Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Prüfung

Informatik II (D-BAUG)

Felix Friedrich

ETH Zürich, 29.1.2018.

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:	18	10	11	20	20	20	21	120
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 (18P)

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 alphabetical order and for numbers we take the ascending order according to their sizes.

/1P (a) Gegeben sind Daten der Länge n. Wie viele Vergleiche sind minimal nötig, um das Maximum des Datensatzes zu finden?

Given data with length n. How many comparisons are minimally required to determine the maximum of the data set?

Minimale Anzahl Vergleiche/*Minimal number of comparisons*:

/3P (b) Nachfolgend sehen Sie drei Folgen von Momentaufnahmen (Schritten) der Algorithmen (a) Sortieren durch Einfügen, (b) Sortieren durch Auswahl und (c) Bubblesort. Geben Sie unter den Folgen jeweils den Namen des zugehörigen Algorithmus an.

Consider the following three sequences of snap-shots (steps) of the algorithms (a) Insertion Sort, (b) Selection Sort and (c) Bubblesort. Below each sequence provide the corresponding algorithm name.

5	4	1	3	2
1	4	5	3	2
1	2	5	3	4
1	2	3	5	4
1	2	3	4	5

5	4	1	3	2
4	1	3	2	5
1	3	2	4	5
1	2	3	4	5

5	4	1	3	2
4	5	1	3	2
1	4	5	3	2
1	3	4	5	2
1	2	3	4	5

(c) Im folgenden ist eine Hashtabelle dargestellt (nachdem die Schlüssel 7, 9, 10 und 19 bereits eingefügt wurden). Die Hashfunktion ist $h(k) = k \bmod 11$. Kollisionen werden mit linearem Sondieren aufgelöst. Die Sondierung läuft immer nach links. Fügen Sie die folgenden Schlüssel in die (teilweise schon belegte) Hashtabelle in. Wie viele Kollisionen treten dabei (beim Einfügen der folgenden drei Zahlen) auf?

In the following a hash table is displayed (after keys 7, 9, 10 and 19 had been inserted previously). The hash function is $h(k) = k \bmod 11$. Collisions are resolved using linear probing. Probing always goes left. Insert the following keys into the (partially occupied) hash table. In doing so (in inserting the following three keys), how many collisions take place?

/4P

Einzufügende Schlüssel/*Keys to be inserted*: 20, 21, 22

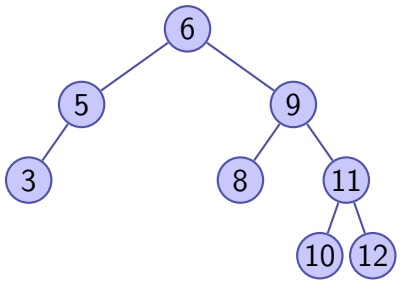
							7	19	9	10
0	1	2	3	4	5	6	7	8	9	10

Anzahl Kollisionen/*Number of collisions*: ◯ 6 ◯ 8 ◯ 10 ◯ 12 ◯ 14

(d) Fügen Sie in untenstehendem binären Suchbaum zuerst den Schlüssel 4 ein und löschen Sie danach im entstandenen Suchbaum den Schlüssel 6.

First insert into the Binary Search Tree below the key 4. After insertion, delete the key 6 from the created binary search tree.

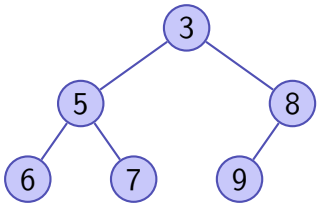
/4P



Nach Einfügen von 4 / *After insertion of 4*	Nach Löschen von 6 / *After deletion of 6*

/4P (e) Fügen Sie in untenstehendem Min-Heap zuerst den Schlüssel 4 ein und löschen Sie danach im entstandenen Min-Heap das Minimum.

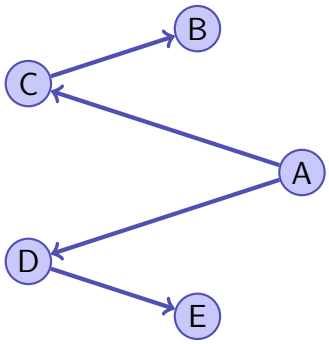
First insert into the Min-Heap below the key 4. After insertion, delete the Minimum from the created Min-Heap.



Nach Einfügen von 4 / *After insertion of* 4	Nach Löschen des Minimums / *After deletion of the Minimum*

/2P (f) Geben Sie zu folgenden gerichteten Graphen G mit Knotenmenge $\{A, B, C, D, E\}$ eine mögliche Besuchsreihenfolge der Knoten bei Breiten- und bei Tiefensuche an. Die Suche startet jeweils am Knoten A.

Provide to the following directed graph G with node set $\{A, B, C, D, E\}$ a possible visiting order when a breadth first search and a depth first search is conducted. The search starts in node A.



Breitensuche/*Breadth first search*	
Tiefensuche/*Depth first search*	

Aufgabe 2: Code Schreiben (10P)

/10P

Vervollständigen Sie folgende Funktion `maxTuple` so, dass Sie die maximale Anzahl gleicher Werte im Array a zurückgibt.
Beispiel: Das Array $[11, 200, 33, 200, 200, 33]$ enthält die Zahl 200 drei Mal, die Zahl 33 zwei Mal und die Zahl 11 ein Mal, also ist die maximale Anzahl gleicher Werte 3.
Es kommt nicht auf die Performanz Ihres Algorithmus an.

Complement the following function `maxTuple` *such that it returns the maximal number of same values in the array* a.
Example: $[11, 200, 33, 200, 200, 33]$ *contains* 200 *three times,* 33 *twice and* 11 *once. Therefore the maximal number of same values is 3.*
The performance of your algorithm is of no importance.

```
// pre: a != null
// post: return max {|{0 <= i < n: a[i] = a[j]}|: 0 <= j < n }
public static int maxTuple(int a[]){
    int max = 0;
    for (int i = 0; i<a.length; ++i){
        
    }
    return max;
}
```

Aufgabe 3: Asymptotik (11P)

/5P (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 richtigen Reihenfolge, da $n^3 \in \mathcal{O}(n^5)$ und $n^5 \in \mathcal{O}(n^7)$.

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

$$n^2 + 2n + 1, \quad n \sum_{i=0}^{n} i, \quad n \log n^n, \quad \sqrt{n} \log n, \quad n \log n^2, \quad n!$$

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 möglichst knapp 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 $f()$ depending on $n \in \mathbb{N}$ using Θ notation as tight as possible. The function f does not call itself. You do not have to justify your answers.

(b) /1P

```
void g(int n){
    for (int i = 0; i<n ; ++i ){
        if (i < 10){
            f();
        }
    }
}
```

Anzahl Aufrufe von f / *Number of calls of f*

(c) /2P

```
void g(int n){
    for (int i = 0; i<n ; ++i ){
        for (int j=0; j<i; ++j){
            f()
        }
    }
}
```

Anzahl Aufrufe von f / *Number of calls of f*

(d) /3P

```
void g(int n){
    if (n > 1){
        g(n/2);
    }
    f();
}
```

Anzahl Aufrufe von f / *Number of calls of f*

/20P

Aufgabe 4: Verkettete Listen (20P)

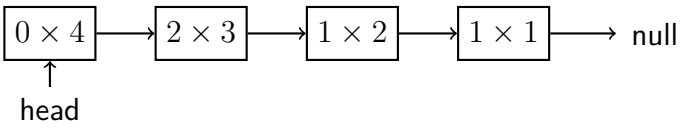
Eine Multi-Menge ist eine Menge, bei der jedes Element mehrfach vorkommen kann. Die Reihenfolge der Elemente in einer Menge ist irrelevant.
In der dargestellten Implementation wird diese als verkettete Liste gespeichert, bei der jedes Element zusätzlich die Häufigkeit seines Auftretens in der Menge speichert. Nachfolgend ist die Datenstruktur eines einzelnen Elementes der Menge dargestellt.

A multi-set is a set that can hold elements multiple times. The order of the elements in a set is irrelevant.
In the following implementation a multiset is stored as linked list where each element additionally contains the number of occurences in the set. Depicted in the following is the data structure corresponding to a single element of the set.

```
class ElementNode{
    ElementNode next;
    int value;
    int quantity;

    ElementNode (int v, ElementNode n){
        value = v;
        next = n;
        quantity = 1;
    }
}
```

Nachfolgend dargestellt ist der Inhalt der Multimenge $\{3, 3, 2, 1\}$, welche folgendermassen entstanden ist: Einer initial leeren Liste wurden die Elemente $1, 2, 3, 2, 3, 4$ hinzugefügt und nachfolgend wurden die Elemente 2 und 4 je einmal entfernt.
Zu erkennen ist auch, dass beim Entfernen eines Elementes aus der Multimenge dieses nicht aus der Liste genommen werden muss, sondern dass das Herunterzählen des Zählers in unserer Implementation genügt. Natürlich darf der Zähler nicht negativ werden.

Presented in the following is the content of the multi-set $\{3, 3, 2, 1\}$, which has been generated in the following way: to an initial empty list have the elements $1, 2, 3, 2, 3, 4$ been added, and subsequently the elements 2 and 4 have been removed once each.
It becomes apparent that when an element is removed from the multi-set, it does not have to be removed from the list but the counter only has to be decreased. The counter must not become negative, of course.



Ergänzen Sie den Code auf der rechten Seite, so dass die Datenstruktur `MultiSet` erwartungs- und dokumentationsgemäss funktioniert.

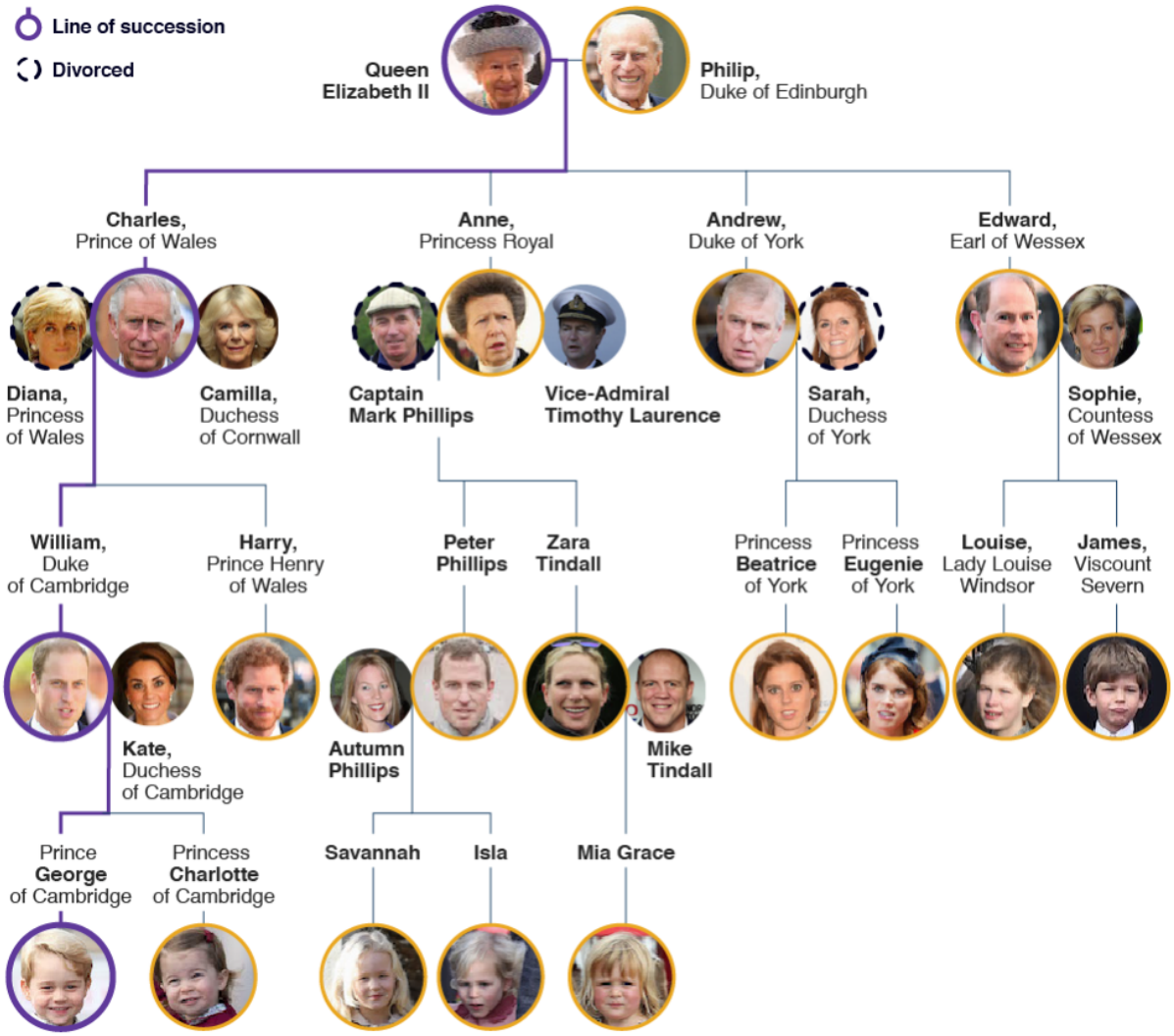
Complement the code on the right hand side such that the data struture `MultiSet` would work as expected and documented.

```
class MultiSet{
    ElementNode head=null;

    // Insert value into the multi-set. If the corresponding element exists
    // then increase its quantity, otherwise add a new element somewhere
    // to the list.
    public void insert(int value){
        ElementNode e = head;
        while (e != null && e.value != value){
            e = e.next;
        }
        if (e == null){
            
        } else {
            ++e.quantity;
        }
    }
    // Remove value from the multi-set.
    // If the element is contained in the set, then decrease its quantity.
    // Elements are at not physically removed from the list.
    public void remove(int value){
        ElementNode e = head;
        while (e != null && e.value != value){
            e = e.next;
        }
        if (                                    ){
            e.quantity--;
        }
    }
    // Return the count of occurences of value in the multi-set
    public int count(int value){
        ElementNode e = head;
        
    }
}
```

Aufgabe 5: Stammbaum (20P)

Die Klasse `Person` auf Seite 12 repräsentiert eine Person durch ihren Namen und ein (möglicherweise leeres) Array von Referenzen auf ihre Kinder. Mit Hilfe dieser Klasse können Stammbäume wie der im Bild angegebene aufgebaut werden.

The class `Person` on Page 12 represents a person by its name and a (possibly empty) array of references to its children. With the help of this class, we can build family trees such as the one in the picture.



http://www.bbc.com/news/uk-23272491

(a) Ergänzen Sie die Methode `addChild` so, dass das neue Kind zur Person hinzugefügt wird.

Complement the method `addChild` such that the new child is added to the person given.

/10P

(b) Ergänzen Sie die Methode `numberOfChildren`, so dass sie für eine gegebene Person korrekt berechnet, wieviele Kinder vom Grad g sie hat. Dabei sind die Kinder vom Grad 0 die eigenen Kinder, die Kinder vom Grad 1 die Enkel, die Kinder vom Grad 2 die Urenkel usw. Im Beispiel hat Elizabeth 4 Kinder vom Grad 0, 8 vom Grad 1, 5 vom Grad 2 und (noch) keine von grösserem Grad.

Complete method `numberOfChildren`, such that for a given person, it correctly computes the number of children of degree g. Children of degree 0 are the own children, children of degree 1 are the grand children, children of degree 2 are the grand grand children etc. In the example, Elizabeth has 4 children of degree 0, 8 of degree 1, 5 of degree 2, and none of higher degree (yet).

/10P

```
class Person{
    String name;
    Person[] children; // invariant: can never be null,
                       // children.length = number of children of degree 0

    Person(String name){
        this.name = name;
        children = new Person[0]; // no children, to start with
    }

    void addChild(Person newKid){
        Person[] newKids =
        children = newKids;
    }

    // returns the number of children of degree g of *this
    //   g = 0: children
    //   g = 1: grand children, g = 2: grand grand children ...
    int numberOfChildren (int g)
    {
        if (g == 0) {
        } else {
            int c = 0;
            return c;
        }
    }
};
```

Der nachfolgende Code zeigt, wie die Klasse Person benutzt werden soll.

The following code shows how the class Person is supposed to be used.

```
public class Main {

    public static void main(String[] args){
        Person elizabeth = new Person("Elizabeth");
        Person charles = new Person("Charles");
        elizabeth.AddChild(charles);
        Person anne = new Person("Anne");
        elizabeth.AddChild(anne);
        Person andrew = new Person("Andrew");
        elizabeth.AddChild(andrew);
        Person edward = new Person("Edward");
        elizabeth.AddChild(edward);
        Person william = new Person("William");
        charles.AddChild(william);
        william.AddChild(new Person("George"));
        william.AddChild(new Person("Charlotte"));
        Person harry = new Person("Harry");
        charles.AddChild(harry);
        Person peter = new Person("Peter");
        anne.AddChild(peter);
        peter.AddChild(new Person("Savannah"));
        peter.AddChild(new Person("Isla"));
        Person zara = new Person("Zara");
        anne.AddChild(zara);
        zara.AddChild(new Person("Mia"));
        andrew.AddChild(new Person("Beatrice"));
        andrew.AddChild(new Person("Eugenie"));
        edward.AddChild(new Person("Louise"));
        edward.AddChild(new Person("James"));

        for (int g = 0; g<4; ++g){
            System.out.println("children of degree " + g + ": "
                               + elizabeth.numberOfChildren(g));
        }
        // output:
        // children of degree 0: 4
        // children of degree 1: 8
        // children of degree 2: 5
        // children of degree 3: 0
    }
}
```

/20P

Aufgabe 6: Besuchshäufigkeit (20P)

Die Besitzer einiger Bars haben sich zusammengeschlossen und legen in ihrer Bar jeweils Werbung der anderen Bars aus. Sie erwarten, dass die Besuchsrate von Nachtschwärmern in Ihren Läden dadurch steigt. Leider jedoch können sich nicht alle Barbesitzer gleich gut leiden und daher legt nicht jeder von jedem die Werbung aus.
Für wen lohnt sich das? Wir modellieren einen hypotetischen Barbesucher. Mit ganzen Zahlen bewerten wir, ob der Barbesucher zu jeder vollen Stunde in der jeweiligen Bar bleibt (5) oder zu einer anderen Bar wechselt. Ein Kunde wechselt eher zu einer Bar, welche ihm per Werbung empfohlen wurde (3). Es kann aber auch sein, dass der Kunde einfach so eine beliebige andere Bar aufsucht (1). Nachfolgend sehen Sie die zugehörige Beliebtheitsmatrix. In Zeile r und Spalte c finden Sie die Beliebtheit von Bar r zu Bar c zu wechseln.

The owners of some bars have allied and display advertisement of the other bars in their bars. They expect that the visiting rate of guests will increase. Unfortunately not all bar owners like each other and therefore not everybody is willing to advertise all of the others.
Who will profit? We model a hypothetical guest. With integer numbers we rate if a guest, at each full hour, rather stays in the bar (5) or rather changes to a different bar. A guest will rather change to a bar that had been advertised (3) than any other, non-advertised bar (1). Displayed below there is a popularity matrix. In row r and column c you find the popularity to change from bar r to bar c.

Bar / *Bar*	0	1	2	3
0 (Brews Brothers)	5	1	1	1
1 (The Tipsy Cow)	1	5	3	3
2 (Wunderbar)	1	3	5	1
3 (Baracke)	1	1	3	5

Als findige IngeneurIn skalieren Sie die Zeilen dieser Tabelle nun so, dass sich eine Wahrscheinlichkeitsmatrix ergibt. Sie simulieren das Verhalten des Barbesuchers, wie in der Vorlesung gesehen, als Markov-Kette. Komplementieren Sie den Code so dass die Main-Funktion die Aufenthaltshäufigkeit in der jeweiligen Bar nach 10^6 Iterationen ausgibt.

As innovative engineer you scale the rows of the table such that it becomes a probability matrix. You simulate the behavior of the guest, as seen in the lectures, as a Markov chain. Complement the code such that the main method outputs the frequency of visits to each bar after 10^6 iterations.

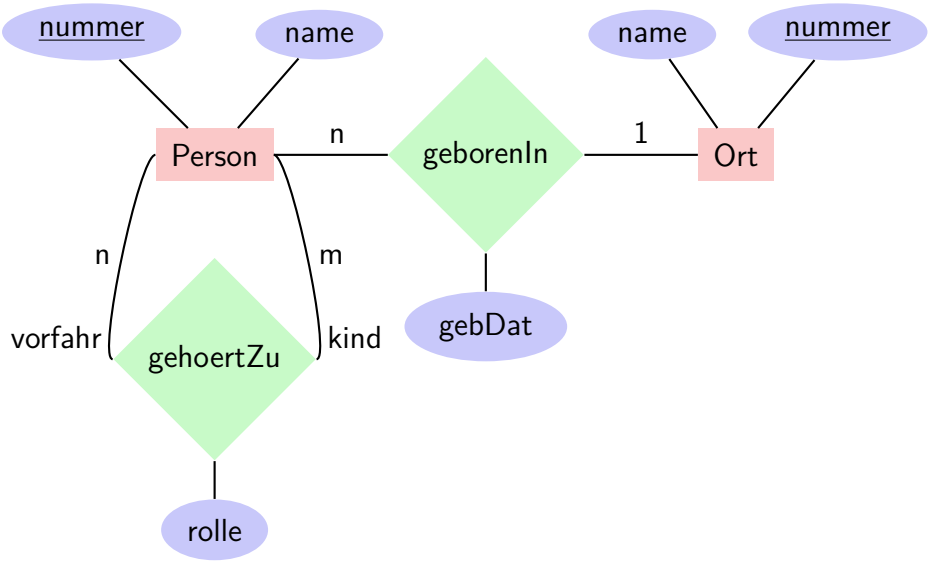
```
class Sampler{
    final float P[][];
    // pre: matrix prob where entries of each row sum up to 1
    Sampler(float prob[][]){
        P = prob;
    }
    // sample from row r of the probability matrix P
    int Sample(int r){
```

```
        double u = Math.random();
        int res = 0;
        while (u > P[r][res]){
            
        }
        return res;
    }
}

public class Main {
    // pre: non-null matrix p with #rows = #columns
    // linearly scale the rows of a popularity matrix p such that the
    // resulting matrix forms a probability matrix
    public static float[][] ProbabilityMatrix(int[][] p){
        float[][] res = new float[p.length][p.length];
        for (int r = 0; r<p.length; ++r){
            
        }
        return res;
    }

    public static void main(String[] args){
        int popularity[][] = {{5,1,1,1},{1,5,3,3},{1,3,5,1},{1,1,3,5}};
        Sampler s = new Sampler(ProbabilityMatrix(popularity));
        int visits[] = new int[popularity.length];
        int current = 0;
        for (int i = 0; i<1000000; ++i){
            
            current = s.Sample(current);
        }
        for (int i = 0; i<popularity.length; ++i) {
            System.out.println(i + ": " + visits[i]);
        }
    }
}
```

Aufgabe 7: Datenbanken (21P)

Sie wollen (ähnlich wie in Aufgabe 5) einen Stammbaum erstellen. Diesmal entscheiden Sie sich, das mit einer Datenbank zu machen. Sie wollen aber nicht nur das klassische Modell Vater, Mutter, Kinder modellieren sondern erlauben, dass ein Kind mehrere Väter oder Mütter hat und dass diese Eltern verschiedene Rollen einnehmen (z.B. Mutter, Vater, Stiefvater). Sie modellieren das ganze mit folgendem ER-Modell.

You want to (similar to task 5) create a family tree. This time you decide to do this with a database. You do not want to model the classical model father, mother, childern but want to allow that a parent has several mothers or fathers and that they have different roles (e.g. father, mother, stepfather). Your ER-Model looks like the following.



/3P (a) Geben Sie unten das Relationale Modell zu obigem ER-Diagramm an. Sie können entweder die formale Notation wählen oder Tabellen zeichnen. Fassen Sie die Tabellen 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.

(b) Formulieren Sie folgende Anfrage in SQL: Geben Sie die Namen aller Kinder von `Peter` aus. Wir gehen davon aus, dass die Datenbank nur eine Person mit Namen Peter enthält.

Formulate the following query in SQL: output the names of all children of `Peter`. Wie assume that the data base contains only one person named Peter.

/4P

(c) Formulieren Sie eine Anfrage in SQL, die alle Paare von Geschwistern auflistet. Jedes Geschwisterpaar sollte nur einmal aufgelistet werden.
Beispiel: wenn Peter der Vater von Alex, Florian und Hermann ist, sollte die Liste enthalten: Alex,Florian ; Alex,Hermann ; Florian,Hermann .

Formulate a query in SQL to output all pairs of siblings (brothers and sisters). Each pair of siblings should be listed only once.
Example: if Peter is the father of Alex, Florian and Hermann, then the result list should contain : Alex,Florian ; Alex,Hermann ; Florian,Hermann.

/5P

/9P (d) Als nächstes haben Sie eine Datenbank (minimaler Ausschnitt) vom öffentlichen Nahverkehr der Stadt Zurich: Linien bedienen Haltestellen.

Next you have a database of a the Zurich public transport (minimal excerpt). Lines serve halts.

***linie** (line)*

id	**typ**
6	Tram
9	Tram
10	Tram
24	Polybahn

***bedient** (serves)*

linie	**halt**
6	1
6	3
6	4
6	6
9	1
9	2
10	1
10	3
10	6
24	5
24	6

***halt** (halt)*

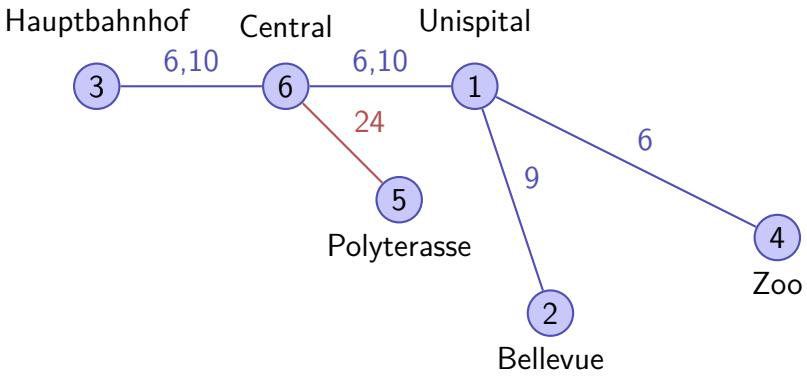
id	**name**
1	Unispital
2	Bellevue
3	Hauptbahnhof
4	Zoo
5	Polyterasse
6	Central

Gegeben sind die vier Anfragen A - C. Schreiben Sie die entsprechenden Buchstaben zu den nachfolgenden Antworten. Ein Semikolon in den Antworten steht für eine neue Zeile. Ein Komma steht für eine neue Spalte. Es gibt Antwortmöglichkeiten ohne entsprechende Abfrage.

Given the SQL queries A - C, write the corresponding characters next to the results below. A semicolon in the answers represents a new line. A comma represents a new column. Some results do not have a corresponding query.

Tipp: versuchen Sie die Anfragen zu verstehen und mit der gewonnenen Information die Anfrage selbst zu beantworten. Obige Tabellen modellieren die folgende Situation.

***Hint:** try to understand the queries before answering the query yourselves. The tables above model the following situation.*



[A]

```
SELECT DISTINCT h1.id, h2.id
FROM  halt h1, halt h2, bedient b1, bedient b2
WHERE b1.linie = b2.linie
      AND h1.halt = b1.halt AND h2.halt = b2.halt
      AND h1.id < h2.id
```

[B]

```
SELECT h.name
FROM halt h, bedient b, linie l
WHERE l.id = b.linie AND b.halt = h.id
      AND l.typ LIKE 'Polybahn'
```

[C]

```
SELECT h.name, count(b.linie)
FROM halt h, bedient b
WHERE h.id = b.halt
GROUP BY h.id
```

☐ 1,3 ; 1,4 ; 1,6 ; 3,4 ; 3,6 ; 4,6 ; 1,2 ; 3,6 ; 5,6

☐ 1,2 ; 1,3 ; 1,4 ; 1,5 ; 2,3 ; 2,4 ; 2,5 ; 3,6

☐ Polyterasse ; Unispital

☐ Unispital,3 ; Bellevue,1 ; Hauptbahnhof,2 ; Zoo,1 ; Polyterasse,1 ; Central,3

☐ Polyterasse ; Central

☐ 2,3 ; 2,4 ; 2,6 ; 3,4 ; 3,5

☐ Zoo,4 ; Polyterasse,5 ; Central,6