UNIVERSITÄT DUISBURG ESSEN

Offen im Denken

Fakultät für Mathematik

Dr. Marcel Klinger – Marco Grimm – Nils Heinemann

Trainingsblatt 7 zu Algorithmische Mathematik GHRSGe

WS 2022/23, 22.11.2022

Dieses Blatt bezieht sich auf die Videos 212–302. Abgabefrist ist der 02.12.2022, 20:00 Uhr.

Aufgabe 1: Gegeben sind die Zahlen $-12, 3, 7, 4, 51, 33, -45, 7, 7, -73, 0, 2$. Nutzen Sie das Bucketsort-Verfahren aus Video 212, um diese zu sortieren. Gehen Sie hierzu wie folgt vor:

(a) Definieren Sie eine geeignete monotone Funktion, die für die gegebenen Zahlen innerhalb des Verfahrens verwendet werden kann. (*Tipp: Suchen Sie eine lineare Funktion, die die gegebenen Zahlen in den positiven Bereich bringt und gleichzeitig alle Werte auf das Intervall* $[0, 1)$ *zusammenstaucht.*) (2 Punkte)

(b) Gehen Sie von $k = 5$ Buckets aus und sortieren Sie die gegebenen Zahlen entsprechend der im Verfahren genannten Regel und basierend auf Ihrer Funktion händisch in die Buckets. (3 Punkte)

(c) Sortieren Sie alle Buckets einzeln (hier dürfen Sie z.B. „per Hand“ sortieren) und reihen Sie die Inhalte der Buckets aneinander. (2 Punkte)

(d) Wie lautet eine allgemeine Funktion $f : \mathbb{R} \to [0, 1)$, die für das Bucketsort-Verfahren verwendet werden kann? (*Tipp: Gesucht ist eine Funktion, die unabhängig von den hier konkret gegebenen Werten funktioniert. Gehen Sie davon aus, dass die größte Zahl* m *und die kleinste Zahl* n *heißt.*) (3 Punkte)

Aufgabe 2: Bestimmen Sie Paper-and-Pencil-basiert $\mathrm{ggT}(391, 544)$ mithilfe der in Video 301 gezeigten rekursiven Variante des modernen Euklidischen Algorithmus. Gehen Sie hierbei wie im Diagramm auf Folie 5 des Videos vor, so dass sich die einzelnen Aufrufe nachvollziehen lassen. (5 Punkte)

Aufgabe 3: Die Fakultät $n! := 1 \cdot 2 \cdot \ldots \cdot (n-1) \cdot n$ definiert man i.d.R. wie eben geschehen.

(a) ★ Implementieren Sie die Fakultät dieser Definition folgend in Scratch. Gehen Sie hierbei zunächst iterativ vor, d.h. arbeiten Sie mit einer geeigneten Schleife. Ein Benutzer soll zu Beginn des Programms nach jenem n gefragt werden, wozu schließlich $n!$ berechnet werden soll. (3 Punkte)

(b) ★ Die Fakultät lässt sich aber auch durch eine rekursive Definition erklären, nämlich:

$$n! := \begin{cases} 1 & n = 1 \\ (n-1)! \cdot n, & n > 1 \end{cases}$$

Schreiben Sie auf Basis dieser Definition ein Scratch-Programm, das rekursiv programmiert ist.
(*Tipp: Gesucht ist also ein Programm, das nicht iterativ wie zuvor, d.h. z.B. über eine Schleife, arbeitet. Gehen Sie stattdessen so vor, dass Sie einen Eigenen Block nutzen, der in seinem eigenen Programmcode auf sich selbst zugreift.*) (7 Punkte)

Die folgende Aufgabe ist eine reine Bonusaufgabe. Ihre Abgabe erfolgt erst zusammen mit Ihrer Bearbeitung von Trainingsblatt 8.

Aufgabe 4: Wir betrachten das sog. *Collatz-Problem*: Hierbei handelt es sich um ein 1937 von Lothar Collatz (1910–1990) formuliertes mathematisches Problem, welches bis heute ungelöst ist. Bei dem Problem geht es um Zahlenfolgen, die nach einem gewissen Bildungsgesetz konstruiert werden:

(1) Wähle eine beliebige Zahl $n \in \mathbb{N}$.

(2) Falls n gerade ist, bestimme $\frac{n}{2}$.

(3) Falls n ungerade ist, bestimme $3n + 1$.

(4) Wiederhole die Vorgehensweise mit der erhaltenen Zahl.

Die bis heute unbewiesene Vermutung lautet, dass man – unabhängig davon, mit welchem $n \in \mathbb{N}$ man startet – immer in den *Zyklus* $4, 2, 1$ läuft und hier „gefangen" bleibt.

(a) Überprüfen Sie die Behauptung händisch für die Startzahl $n = 19$. (`2 Bonuspunkte`)

(b) ★ Implementieren Sie ein Scratch-Programm, das die Collatz-Vermutung für eine beliebige Startzahl n_0 überprüft. Arbeiten Sie hierbei mit einer iterativen Programmierung. Es ist also im Wesentlichen mit einem Wiederhole-Block zu arbeiten (`4 Bonuspunkte`)

(c) ★ Implementieren Sie ein Scratch-Programm, das die Collatz-Vermutung für eine beliebige Startzahl n_0 überprüft. Arbeiten Sie hierbei mit einer rekursiven Programmierung. Es ist also im Wesentlichen mit einem Eigenen Block zu arbeiten, der auf sich selbst verweist. (`4 Bonuspunkte`)

(d) Überprüfen Sie mit Hilfe einer Ihrer Implementierungen alle natürlichen Zahlen bis zu einer von Ihnen gewählten möglichst großen Maximalzahl n_{max}. Wie sind Sie vorgegangen? Schaffen Sie es die Collatz-Vermutung für eine größere Maximalzahl als Ihre KommilitonInnen zu „beweisen"? Dokumentieren Sie Ihre Arbeiten geeignet mit einem entsprechenden Beleg (z.B. mit einem Screen-Capture). (*Tipp: Dasjenige Team, das den Nachweis bis zur größten Maximalzahl erbringt, erhält zusätzliche Bonuspunkte*) (`5 + ggfs. 20 Bonuspunkte`)