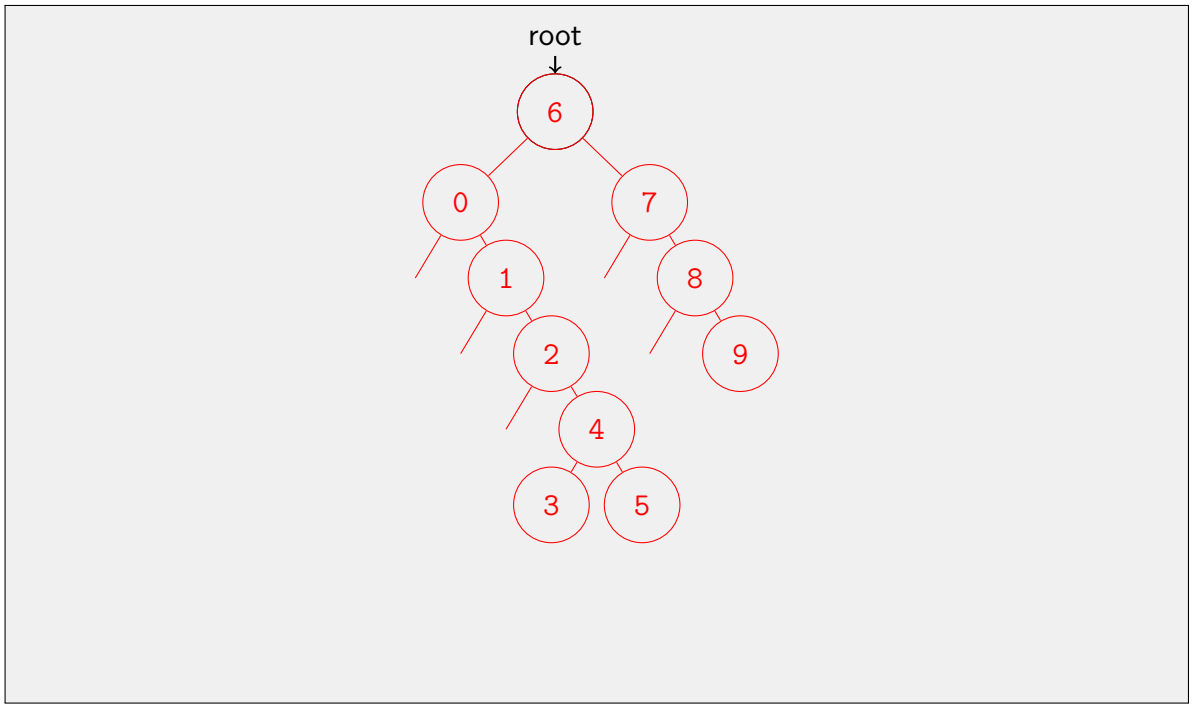
6, 7, 8, 9, 0, 1, 2, 4, 3, 5

Vervollständigen Sie die Figuren auf der nächsten Seite, so dass sie deutlich machen, wie die jeweilige Datenstruktur – wie in Vorlesung und Übungen gezeigt – aussieht nach Einfügen der gegebenen Daten in obiger Reihenfolge (von links nach rechts).

Complete the figures on the next page such that they illustrate the respective data structures – as presented in lectures and exercises – after insertion of the given data in the order above (from left to right).

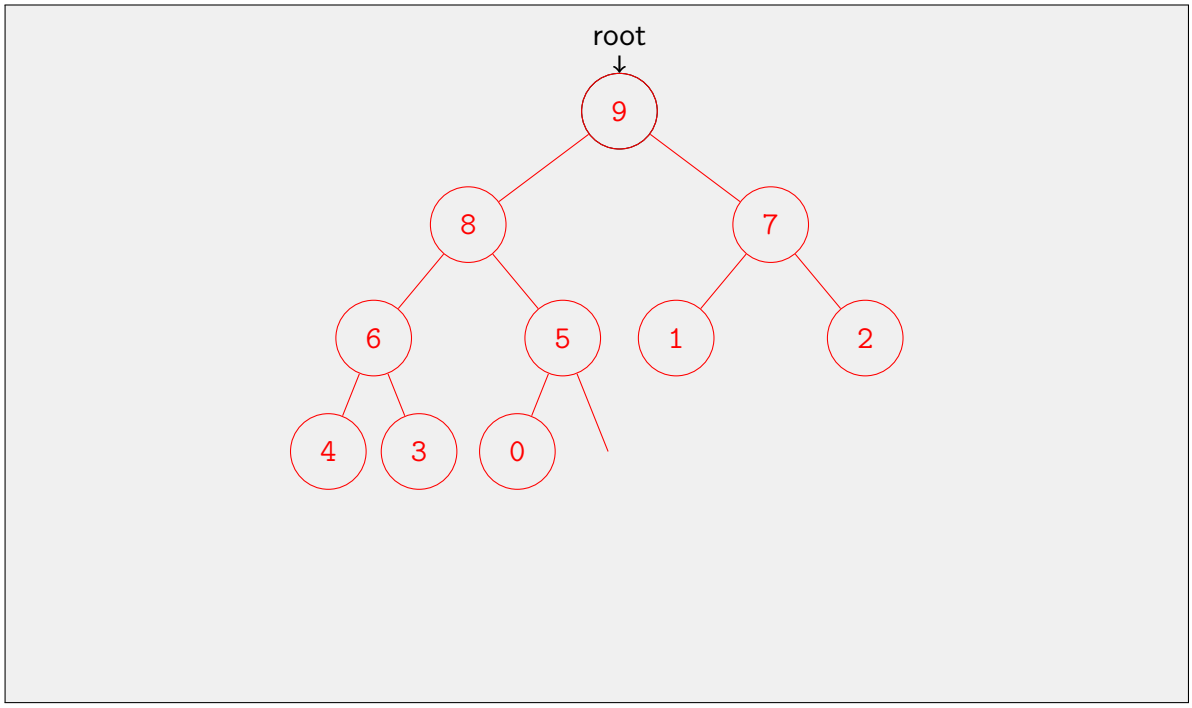
/1P (d) Binärer (unbalancierter) Suchbaum *Binary (unbalanced) search tree*

daten / *data*: 6, 7, 8, 9, 0, 1, 2, 4, 3, 5



/2P (e) Max-Heap (als Baum repräsentiert). Tipp: zeichnen Sie sich Zwischenschritte auf. *Max-Heap (represented as Tree). Hint: draw intermediate steps.*

daten / *data*: 6, 7, 8, 9, 0, 1, 2, 4, 3, 5



Aufgabe 2: Code Schreiben (4P)

/4P

Vervollständigen Sie folgende Funktion `minDist` so, dass Sie die minimale absolute Differenz $|x - y|$ zweier unterschiedlicher Werte x und y aus dem Array a zurückgibt.

Complement the following function `minDist` *such that it returns the minimal absolute difference* $|x - y|$ *of two different points* x *and* y *from the array* a.

```
// pre: a != null, a.length >= 2
// post: return distance Math.abs(a[i]-a[j]) of closest pair i and j (i != j)
public static int minDist(int a[]){
```

```
int dist = Math.abs(a[0]-a[1]);
for (int i = 0; i<a.length; ++i){
    for (int j = i+1; j<a.length; ++j){
        int d = Math.abs(a[i]-a[j]);
        if (d < dist){
            dist = d;
        }
    }
}
return dist;
```

```
}
```

Aufgabe 3: Asymptotik (6P)

/2P

(a) Geben Sie für die untenstehenden Funktionen eine Reihenfolge an, so dass folgendes gilt: Wenn eine Funktion f links von einer Funktion g steht, dann gilt $f \in \mathcal{O}(g)$.
Beispiel: die drei Funktionen n^3, n^5 und n^7 sind bereits in der entsprechenden Reihenfolge, da $n^3 \in \mathcal{O}(n^5)$ und $n^5 \in \mathcal{O}(n^7)$.

Provide for the following functions an order such that the following holds: If a function g *is left of* g *then it holds that* $f \in \mathcal{O}(g)$. *Example: the functions* n^3, n^5 *and* n^7 *are already in the respective order because* $n^3 \in \mathcal{O}(n^5)$ *and* $n^5 \in \mathcal{O}(n^7)$.

$$n!, \quad \log(\sum_{i=1}^{n} i), \quad n \log \sqrt{n}, \quad \sqrt{n} \log n, \quad \sum_{i=1}^{n} i, \quad 1/n$$

$1/n, \quad \log(\sum_{i=1}^{n} i), \quad \sqrt{n} \log n, \quad n \log \sqrt{n}, \quad \sum_{i=1}^{n} i, \quad n!$

In den folgenden Aufgabenteilen wird jeweils angenommen, dass die Funktion g mit $g(n)$ aufgerufen wird. Geben Sie jeweils die asymptotische Anzahl von Aufrufen der Funktion $f()$ in Abhängigkeit von $n \in \mathbb{N}$ mit Θ-Notation an. Die Funktion f ruft sich nicht selbst auf. Sie müssen Ihre Antworten nicht begründen.

In the following parts of this task we assume that the function g is called as $g(n)$. Provide the asymptotic number of calls of $f()$ as a function of $n \in \mathbb{N}$ using Θ notation. The function f does not call itself. You do not have to justify your answers.

/1P (b)

```
void g(int n){
    for (int i = 1; i<n ; i++ )
        f()
}
```

Anzahl Aufrufe von f / *Number Calls of f*

$\Theta(n)$

/1P (c)

```
void g(int n){
    for (int i = 1; i<n ; i++ )
        for (int j = 1; j<i; ++j)
            f()
}
```

Anzahl Aufrufe von f / *Number Calls of f*

$\Theta(n^2)$

/1P (d)

```
void g(int n){
    if (n > 0){
        g(n-1);
    }
    f();
}
```

Anzahl Aufrufe von f / *Number Calls of f*

$\Theta(n)$

/1P (e)

```
void g(int n){
    for (int i = 1; i<n ; i*=2 )
        f()
}
```

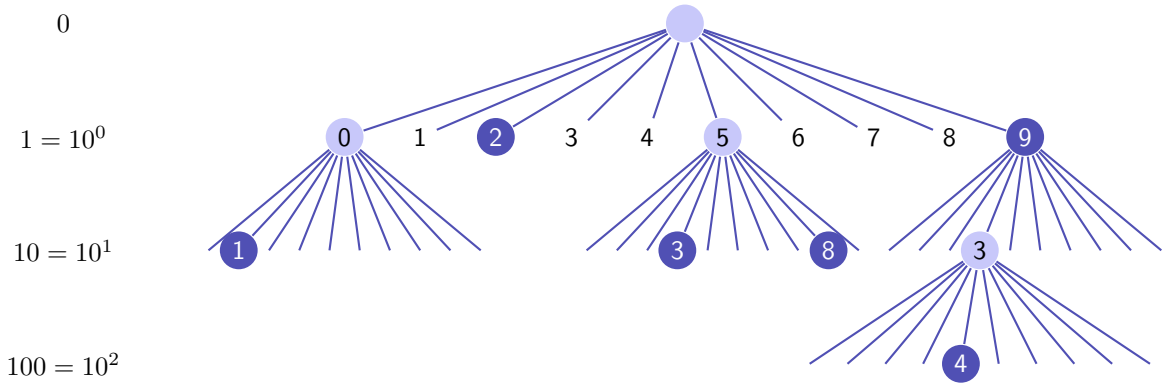
Anzahl Aufrufe von f / *Number Calls of f*

$\Theta(\log n)$

Aufgabe 4: Bäume (6P)

In der Vorlesung wurden binäre Suchbäume behandelt. In dieser Aufgabe werden wir sie zu Tries verallgemeinern. Tries werden manchmal als Alternative zu Hashtabellen verwendet und stellen typischerweise Bäume mit Ordnung grösser als zwei dar. Der in dieser Aufgabe vorgestellte Baum hat die Ordnung 10: jeder Knoten hat 10 Kinder.

In the course we presented binary search trees. In this task they are generalized to tries. Tries are sometimes used as an alternative to hash-tables and usually represent trees with an order greater than two. The tree proposed in this task has an order of 10: every node has 10 children.



Unser Baum speichert beliebige positive ganze Zahlen. Jede Ebene des Baumes (ausser der Wurzel) steht für eine Zehnerpotenz. Die Zehnerpotenz ist auf der linken Seite des Baumes dargestellt. Eine Zahl ist dann im Baum enthalten, wenn die Knoten Ihrer Darstellung als Dezimalzahl im Baum enthalten sind und der Blattknoten der Zahl markiert ist (im Graphen dargestellt durch dunkle Einfärbung). Der Baum in obiger Abbildung enthält so die Zahlen 10, 2, 35, 85, 9 und 439.

Our tree can store arbitrary positive integers. Every level of the tree (besides the roots) stands for a power of ten. The powers of ten are displayed on the left of the tree. A number is contained in the tree if the nodes of its representation as decimal number are contained in the tree and if the leaf node of a number is marked (represented with a dark background). The tree in the figure above contains the numbers 10, 2, 35, 85, 9 *and* 439.

Weiter unten und auf der nächsten Seite sehen Sie die Definition der Klassen TrieNode und Trie. Beantworten Sie nun folgende Fragen.

Further below and on the next page you find the definition of the classes TrieNode and Trie. Answer the following questions.

/1P (a) Unter der Voraussetzung, dass die Methode `insert` der Klasse `Trie` korrekt implementiert ist. Was ist dann die Ausgabe des folgenden Codestückes?

Provided that the method `insert` of the class `Trie` is correctly implemented. What is then the output of the following piece of code?

```
Trie t = new Trie();
t.Insert(90); t.Insert(5); t.Insert(99); t.Insert(1); t.Insert(199);
t.print();
```

Ausgabe / *Output* `90 1 5 99 199`

/2P (b) Vervollständigen Sie den Code der Methode `insert` der Klasse Trie, so dass die Zahl in den Trie entsprechend eingefügt wird.

Complement the code of the method `insert` of the class Trie such that the number is inserted in the Trie accordingly.

/2P (c) Vervollständigen Sie den Code der Methode `has` der Klasse Trie, so dass sie korrekt zurückgibt, ob die angegebene Zahl im Trie enthalten ist.

Complement the code of the method `has` of the class Trie such that it returns correctly if the Trie contains the number provided.

/1P (d) Charakterisieren Sie die asymptotische Laufzeit der Methode `has` in Abhängigkeit von der Anzahl n aller im Trie vorhandenen Zahlen. Gehen Sie von der realistischen Annahme aus, dass die eingegebenen Zahlen beschränkte Grösse haben.

Characterize the asymptotic runtime of the method `has` depending on the number n of all numbers contained in the Trie. You can safely assume that the numbers inserted are of limited size.

√ $\Theta(1)$ ○ $\Theta(\log n)$ ○ $\Theta(n)$ ○ $\Theta(n \log n)$ ○ $\Theta(n^2)$

```
public class TrieNode {
    TrieNode[] children;
    boolean marked;

    TrieNode() {
        children = new TrieNode[10];
        marked = false;
    }
}
```

```
public class Trie {
    TrieNode root; // root node
    Trie() { root = new TrieNode(); } // constructor: generate empty root

    // pre: number >= 0
    // post: number is inserted into trie;
    void insert(int number){
        TrieNode n = root;
        while (number > 0){
            if (n.children[number % 10] == null){
                n.children[number % 10] = new TrieNode();
            }
            n = n.children[number % 10];
            number = number / 10;
        }
        n.marked = true;
    }
    // pre: number >= 0
    // post: returns if number is contained in trie
    boolean has(int number){
        TrieNode n = root;
        while (n != null && number > 0 ){
            n = n.children[number % 10];
            number = number / 10;
        }
        return n != null && n.marked ;
    }

    void printNode(TrieNode n, int value, int dec){
        if (n == null){ return; }
        if (n.marked){ System.out.print(value + " "); }
        for (int i = 0; i<10; ++i){
            printNode(n.children[i],value+dec*i,dec*10);
        }
    }
    // post: output all numbers contained in the tree
    void print(){
        printNode(root,0,1);
    }
}
```

Aufgabe 5: Verkettete Listen (5P)

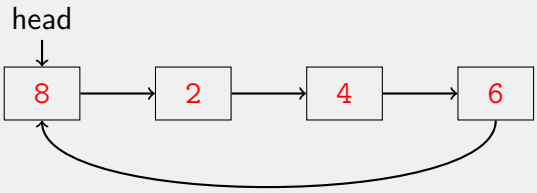
Die Datenstruktur `Ring` auf der nächsten Seite wird verwendet um eine ungeordnete Menge von Elementen zu speichern. Lesen Sie sich den Code genau durch und beantworten Sie dann folgende Fragen.

The data structure `Ring` on the next page is used to store an unordered set of elements. Read the code carefully and answer the following questions.

/2P (a) Vervollständigen Sie die Datenstruktur, welche durch folgendes Codestück erstellt wird.

Complement the data structure that is generated by the following code piece.

```
Ring r = new Ring();
r.put(2); r.put(4); r.put(6); r.put(8);
```



/2P (b) Wie viele Vergleiche der Form `n.value == value` in Zeile 30 finden beim Ausführen des folgenden Codes statt? Geben Sie die Teilresultate für die vier einzelnen Ausführungen von `r.has` an.

How many comparisons of the form `n.value == value` in line 30 take place overall when the following code is executed? Provide the partial results for the four executions of the method `r.has`.

```
Ring r = new Ring();
r.put(2); r.put(4);
boolean b = r.has(2); b = r.has(2); b = r.has(4); b = r.has(4);
```

Anzahl Vergleiche / *Number comparisons*: 2 + 1 + 1 + 2 = 6

/1P (c) Geben Sie für das folgenden Codebeispiel die asymptotische Anzahl von Zugriffen auf das `next`-Feld der Node-Datenstruktur in Abhängigkeit von $n \in \mathbb{N}$ in Θ-Notation an.

Provide for the following code piece the asymptotic number of accesses to the `next`-field of the Node data structure as a function of $n \in \mathbb{N}$ in Θ-Notation.

```
Ring r = new Ring();
for (int i = 0; i < n ; ++i){
    r.put(i);
}
for (int i = 0; i < n ; ++i){
    boolean b = r.has(i);
}
```

Anzahl Operationen / *Number Operations*

$\Theta(n)$

```
1class Node{
2    Node next;
3    int value;
4
5    Node (int v, Node nxt){
6        value = v;
7        next = nxt;
8    }
9}
10
11class Ring{
12    Node head=null;
13
14    // enter new value into the ring
15    public void put(int value){
16        if (head == null){
17            head = new Node(value, null);
18            head.next = head;
19        }
20        else {
21            head.next = new Node(value, head.next);
22            head = head.next;
23        }
24    }
25    // return if value is contained
26    public boolean has(int value){
27        if (head == null) return false;
28        Node n = head;
29        do{
30            if (n.value == value){
31                head = n; // optimize later search for same element
32                return true;
33            }
34            n = n.next;
35        } while (n != head);
36    }
37    // output all elements
38    public void out(){
39        Node  n = head;
40        do{
41            System.out.print(n.value + " ");
42            n = n.next;
43        } while (n != head);
44    }
45}
```

/6P

Aufgabe 6: Der Aufzug (6P)

Sie sind als Angestellter einer Aufzugfirma mit der Aufgabe betraut worden, das typische Bewegungsmuster eines Aufzugs zu simulieren.
Dazu beobachten Sie von jedem Stockwerk aus die Ziele von jeweils 100 Fahrten und tragen diese in einer Tabelle wie folgt ein.

As an employee of a lift (elevator) company you are asked to simulate the typical movement pattern of a lift.
From each floor you observe the destination floor of 100 trips and enter that information into a table as follows

Etage / *Floor*	0	1	2	3
0	0	20	30	50
1	50	0	20	30
2	50	20	0	30
3	60	20	20	0

Sie betrachten diese Tabelle nun als Wahrscheinlichkeitsmatrix (mit Prozenten) und simulieren das Verhalten des Aufzugs: die Zeile x und Spalte y der Tabelle gibt an, mit welcher Wahrscheinlichkeit der Lift als nächstes zum Stockwerk y fährt, wenn er im Stockwerk x steht.
Vervollständigen Sie den Code der Klasse Lift so, dass folgende Main-Funktion das Verhalten des Aufzugs simuliert und zum Schluss zu jedem Stockwerk die Gesamtanzahl Aufenthalte ausgibt. Der Lift startet im Stockwerk 0.

You now treat this table as a probability matrix (with percentages) and now simulate the behavior of the lift: the value at row x and column y provides the probability that the lift next travels to floor y provided that it currently halts at floor x. Complement the code of the class Lift such that the following Main-function simulates the behavior of the lift and that it provides the overal number of stops for each floor. The lift starts in floor 0.

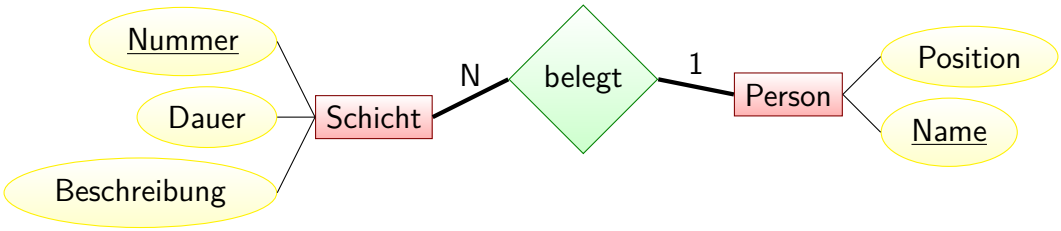
```
public class Main {
    public static void main(String[] args){
        Lift lift = new Lift();
        int[] sum = new int[4];

        for (int i = 0; i<1000; ++i){
            ++sum[lift.getPos()];
            lift.move();
        }
        for (int i = 0; i<4;++i){
            System.out.println("floor " + i + ", stops = " + sum[i]);
        }
    }
}
```

```
// Lift: simulation of a lift
// A lift object stores its current position and knows about the
// probabilistic movement model described in the task.
class Lift{
    // lift movement probabilities
    final int px[][] = {
        {0,20,30,50},
        {50,0,20,30},
        {50,20,0,30},
        {60,20,20,0}
    };

        // current position (floor) of the lift
        int pos=0;

    // simulate one trip of the lift, taking into account its current position
    void move(){
        double u = Math.random();

            u *= 100;
            int dest = 0;
            while (u > px[pos][dest]){
                u -= px[pos][dest];
                dest++;
            }
            pos = dest;

    }

    // return current position of the lift
    int getPos(){

            return pos;

    }
}
```

Aufgabe 7: Datenbanken (8P)

Neben dem Studium arbeiten Sie in einem Coffee-Shop. Die Arbeitsschichten und ihre Belegungen durch die Mitarbeiter sind in einer Datenbank gespeichert. Die Arbeitsschichten sind nummeriert und weisen eine Dauer (in Stunden) sowie eine Beschreibung (z.B. 'Morgenschicht') auf. Das ER-Diagramm dieser Datenbank sieht folgendermassen aus:

During your studies you are working in a coffee shop. The working shifts and their assignments to the employees are stored in a database. The working shifts are enumerated and consist of a duration (in hours) and a description (e.g. 'Morningshift'). The ER-Model of that database looks like this:



/2P (a) Geben Sie unten das Relationale Modell zu obigem ER-Diagramm an. Sie können entweder die formale Notation wählen oder Tabellen zeichnen. Fassen Sie ggfs. korrekt zusammen, wie in der Vorlesung gezeigt.

Provide the relational model to the ER-Diagram above. You can either use a formal notation or draw tables. Combine tables correctly as presented in the lectures.

```
Person:{Name, Position}
Schicht:{Nummer, Dauer, Beschreibung}
Belegt:{Name, Nummer}

Schicht und Belegt zusammenfassen ergibt

Person:{Name, Position}
Schicht: {Nummer, Dauer, Beschreibung, belegtVon}
```

(b) Formulieren Sie folgende Anfrage in SQL: Geben Sie die Namen und Positionen aller Mitarbeiter aus die in einer Schicht arbeiten, die weniger als 6 Stunden dauert.

Formulate the following query in SQL: Return the names and positions of all employees which work in a shift which takes less than 6 hours.

/2P

```
SELECT p.Name, p.Position
FROM Person p, Schicht s
WHERE s.belegtVon = p.Name
AND s.Dauer < 6;
```

/4P (c) Als nächstes haben Sie eine Datenbank eines sozialen Netzwerks. Die Datenbank besteht aus einer Benutzer, Foto und 'Gefällt' Tabelle. Jedes Foto hat eine Anmerkung die das Objekt auf dem Foto beschreibt sowie eine UserID, die auf den Benutzer verweist, der dieses Foto veröffentlicht hat. Die 'Gefällt' Tabelle stellt die Beziehung dar, ob einem bestimmten Benutzer (UserID) ein bestimmtes Foto (FotoID) gefällt.

Next you have a database of a social network. The database consists of a User, Photos, and 'Likes' table. Each photo has an annotation (Anmerkung) which describes the object on the photo as well as a UserID which identifies to user who published this photo. The 'Likes' table represents the relation which shows if a specific user (UserID) likes a specific photo (FotoID).

***Benutzer** (Users)*

UserID	Name	Alter
1	Johannes	22
2	Annabelle	16
3	Peter	17
4	Heidi	19
5	Felix	34
6	Andrea	22

***Fotos** (Photos)*

FotoID	Anmerkung	UserID
1	Blume	3
2	See	5
3	Baum	3
4	Biene	1
5	Schiff	4
6	Fahrrad	2
7	Baum	1

***Gefällt** (Likes)*

UserID	FotoID
1	2
2	1
4	2
3	4
3	3
5	1
2	2

Gegeben sind die vier Abfragen (a)-(d). Schreiben Sie die entsprechenden Buchstaben zu den nachfolgenden Antworten. Es gibt Antwortmöglichkeiten ohne entsprechende Abfrage.

Given the SQL queries (a)-(d), write the corresponding characters next to the results below. Some results do not have a corresponding query.

a)
```
SELECT b.Name, b.Alter
  FROM Benutzer s
  WHERE b.Alter < 18
  ORDER BY b.Alter DESC
```

b)
```
SELECT b.Name
  FROM Benutzer b, Fotos f
  WHERE b.UserID = f.UserID
  AND (f.Anmerkung = 'Baum'
  OR f.Anmerkung = 'Blume')
```

c)
```
SELECT f.Anmerkung, count(*)
  FROM Fotos f, Likes L
  WHERE f.FotoID = L.FotoID
  AND f.UserID = 3
  GROUP BY f.FotoID
```

d)
```
SELECT b.Name
  FROM Benutzer b
  WHERE b.UserID not in (SELECT L.UserID FROM Likes L)
```

	Johannes, Peter	*Johannes, Peter*
D	Andrea	*Andrea*
C	Blume; 2, Baum; 1	*Blume; 2, Baum; 1*
	Peter, Annabelle	*Peter, Annabelle*
B	Peter, Johannes, Peter	*Peter, Johannes, Peter*
	Felix	*Felix*
A	Peter; 17, Annabelle; 16	*Peter; 17, Annabelle; 16*
	Peter, Johannes	*Peter, Johannes*
	Baum; 1, Biene; 1	*Baum; 1, Biene; 1*
	Heidi	*Heidi*