

## Übungen zu Informatik I (Lösungsvorschlag)

### Aufgabe 2-1

### Zinsberechnung

(keine Abgabe)

- a) Der folgende Algorithmus berechnet das Guthaben nach  $n$  Jahren für die einmalige Einzahlung eines Betrags  $b$  bei Zinssatz  $p$ .

#### Algorithmus *Zinsen1*.

Falls  $n = 1$ : Das Ergebnis ist  $(1 + p) \cdot b$ .

Falls  $n > 1$ : Führe den Algorithmus *Zinsen1* für die Werte  $b$ ,  $p$  und  $n - 1$  durch, berechne  $1 + p$  und multipliziere die Ergebnisse.

Der Algorithmus *Zinsen1* ist terminierend, da  $n$  bei jedem Aufruf kleiner wird,  $n \geq 1$  gilt und der Wert für  $n = 1$  explizit definiert ist.

Der Algorithmus *Zinsen1* ist deterministisch. Es lassen sich auch nichtdeterministische und parallele Formulierungen angeben:

#### Algorithmus *Zinsen2*.

Falls  $n = 1$ : Das Ergebnis ist  $(1 + p) \cdot b$ .

Falls  $n > 1$ : Führe *einen* der folgenden Schritte durch:

- Führe den Algorithmus *Zinsen2* für die Werte  $b$ ,  $p$  und  $n - 1$  durch, berechne  $1 + p$  und multipliziere die Ergebnisse.
- Berechne  $1 + p$ , führe den Algorithmus *Zinsen2* für die Werte  $b$ ,  $p$  und  $n - 1$  durch und multipliziere die Ergebnisse.

Dieser Algorithmus ist nichtdeterministisch, da für  $n > 1$  die Berechnung der Summe  $1 + p$  vor oder nach der Berechnung von *Zinsen2*( $b, n - 1$ ) erfolgen kann. Eine nebenläufige Variante ist:

#### Algorithmus *Zinsen3*.

Falls  $n = 1$ : Das Ergebnis ist  $(1 + p) \cdot b$ .

Falls  $n > 1$ : Führe *gleichzeitig* die folgenden Schritte durch:

- Berechne  $1 + p$
- Führe den Algorithmus *Zinsen3* für die Werte  $b$ ,  $p$  und  $n - 1$  durch.

Multipliziere die Ergebnisse der beiden parallelen Berechnungen.

Der Algorithmus *Zinsen3* ist parallel, da die beiden Berechnungen  $(1 + p)$  und *Zinsen3*( $b, n - 1$ ) gleichzeitig ausgeführt werden.

Der Algorithmus ist in jedem Fall determiniert, da das Ergebnis der Berechnung nicht von der Reihenfolge abhängt, in der die einzelnen Teilterme berechnet werden.

Man kann diese Berechnung geschickter in geschlossener Form schreiben:

#### Algorithmus *Zinsen4*.

Das Ergebnis ist  $(1 + p)^n \cdot b$ .

Der Algorithmus *Zinsen4* ist deterministisch, determiniert und terminierend.

b) **Algorithmus Zinsen5.**

Falls  $n = 1$ : das Ergebnis ist  $(1 + p) \cdot b$ .

Falls  $n > 1$ : Führe den Algorithmus *Zinsen5* für die Werte  $b$ ,  $p$  und  $n - 1$  durch, addiere  $b$  zum Ergebnis und multipliziere dieses Ergebnis mit  $1 + p$ .

Zu Terminierung, Determinismus und Determiniertheit siehe Teilaufgabe a).

Man kann auch dieses Ergebnis in geschlossener Form schreiben:

$$\frac{(p + 1)^{(n+1)} - (p + 1)}{p} \cdot b$$

Allerdings ist es bei dieser Aufgabe schwieriger eine geschlossene Form zu finden als in Teilaufgabe a). Es ist auch nicht offensichtlich, dass diese Formel das gleiche Ergebnis berechnet wie Algorithmus *Zinsen5*. Die Korrektheit lässt sich durch Induktion über  $n$  beweisen.

## Aufgabe 2-2

## Wechselgeld

(keine Abgabe)

Wir modellieren die auszugebenden Münzen durch Multimengen. Eine Multimenge ist, ähnlich wie eine Menge, eine Zusammenfassung von Dingen zu einer Gesamtheit, wobei die Reihenfolge der Elemente nicht berücksichtigt wird. Im Gegensatz zu Mengen dürfen die einzelnen Elemente in einer Multimenge aber mehrfach vorkommen. Z.B. ist  $\{1, 2, 1, 1, 3\}$  eine Multimenge aber keine Menge. Die auszugebenden Münzen und Scheine können wir dann als Multimenge ihrer Beträge darstellen.

Die erste Variante des Algorithmus berechnet zuerst die Höhe des Wechselgeldes für den Betrag  $n$  und dann die Stückelung.

### Algorithmus W1

Setze  $r = 20 - n$ .

Falls  $r \geq 5$ :  $\{5\} \cup W1(n + 5)$

Sonst: Falls  $r \geq 2$ :  $\{2\} \cup W1(n + 2)$

Sonst: Falls  $r \geq 1$ :  $\{1\} \cup W1(n + 1)$

Sonst: kein Rückgeld.

Dieser Algorithmus ist deterministisch und determiniert, durch die kaskadierenden Fallunterscheidungen ist die Ausführungsreihenfolge festgelegt.

Wir erhalten einen nichtdeterministischen Algorithmus, indem wir nicht mehr verlangen, dass zuerst 5-Euro-Scheine, dann 2-Euro-Münzen und schließlich 1-Euro-Münzen herausgegeben werden:

### Algorithmus W2

Falls  $n = 20$ : Beende den Algorithmus mit Ergebnis  $\emptyset$ .

Sonst: Setze  $r = 20 - n$ . Fahre mit einem beliebigen Schritt fort, dessen Bedingung erfüllt ist.

Falls  $r \geq 5$ :  $\{5\} \cup W2(n + 5)$

Falls  $r \geq 2$ :  $\{2\} \cup W2(n + 2)$

Falls  $r \geq 1$ :  $\{1\} \cup W2(n + 1)$

Dieser Algorithmus ist nicht deterministisch und nicht determiniert, da es passieren kann, dass das Wechselgeld nur aus 1-Euro Münzen besteht. Im Unterschied zu *W1* ist nicht gewährleistet, dass eine möglichst geringe Anzahl von Münzen zurückgegeben wird.

Eine andere Algorithmusidee ist, das Wechselgeld heraufzuzählen:

### Algorithmus W3

Falls  $n = 20$ : Beende den Algorithmus mit Ergebnis  $\emptyset$ .

Sonst: Falls die letzte Ziffer von  $n$  eine 2, 4, 7 oder 9 ist:  $\{1\} \cup W3(n+1)$

Sonst: Falls die letzte Ziffer von  $n$  eine 1, 3, 6 oder 8 ist:  $\{2\} \cup W3(n+2)$

Sonst:  $\{5\} \cup W3(n+5)$  (In diesem Fall ist die letzte Ziffer von  $n$  eine 0 oder eine 5.)

Auch diese Variante lässt sich nichtdeterministisch machen, indem wir die Reihenfolge der Münz-  
ausgabe nicht festlegen:

#### **Algorithmus W4**

Falls  $n = 20$ : Beende den Algorithmus mit Ergebnis  $\emptyset$ . Sonst: Fahre mit einem beliebigen Schritt  
fort, dessen Bedingung erfüllt ist.

Falls  $n < 20$  und die letzte Ziffer von  $n$  eine 2, 4, 7 oder 9 ist:  $\{1\} \cup W4(n+1)$

Falls  $n < 20$  und die letzte Ziffer von  $n$  eine 1, 2, 3, 6, 7 oder 8 ist:  $\{2\} \cup W4(n+2)$

Falls  $n < 20$  und die letzte Ziffer von  $n$  eine 0, 1, 2, 3, 4 oder 5 ist:  $\{5\} \cup W4(n+5)$

Diese Variante ist – im Unterschied zu  $W3$  – zwar nichtdeterministisch, aber determiniert, da in  
einer Multimenge die Reihenfolge der Elemente unwesentlich ist.

*Bemerkung:* Für  $n \neq 20$  ist immer mindestens eine der Bedingungen erfüllt.

Wenn man die Datenmodellierung ändert und statt dessen eine *Folge* von Münzen und Scheinen  
zurückgibt, ist der Algorithmus  $W4$  nicht deterministisch und nicht determiniert.

### **Aufgabe 2-3**

#### **Parkgebühren**

(6 Punkte)

Der folgende Algorithmus berechnet die Parkgebühren für eine Parkdauer von  $m$  Minuten.

#### **Algorithmus Parkgebühren**

Falls  $m \leq 60$ : 4 Euro

Sonst: Falls  $m \leq 120$ : 6 Euro

Sonst: Setze  $h = (m - 120)/60$ , aufgerundet zur nächsten ganzen Zahl. Das Ergebnis ist  $6 + h$   
Euro.

*Bewertung:* Je 1 Punkt für die Fälle  $m \leq 60$  und  $m \leq 120$ , 3 Punkte für die richtige Berechnung  
der Stundenanzahl bei längerer Parkdauer, 1 Punkt für die Berechnung des Ergebnisses.

### **Aufgabe 2-4**

#### **Binärzahlen**

(6 Punkte)

Der Algorithmus *Binärzahlen* berechnet die minimale Anzahl der Binärstellen, die zur Darstel-  
lung der Zahl  $z$  benötigt werden, wobei  $z \geq 1$  gilt.

Bei dem im Folgenden benutzten Divisionsoperator *div* wird nur der ganzzahlige Anteil der  
Division (d.h. ohne Nachkommastellen) als Ergebnis zurückgegeben.

#### **Algorithmus Binärzahlen**

Falls  $z = 1$ : 1

Sonst  $1 + \text{Binärzahlen}(z \text{ div } 2)$

Andere Lösungsmöglichkeit: mit der Logarithmusfunktion (zur Basis 2), d.h.  $\lceil \log_2(z) \rceil + 1$  be-  
rechnen. Dabei bezeichnet  $\lceil r \rceil$  für eine reelle Zahl  $r$  den ganzzahligen Anteil von  $r$ .

(Eine weitere Alternative ist:  $\log_2(z+1)$  berechnen und das Ergebnis aufrunden.)

*Bewertung:* 2 Punkte für den Anfangsfall, 4 Punkte für die richtige Rekursion. Bei Verwendung  
der Log-Funktion: 4 Punkte für Berechnung von dem ganzzahligen Anteil von  $\log_2(z)$ , 2 Punkte  
für das Addieren von 1. (bzw. 4 Punkte für Berechnung des Logarithmus, 2 Punkte für das  
Aufrunden.)