Prof. Dr. Ir. Joost-Pieter Katoen

Sebastian Junges, Benjamin Kaminski, David Korzeniewski, Tim Quatmann

Übung 1

– Musterlösung –

Hinweise:

- Die Lösungen müssen bis **Donnerstag, den 19. April um 16:00 Uhr** in den entsprechenden Übungskasten eingeworfen werden. Sie finden die Kästen am Eingang Halifaxstr. des Informatikzentrums (Ahornstr. 55).
- Da die Zuteilung zu den Übungsgruppen erst am Tag der Abgabe bekannt gegeben wird, **muss** dieses Blatt **ausnahmsweise alleine** abgegeben werden. Ansonsten müssen die Übungsblätter in Gruppen von je 3 Studierenden aus der gleichen Kleingruppenübung bearbeitet werden.
- Drucken Sie ggf. digital angefertigte Lösungen aus. Abgaben z.B. per Email sind nicht zulässig.
- Namen und Matrikelnummer sowie die **Nummer der Übungsgruppe** sind auf jedes Blatt der Abgabe zu schreiben. Abgaben, die aus mehreren Blättern bestehen, **müssen geheftet bzw. getackert** werden! Die **Gruppennummer muss sich auf der ersten Seite oben links** befinden.
- **Bei Nichtbeachten der obigen Hinweise müssen Sie mit erheblichen Punktabzügen rechnen!**
- Folgende Kriterien müssen für die **Zulassung zur Klausur** erfüllt werden:
  1. Mindestens 50% aller in den Übungen erreichbaren Punkte *bis* zur Präsenzübung
  2. Mindestens 50% aller in den Übungen erreichbaren Punkte *ab* der Präsenzübung
  3. Mindestens 50% der in der Präsenzübung (PÜ) erreichbaren Punkte.

  CES-Studierende brauchen kein Zulassungskriterium zu erfüllen.
- Es gilt außerdem folgende Bonusregelung für alle Studiengänge: Bei $\geq 70\%$ in allen drei oben genannten Punkten wird die Klausurnote um eine Notenstufe (0.3) verbessert. Das heißt aus einer 2.3 wird beispielsweise eine 2.0 (gilt nicht für die Noten 1.0 und 5.0).

Aufgabe 1 (Anmelden): (50 Punkte)

Melden Sie sich korrekt und vollständig in unserem Übungssystem auf

`https://lufgi2.informatik.rwth-aachen.de/dsal18/`

an, aktivieren Sie Ihren Account und geben Sie Ihre Präferenzen für die Tutorien ein. Sie müssen zur Anmeldung Ihre @rwth-aachen.de Adresse benutzen.

Die Anmeldung ist bis Mittwoch, den 18. April um 12:00 Uhr möglich.

Die Zuteilung zu den Übungsgruppen wird bis spätestens Mittwoch, den 18. April, um 20:00 Uhr über das Übungssystem bekannt gegeben. Notieren Sie die Nummer Ihrer Übungsgruppe oben links auf der ersten Seite Ihrer Abgabe.

Sie bekommen nur Punkte für diese Aufgabe, wenn Sie die korrekte Übungsgruppennummer angeben!

Hinweise:

- **Lesen Sie die Hinweise im Übungssystem sorgfältig!**
- Da es für die meisten Studiengänge nur eine Variante des Bachelors gibt (z. B. gibt es für Informatik nur den Bachelor of Science, aber nicht den Bachelor of Arts), sind die Studiengänge lediglich mit „Bachelor“ bezeichnet. Analoges gilt für Masterstudiengänge.
- Studierende, welche Informatik als eines ihrer beiden Hauptfächer auf Lehramt im Bachelorstudiengang studieren, sollten den Studiengang „Bachelor Computer Science (Lehramt)“ wählen. Wer Informatik als Ergänzungsfach oder auf Lehramt im Staatsexamenstudiengang studiert, sollte „Other“ wählen und eine entsprechende Angabe machen.
- Falls Sie in einem Studiengang eingeschrieben sind, für den es einen entsprechenden Eintrag im System gibt (wie z. B. „Bachelor Informatik“), sollten Sie auch diesen existierenden Eintrag wählen. Die Wahl von „Other“ mit der Angabe eines Studiengangs, für den es einen vorgefertigten Eintrag gibt, gilt nicht als korrekt.
- Hohe Präferenzen sollten Sie für diejenigen Tutorien wählen, die Sie gerne besuchen möchten. Tutorien, die Sie bevorzugt belegen wollen, sollten Sie möglichst weit oben einordnen.
- Einige Tutorien haben eine spezielle Zielgruppe (Studierende im Studiengang CES oder Erstsemester, die ihr Studium also im Sommersemester begonnen haben). Diese Tutorien sind trotzdem offen für alle Studierenden (genauso wie die übrigen Tutorien auch offen für Studierende aus diesen speziellen Zielgruppen sind), aber dort ist die jeweilige Gruppenleitung besser auf typische Fragen aus den speziellen Zielgruppen vorbereitet.
- Sie können (müssen aber nicht) **genau einen Abgabepartner** angeben. Dieser muss umgekehrt auch Sie angeben. Damit wird garantiert, dass Sie beide in die gleiche Gruppe eingeteilt werden. Es ist nicht möglich, mehr als einen Abgabepartner anzugeben.

Aufgabe 2 (Reihen): (10+10 Punkte)

Zeigen Sie, dass die folgenden Aussagen für beliebige $n \in \mathbb{N}^{>0}$ gelten:

a) $\sum_{k=1}^{n} k = \frac{n(n+1)}{2}$ **b)** $\sum_{i=0}^{n} 2^i = 2^{n+1} - 1$

Lösung:

(a) Beweis durch vollständige Induktion:

Induktionsverankerung für $n = 1$: $\sum_{k=1}^{1} k = 1 = \frac{2}{2} = \frac{1(1+1)}{2}$ Die Aussage gilt also für $n = 1$.

Induktionsschritt von n auf $n+1$: $\sum_{k=1}^{n+1} k = \sum_{k=1}^{n} k \quad + \quad (n+1)$

nach Induktionsannahme ($\sum_{k=1}^{n} k = \frac{n(n+1)}{2}$) folgt:

$\sum_{k=1}^{n+1} k = \frac{n(n+1)}{2} + (n+1) = \frac{n(n+1)+2(n+1)}{2} = \frac{(n+2)(n+1)}{2} = \frac{(n+2)(n+1)}{2} = \frac{(n+1)((n+1)+1)}{2}$

(b) Beweis durch vollständige Induktion:

Induktionsverankerung für $n = 1$: $\sum_{i=0}^{1} 2^i = 3 = 2^{(1+1)} - 1$ Die Aussage gilt also für $n = 1$.

Induktionsschritt von n auf $n+1$: $\sum_{i=0}^{n+1} 2^i = \sum_{i=0}^{n} 2^i \quad + \quad 2^{n+1}$

nach Induktionsannahme ($\sum_{i=0}^{n} 2^i = 2^{n+1} - 1$) folgt:

$\sum_{i=0}^{n+1} 2^i = 2^{n+1} - 1 + 2^{n+1} = 2 \cdot 2^{n+1} - 1 = 2^{(n+1)+1} - 1$

Aufgabe 3 (Relationen): (10 Punkte)

Sei R eine beliebige irreflexive und transitive Relation über einer endlichen Menge X.
Zeigen oder widerlegen Sie folgende Aussagen immer gelten:

- R ist die leere Relation.
- R ist symmetrisch.
- R ist anti-symmetrisch.

Lösung:

- Gegenbeispiel R' über $X = \{1, 2\}$: $R' = \{(1, 2)\}$.
  R' ist irreflexiv, und transitiv.
- Gegenbeispiel. R' (oben) ist nicht symmetrisch.
- Beweis. Zu Zeigen: $xRy \wedge yRx \implies x = y$.
  - Angenommen, $xRy \wedge yRx$.
  - Mit transitivät: xRx.
  - Widerspruch, da R irreflexiv.
  - Demnach ist Prämise immer falsch.
  - Die Aussage gilt.

Aufgabe 4 (Algorithmus): (20 Punkte)

Gegeben sei eine Liste L der Länge $n > 2$, mit natürlichen Zahlen als Eintrag.

- L.first beschreibt den ersten Eintrag.
- L.last beschreibt den letzten Eintrag.
- Jeder Eintrag der Liste enthält eine Referenz auf den vorherigen Eintrag, eine Referenz auf den nächsten Eintrag und den Wert des Eintrags.

Schreiben Sie einen Algorithmus, der die zweitgrößte Zahl in der Liste bestimmt, unter Beachtung der folgenden Einschränkung:

(a) Der Algorithmus soll iterativ sein (keine Rekursion erlaubt).

(b) Der Algorithmus soll rekursiv sein (keine Schleifen erlaubt).

Hinweise:

- Geben Sie den Algorithmus in Pseudo-code oder Java an.
- Sie dürfen keine extra Funktionen nutzen, also insbesondere kein *sort()*
- Die zweitgrößte Zahl kann identisch zur größten Zahl sein, wenn sie mehrmals in der Liste vorkommt. Das heißt die zweitgrößte Zahl der Liste $[1, 2, 4, 3, 4]$ ist 4.

Lösung:

```
second_best_iterative(L):
    max = 0
    sec = 0
    curr = L.first
    do:
        if curr.value >= max:
            sec = max
            max = curr.value
        else if curr.value > sec:
            sec = curr.value
        curr = curr.next
    until curr = L.last
    return sec
```

```
second_best_recursive(L):
    return recursive_second(L.fist, L.last, 0, 0)

recursive_second(first, last, max, sec)
    if first.value >= max:
            sec = max
            max = first.value
        else if first.value > sec:
            sec = first.value

    if first is last:
        return second
    else:
        return recursive_second(first.next, last, max, sec)
```