

2. Präsenzübungsblatt zur Vorlesung Algorithmen und Datenstrukturen (Winter 2025/26)

Aufgabe 1

Gegeben sei folgender Algorithmus `SomeAlgo`. Berechnen Sie den erwarteten Rückgabewert des Algorithmus.

`SomeAlgo(int n)`

```
    s = 0
    for i = 1 to n do
        /* Random(a, b) gibt zufällig und gleichverteilt eine ganze Zahl aus dem
           Bereich {a, ..., b} aus. */
        if Random(1, 2 · i) ≤ i then
            s = s + i
    return s
```

Aufgabe 2 – Tödlicher Bocksbeutel

Vor langer Zeit lebte in Franken ein guter König, der in seinem Weinkeller tausend Bocksbeutel kostbaren Weines verwahrte. Eines Tages ertappten seine Wachen einen bayerischen Agenten, der gerade dabei war, des Königs Wein zu vergiften. Zwar wusste man, dass der Agent es nur geschafft hatte, eine einzige Flasche zu präparieren, allerdings war es ihm im Handgemenge bei seiner Festnahme gelungen, die Flasche zu den anderen zu stellen, so dass niemand wusste, welcher Bocksbeutel vergiftet war. Das Gift war tödlich; schon ein Tropfen des vergifteten Weines führte unweigerlich zum Tod. Dabei wirkte das Gift sehr langsam, erst nach einem Monat erlag man an seinen Folgen.

Da der König seinen Wein sehr liebte, wollte er innerhalb eines Monats wissen, welcher seiner Weine vergiftet wurde. Leider standen ihm dafür nur zehn mutige Vorkoster zur Verfügung. Zeigen Sie dem König, dass diese zehn Vorkoster ausreichen.

Aufgabe 3 – Stapel rekonstruieren

Die Zahlen $1, 2, \dots, n$ werden in aufsteigender Reihenfolge mit Push in einen Stapel eingefügt.

Alle eingefügten Zahlen werden durch Aufruf von Pop wieder aus dem Stapel entfernt, wobei die Methode Pop sowohl zwischen den Aufrufen von Push als auch nach dem letzten Aufruf von Push ausgeführt werden kann. Insgesamt wird die Methode Pop n -mal ausgeführt und wird nie auf einen leeren Stapel angewandt.

- a) Welche der gegebenen Folgen gibt eine mögliche Reihenfolge wieder, in welcher die Zahlen aus dem Stapel mit Pop entfernt wurden? Bei welcher der Folgen ist dies nicht möglich? Begründen Sie Ihre Antwort.

- 3, 2, 4, 1, 8, 9, 10, 7, 6, 5
- 5, 4, 3, 10, 9, 8, 7, 6, 1, 2

- b) Geben Sie in Pseudocode einen Algorithmus an, der (wie in Aufgabenteil (a)) für eine gegebene Permutation A der Zahlen $1, 2, \dots, n$ ermittelt, ob A die Folge der durch Pop entfernten Zahlen sein kann.

Erläutern Sie *zusätzlich zum Pseudocode* die Funktionsweise Ihres Algorithmus in Worten, und begründen Sie, warum der Algorithmus korrekt ist.

Tipp: Rekonstruieren Sie die Folge von Push- und Pop-Anweisungen, und geben Sie false aus, wenn Sie auf einen Widerspruch stoßen.

PABS

Aufgabe 4 – QuickSort

In dieser Aufgabe sollen Sie QuickSort in Java implementieren. Im folgenden sei stets A ein Feld aus Ganzzahlen und $n = A.length$. Außerdem gilt für $\ell, r \in \mathbb{N}$ stets $0 \leq \ell \leq r < n$ und Ihre Methoden sollen stets eine `IllegalArgumentException` werfen, wenn eine dieser Ungleichungen verletzt ist. Erstellen Sie dazu im Paket `sorting` die Klasse `public class QuickSort` und implementieren Sie dort die folgenden Methoden.

- a) Zunächst benötigen Sie die beiden folgenden Subroutinen:

- `public static void swap(int[] A, int i, int j)`
- `public static int partition(int[] A, int ℓ , int r)`

Die Methode `swap` tauscht im Feld A die Einträge i und j aus – nach Aufruf von `swap` steht in $A[i]$ der Wert, der vormals in $A[j]$ stand und umgekehrt. Die Laufzeit von `swap` soll in $O(1)$ sein.

Die Beschreibung der Methode `partition` finden Sie als Pseudocode im entsprechenden Foliensatz der Vorlesung. Die Laufzeit Ihrer Implementation soll in $O(n)$ sein.

- b) Implementieren Sie nun, wie aus der Vorlesung bekannt, mit Hilfe Ihrer Subroutinen

- `public static void quickSort(int[] A, int ℓ , int r)`

c) Implementieren Sie nun

- `public static int randomizedPartition(int[] A, int ℓ , int r),`

welche ein zufälliges Pivot-Element wählt, und nutzen Sie diese Implementation um

- `public static void randomizedQuickSort(int[] A, int ℓ , int r)`

zu realisieren.

Hinweis: Um in Java zufällige Ganzzahlen aus dem Intervall $[0, n]$ zu erzeugen, verwenden Sie ein `Random` Objekt wie folgt:

```
Random rand = new Random();  
int k = rand.nextInt(n);
```

Diese Aufgaben werden eventuell gemeinsam in den Übungen am 09. und 10. Dezember 2025 gelöst. Sie brauchen Sie nicht vorher zu lösen und auch nicht abzugeben.