

# Einführung in die Programmierung

## Probeklausur

### 1 Funktionen

Gegeben sind die Funktionen

$$g(x) := \begin{cases} 0 & \text{für } x < 0 \\ x \cdot (1 - x) & \text{für } 0 \leq x < 1 \\ 0 & \text{für } x \geq 1 \end{cases}$$

und

$$h(x, y) := \begin{cases} -x & \text{für } x < 0 \text{ und } y < 0 \\ -y & \text{für } x < 0 \text{ und } y \geq 0 \\ x & \text{für } x \geq 0 \text{ und } y < 0 \\ y & \text{für } x \geq 0 \text{ und } y \geq 0. \end{cases}$$

Schreiben Sie zwei Maple-Prozeduren `g` und `h`, die diese beiden Funktionen berechnen.

### 2 Listen

Die im Folgenden zu schreibenden Prozeduren sollen jeweils einen Parameter (z.B. `li`) erhalten, von dem vorausgesetzt werden darf, dass er mit einer nicht-leeren Liste von Zahlen belegt ist, z.B. `[1, 2, 3, 4]`. Von den Maple-Bibliotheksfunktionen dürfen dabei (falls überhaupt benötigt) nur die Funktionen `seq()`, `nops()` und `op()` benutzt werden

- a) Schreiben Sie zwei Prozeduren `maxelem_for` und `maxelem_rek`, die beide das größte Element der übergebenen Liste als Funktionswert liefern (demnach sollen zum

Beispiel sowohl `maxelem_for([1,2,3,2,1])` als auch `maxelem_rek([1,2,3,2,1])` das Ergebnis 3 liefern.

Dabei soll `maxelem_for` mit einer `for`-Schleife implementiert werden, `maxelem_rek` dagegen soll das Ergebnis rekursiv bestimmen.

Tipp: Für die rekursive Variante lohnt es sich vielleicht, sich zwei Hilfsprozeduren `first` und `rest` zu definieren (wie sie aus Vorlesung/Übung bekannt sind).

- b) Schreiben Sie eine weitere Prozedur `maxelem_anz`, deren Ergebnis eine Sequenz aus zwei Zahlen ist, nämlich dem größten Element der Liste, sowie der Anzahl, wie oft dieses in der Liste vorkommt.

Für `maxelem_anz([1,4,2,3,4,4])` sollte also das Ergebnis 4,3 geliefert werden, da die 4 dreimal in der Liste vorkommt.

Die Prozedur kann wahlweise mit einer `for`-Schleife oder rekursiv implementiert werden.

### 3 Bäume

Ähnlich wie in der Vorlesung definieren wir einen (nicht-leeren!) Baum als 2-elementige Liste, wobei:

- das erste Listenelement einen beliebigen Ausdruck enthält
- das zweite Element eine Liste von Bäumen ist (die „Söhne“). Diese Liste darf leer sein, dann ist der (Unter-)Baum ein „Blatt“. Ebenso wenig ist die Zahl der Söhne nach oben begrenzt.

Schreiben Sie folgende Prozeduren:

- a) eine Funktion `tiefe`, die einen Baum als Parameter enthält und dessen maximale Tiefe zurückgibt.
- b) eine Funktion `knotenzahl`, die einen Baum als Parameter enthält und die Zahl seiner Knoten zurückgibt.

Sie dürfen auch beide Prozeduren zusammenfassen in eine Prozedur `knoten_tiefe`, die beide Rückgabewerte gleichzeitig (etwa als Sequenz) liefert.