

8. Übung

Abgabe in der Woche vom 26. - 30. Juni 2006

Aufgabe 25 (1 + 3 + 1 Punkte)

Entwerfen Sie eine Datenstruktur `Stack` zur Speicherung von 2-Tupeln von Integern und die dazu gehörigen Funktionen

- `newstack` – bekommt eine Zahl `n` und erzeugt einen $(0,0)$ -initialisierten Stack der Höhe `n`,
- `push` – bekommt als Argumente ein Element und einen Stack und legt das Element oben auf den Stack,
- `pop` – bekommt als Argument einen Stack und gibt zurück das oberste Element des Stacks und den Stack, der um dieses Element verringert wurde.

Gegeben sei weiterhin die Funktionsdefinition

```
mystack n = (push (1,2) . push (2,2) . push (2,3)) (newstack n).
```

Geben Sie den Typ der Funktion an und begründen Sie diesen.

Aufgabe 26 (2 + 5 + 4 + 3 + 1 Punkte)

Entwerfen Sie eine Datenstruktur `Bruch` zur Speicherung von Brüchen mit separatem Zähler und Nenner. Lassen Sie Haskell Ihren Datentyp in die Klassen `Eq` und `Show` automatisch einfügen.

1. Entwerfen Sie die Funktionen `reziproke`, `summe` und `division` zur Reziprokbildung, Addition und Division von Brüchen. Denken Sie daran, Brüche immer zu kürzen!
2. Definieren Sie eine Funktion `ausgabe` zur formatierten Darstellung eines Bruches auf dem Bildschirm. Beispiel:

```
ausgabe Bruch 110 5586  
>> 55  
>>----  
>>2793
```

3. Ordnen Sie Ihren Datentyp `Bruch` in die Typklassen `Ord` und `Num` ein und definieren Sie die Operationen `<`, `>` und `*`. Benutzen Sie für die `*` Operation die Funktion `division` aus Punkt 1.
4. Testen Sie die beiden Brüche `Bruch 1 5` und `Bruch 2 10` auf `==`, `<` und `>`. Erklären das auftretende Phänomen und machen Sie einen Lösungsvorschlag.