Zusatzaufgaben OMP – SML

Aufgabe 1:

Hinweis: Addition, Subtraktion, Multiplikation und (Integer-)Division können Sie in SML als +, -, * und div schreiben.

a) Schreiben Sie eine Funktion in Standard ML
val double = fn : i n t l i s t −> i n t l i s t
die eine Liste von Integer-Werten erwartet und jedes Element der Liste mit 2 multipliziert. Beispiel: double([1,2,3]); ergibt val it = [2,4,6] : int list.

b) Schreiben Sie eine Funktion in Standard ML
val time s = fn : i n t ∗ i n t −> i n t
die zwei positive Zahlen miteinander multipliziert. Vorsicht, Sie dürfen für diese Funktion die Standard-Multiplikation * nicht verwenden!
Zur Erinnerung: Die Multiplikation a · b von a und b ist definiert als a + a + . . . + a (b-mal).
Hinweis: Definieren Sie die Funktion rekursiv.
Hinweis: Vergessen Sie nicht, Basisfälle zu definieren.
Hinweis: Negative Zahlen müssen Sie hier nicht betrachten.
Beispiel: times(2,3); ergibt val it = 6 : int.

c) Gegeben sei eine Datenbank von Klausurergebnissen in folgendem Format:
val db = [(" Ali c e " , 230) , ("Bob" , 300) , (" C h a rli e " , 3 7 0)] ;
Schreiben Sie eine Funktion in Standard ML
val avg = fn : (' a ∗ i n t) l i s t −> i n t
die die Durchschnittsnote (arithmetisches Mittel) der Klausur berechnet.
Hinweis: Sie müssen ggf. mehrere Hilfsfunktionen definieren. Sie dürfen keine vordefinierten Funktionen verwenden!
Hinweis: Das arithmetische Mittel für eine Folge von n Zahlen (x1, x2, . . . , xn) ist definiert als $\frac{x_1+x_2+\ldots+x_n}{n} = \frac{1}{n}\sum_{i=1}^{n} x_i$
Beispiel: avg(db); ergibt val it = 300 : int.

Quelle: Altklausuren OMP 2018

Aufgabe 2:

Gegeben sei die Exception `Empty`, die in Standard ML mit dem Schlüsselwort `raise` geworfen werden kann:

```
exception Empty;
```

Schreiben Sie folgende Funktionen in Standard ML:

a)
```
val hd = fn : 'a list -> 'a
```

die das erste Element einer Liste zurückgibt (*head*).
Beispiel: `hd([1,2,3]);` ergibt `val it = 1 : int`.

b)
```
val tl = fn : 'a list -> 'a list
```

die den Rest einer Liste nach dem ersten Element zurückgibt (*tail*).
Beispiel: `tl([1,2,3]);` ergibt `val it = [2,3] : int list`.

c)
```
val filter = fn : ('a -> bool) * 'a list -> 'a list
```

die alle Elemente aus einer übergebeben Liste (zweiter Parameter) herausfiltert, für die die übergebene Funktion (erster Parameter) den Wert `false` zurückgibt.
Beispiel: Sei `odd` eine Funktion, die nur für ungerade Zahlen `true` zurückgibt.
`filter(odd, [1,2,3]);` ergibt `val it = [1,3] : int list`.

d) Gegeben sei eine Datenbasis als Liste von Zweitupeln aus Name und Alter von Personen:

```
val db = [("Alice", 21), ("Bob", 32), ("Charlie", 27)];
```

Schreiben Sie eine Funktion

```
val filterAge = fn : ('a * int) list -> ('a * int) list
```

welche diese Datenbasis auf Personen unter 30 Jahren beschränkt. Verwenden Sie dabei die Funktion `filter` (unabhängig davon, ob Sie die Methode `filter` bereits geschrieben haben).
Beispiel: `filterAge(db);` ergibt

```
val it = [("Alice",21),("Charlie",27)] : (string * int) list
```

Achten Sie bei allen Funktionen auf eine sinnvolle Fehlerbehandlung bei ungültigen Eingaben, sofern nötig.

Quelle: Altklausuren OMP 2018

Quelle: Altklausuren OMP 2018