Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Prüfung

Informatik II (D-BAUG)

Felix Friedrich, Hermann Lehner, Departement Informatik

ETH Zürich, 10. August 2020.

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: 90 Minuten.
2. Erlaubte Unterlagen: Wörterbuch (für von Ihnen gesprochene Sprachen). 4 A4 Blätter ohne inhaltliche oder formale Einschränkung.
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:

Exam duration: 90 minutes.

Permitted examination aids: dictionary (for languages spoken by yourself). 4 sheet of A4 paper, no formal or content-wise restrictions.

Use a pen (black or blue), not a pencil. Please write legibly. We will only consider solutions that we can read.

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!

There are no negative points for wrong answers.

If you feel disturbed by anyone or anything, let the supervisor of the exam know immediately.

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.

If you need to go to the toilet, raise your hand and wait for a supervisor.

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	Total
Points:	17	9	9	10	15	60
Score:						

Generelle Anmerkung / *General Remark*

Verwenden Sie die Notation, Algorithmen und Datenstrukturen aus der Vorlesung. Falls Sie eine andere Herangehensweise wählen, erklären Sie Ihre Antworten in nachvollziehbarer Weise!

Use notation, algorithms and data structures from the course. If you use a different approach, explain your answers in a comprehensible way!

Aufgabe 1: Verschiedenes (17P)

In dieser Aufgabe sollen nur Ergebnisse angegeben werden. Begründungen sind nicht notwendig.

In this task only results have to be provided. Explanations are not required.

/7P (a) Kreuzen Sie an, ob die folgenden Aussagen wahr oder falsch sind. *Mark if the following statements are true or false.*

Während die O-Notation die asymptotische Laufzeit eines Algorithmus im schlechtesten Fall charakterisiert, verwendet man die Θ-Notation, um gleichzeitig auch die Laufzeit im besten Fall zu charakterisieren. / *While the O-Notation is used to characterize the asymptotic runtime of an algorithm in the worst case, the Θ-Notation is used in order to characterize the best case runtime as well.*

◯Wahr / *True*
◯Falsch / *False*

Mergesort ist ein Divide-and-Conquer-Algorithmus (teile und herrsche). / *Mergesort is a divide-and-conquer algorithm.*

◯Wahr / *True*
◯Falsch / *False*

Die asymptotische Laufzeit des Einfügens im schlechtesten Fall in ein sortiertes Array ist $O(\log n)$ / *The asymptotic worst-case running time of inserting a value into a sorted array is $O(\log n)$.*

◯Wahr / *True*
◯Falsch / *False*

Die asymptotische Laufzeit der Suche in einem beliebigen binären Suchbaum ist im schlechtesten Fall $O(\log n)$ / *The asymptotic worst-case running time of a search in an arbitrary binary search tree is $O(\log n)$*

◯Wahr / *True*
◯Falsch / *False*

Die Angabe der Werte der Elemente eines binären Suchbaumes in Hauptreihenfolge bestimmt den dazugehörigen Baum eindeutig. / *The specification of the values of elements of a binary search tree in preorder uniquely determines the corresponding tree.*

◯Wahr / *True*
◯Falsch / *False*

Wenn man in einem positiv gewichteten Graphen das Gewicht jeder Kante um eins erhöht, bleibt jeder kürzester Weg zwischen zwei Knoten ein kürzester Weg. / *If in a positively weighted graph the weight of each edge is increased by one, every shortest path between two points remains a shortest path.*

◯Wahr / *True*

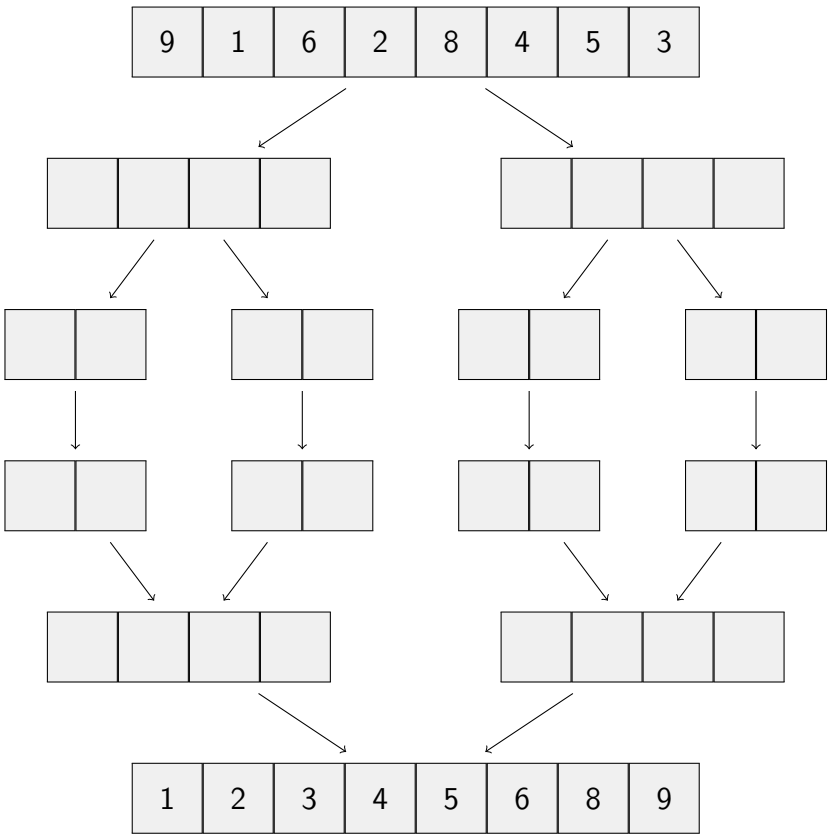
◯Falsch / *False*

Die erlaubte Balance jedes Knotens eines AVL Baumes ist ein Wert zwischen -1 und 1. Daher darf sich die Höhe eines linken Teilbaumes von der Höhe des rechten Teilbaumes um 2 unterscheiden. / *The permitted balance of each node of an AVL tree is a value between -1 and 1. Therefore the height of a left subtree may differ from the height of the right subtree by 2.*

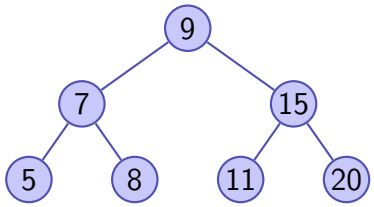
◯Wahr / *True*

◯Falsch / *False*

(b) Führen Sie auf dem folgenden Array den Mergesort-Algorithmus schematisch durch. *On the following array, perform the mergesort-algorithm using the scheme given.* /1P



/2P (c) Gegeben sei der folgende Suchbaum *Let the following search tree be given*



Geben Sie nachfolgend die Werte der Knoten der verlangten Besuchsreihenfolge entsprechend aus.

Provide in the following the values of the nodes according to the required visiting order.

Hauptreihenfolge / *preorder*

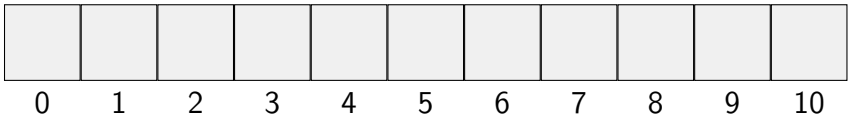
Nebenreihenfolge / *postorder*

/2P (d) Fügen Sie die folgenden Schlüssel (in der angegebenen Reihenfolge) in die Hashtabelle ein. Verwenden Sie offene Addressierung und doppeltes Hashing. Die verwendeten Hash-Funktionen $h(k)$ und $h'(k)$ sind nachfolgend angegeben (es wird addiert, also nach rechts sondiert).

Enter the following keys (in the order provided) into the hash-table. Use open addressing and double hashing. The used hash functions $h(k)$ and $h'(k)$ are provided below (values are added, i.e. probing runs to the right).

Hashfunktionen / *hash functions* : $h(k) = k \bmod 11, h'(k) = 1 + k \bmod 9$

Schlüssel / *keys* : 2,13,24,39



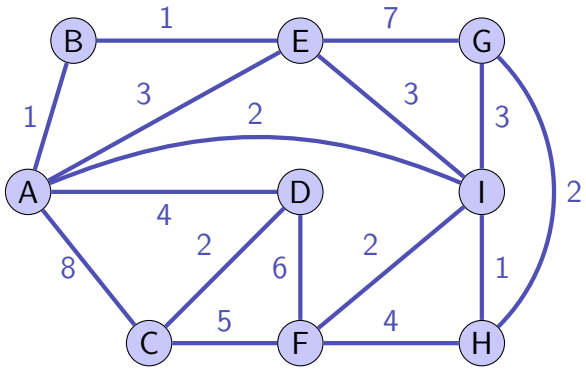
Wie viele Kollisionen treten bei der nachfolgenden (erfolglosen) Suche nach dem Schlüssel 101 auf? (Gezählt werden nur Kollisionen mit belegten Plätzen)

How many collisions occur when key 101 is searched now (unsuccessfully)? (We only count collisions with occupied spaces)

(e) Markieren Sie in folgendem ungerichteten, gewichteten Graphen die Kanten eines minimalen Spannbaumes.

In the following undirected weighted graph, mark the edges of a minimum spanning tree.

/1P



(f) In der folgenden Tabelle ist ein Min-Heap in seiner üblichen Form gespeichert. Wie sieht die Tabelle aus, wenn das kleinste Element entfernt und die Heapbedingung wieder hergestellt wurde?

The following table comprises a Min-Heap in its canonical form. What does the table look like when the smallest element has been removed and the heap-condition has been re-established?

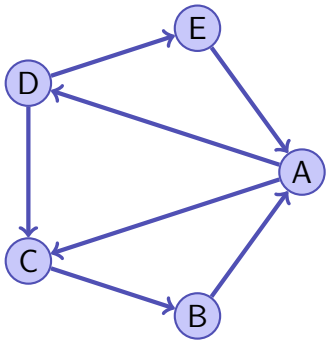
/2P

1	3	5	12	6	8	11	14	13	17	42	16	15
1	2	3	4	5	6	7	8	9	10	11	12	13

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

/2P (g) Geben Sie zu folgendem 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. Wenn mehrere Möglichkeiten vorhanden sind, wird immer der alphabetisch kleinere Knoten zuerst genommen.

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. If there are more than one possible ways to choose the next node, always the alphabetically smaller element is taken first.



Breitensuche/*Breadth first search*

Tiefensuche/*Depth first search*

Aufgabe 2: Asymptotik (9P)

/3P (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 $n^5 \in \mathcal{O}(n^7)$.

$$n^2 \log n \quad n \log n^2 \quad \sqrt{\log n} \quad \log \sqrt{n} \quad \sum_{i=0}^{n} n - i \quad n^3 - n^{-3}$$

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 $g(n)$. Fill in the asymptotic number of calls of $f()$ depending on $n \in \mathbb{N}$ using Θ notation as succinct as possible. The function f does not call itself. You do not have to justify your answers.

(b) /2P

```
# pre: n is an integer
def g(n):
    m = 0;
    while m*m < n:
        f()
        m = m + 1
```

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

(c) /2P

```
# pre: n is an integer
def g(n):
    for i in range(0,n*n):
        for j in range(0,i):
            f()
```

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

(d) /2P

```
# pre: n is an integer and power of 2
def g(n):
    if n>1:
        g(n/2)
    f();
```

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

Aufgabe 3: Maximaler Fluss (9P)

Sie müssen das Abendessen für $n > 0$ **Personen** P_i **(**$i = 1, \ldots, n$**)** organisieren. Sie haben $k > 5$ **Restaurants** R_j **(**$j = 1, \ldots, k$**)** zur Auswahl. Jeder der n Personen gibt eine **Favoritenliste** von 5 Restaurants an. Die Restaurants haben jeweils $f_j \in \mathbb{N}$ **freie Plätze (**$j = 1, \ldots, k$**)**
Sie sollen die Personen nun auf die Restaurants verteilen, so dass jede Person in einem der angegebenen Restaurants sitzt und natürlich auch so, dass kein Restaurant überbucht ist.

You have to organize dinner for $n > 0$ **people** P_i **(**$i = 1, \ldots, n$**).** *You have a choice of $k > 5$* **restaurants** R_j **(**$j = 1, \ldots, k$**).** *Each of the n people provides you with a list of 5* **favourite restaurants**. *The restaurants have $f_j \in \mathbb{N}$* **available seats (**$j = 1, \ldots, k$**)** *each*
You need to distribute the people on the restaurants such that each person is assigned to one of the favourite restaurants and so that no restaurant is overbooked.

/4P (a) Sie müssen entscheiden, ob die Verteilung der Leute auf die Restaurants überhaupt möglich ist. Modelliern Sie die Frage als Flussproblem. Erläutern / skizzieren Sie ein geeignetes Flussnetzwerk $G = (V, E, c)$. Geben Sie Knoten, Kanten und Kapazitäten an und erläutern Sie, wie Sie aus dem maximalen Wert eines Flusses schlussfolgern können, ob eine geeignete Verteilung existiert oder nicht.

You have to decide if an acceptable assignment of the n people to the restaurants is possible at all. You model that question as a network flow problem. Described / sketch a suitable flow network $G = (V, E, c)$. Specify nodes, edges and capacities of the network and describe how you can deduce from the value of the flow if a suitable distribution exists or not.

(b) Nennen Sie einen Algorithmus, der das Problem aus (a) möglichst effizient löst. Geben Sie die beste Abschätzung der Laufzeit des Algorithmus in Abhängigkeit von n und k an.

Provide the name of an algorithm that solves the problem from (a) as efficient as possible. Provide the best characterization of the running time of the algorithm as a function of n and k.

/3P

(c) Angenommen, Sie haben das Problem aus (a) schon gelöst und kennen damit den maximalen Fluss und und der dazugehörige Graph ist verfügbar. Nun wollen Sie eine Tabelle ausgeben, in der jeder der n Personen das Restaurant zugeordnet wird. Wie machen Sie das. Geben Sie die asymptotische Laufzeit des Algorithmus an.

Provided you have already solved the problem from (a) and you know the maximum flow and the corresponding graph is available. Now you want to output a table that maps each of the n people to a restaurant. How can you do this? Provide the asymptotic running time of the algorithm.

/2P

/10P

Aufgabe 4: Python / Memoization (10P)

Diese Aufgabe zum Thema Memoization und Programmieren in Python soll am Computer gelöst werden. Sie können sich die Zeit frei einteilen. Wir **empfehlen** aber, dass Sie **nicht mehr als 15 Minuten** für diese Aufgabe aufwenden.

Lösen Sie diese Aufgabe in der Online-Umgebung (Code Expert via Moodle).

This task on memoization and programming with Python needs to be solved at the computer. You are free in how you divide the time. However, we **recommend to spend not more than 15 minutes** *on this problem.*

Solve this task in the online environment (Code Expert via Moodle).

/15P

Aufgabe 5: Spannbäume: UnionFind, Kruskal, Dijkstra (15P)

Diese Aufgabe zum Thema Spannbäume und kürzeste Wege soll am Computer gelöst werden. Sie können sich die Zeit frei einteilen. Wir **empfehlen** aber, dass Sie **nicht mehr als 30 Minuten** für diese Aufgabe aufwenden.

Lösen Sie diese Aufgabe in der Online-Umgebung (Code Expert via Moodle).

This task on spanning trees and shortest paths needs to be solved at the computer. You are free in how you divide the time. However, we **recommend to spend not more than 30 minutes** *on this problem.*

Solve this task in the online environment (Code Expert via Moodle).