Institut für Informatik
der Universität München
Prof. Dr. M. Hofmann
Dr. J. Johannsen, Dr. M. Lange

WS 03/04

8.1.04

Übungen zur Vorlesung
Informatik I

Blatt 10

Abgabe der Hausaufgaben spätestens am 19.01.04, 11:00 Uhr. Programmieraufgaben über
`http://miles.tcs.informatik.uni-muenchen.de/info1/abgabe.php`,
schriftliche Aufgaben auf Papier im Briefkasten in der Theresienstraße 39, 1. Stock. Notieren Sie Namen, Matrikelnummern und Ihre Übungsgruppe auf den Blättern. Bearbeitung in Gruppen zu max. 3 Personen ist zulässig. Besprechung der Aufgaben in den Übungen ab 26.01.04.

Programmieraufgabe P-39 (`listen.ml`): **6 Punkte**
Schreiben Sie Ocaml-Funktionen

- `minmax : int list -> int * int`, die das Minimum und das Maximum einer Liste von Integern berechnet. Behandlen Sie den Fall der leeren Liste mit einer Exception.
- `summanden : int -> (int * int) list`, die zu einer gegebenen, positiven Zahl n die Liste aller Paare (n_1, n_2) berechnet, so dass $n = n_1 + n_2$ und $n_1 \leq n_2$ gilt.
- `faktoren : int -> (int * int) list`, die zu einer gegebenen, positiven Zahl n die Liste aller Paare (n_1, n_2) berechnet, so dass $n = n_1 * n_2$ und $1 < n_1 \leq n_2$ gilt. *Hinweis:* Mit `truncate (floor (sqrt (float_of_int m)))` erhalten Sie die abgerundete Quadratwurzel von m als `int`.

Programmieraufgabe P-40 (`sieb.ml`): **5 Punkte**
Implementieren Sie eine Funktion `primzahlen : int -> int list`, die für ein eingegebenes n die Liste aller Primzahlen p mit $p \leq n$ ausgibt.

Benutzen Sie dazu das *Sieb des Eratosthenes*: es wird zunächst die Liste aller Zahlen von 2 bis n erzeugt, aus dieser werden dann alle Vielfachen von 2 gestrichen, aus der verbleibenden Liste werden alle Vielfachen von 3 gestrichen, dann alle Vielfachen von 5 etc. Im Allgemeinen werden immer aus der Restliste die Vielfachen der kleinsten überlebenden Zahl gestrichen.

Programmieraufgabe P-41 (`permutiere.ml`): **5 Punkte**
Schreiben Sie eine Funktion

```
permutiere : 'a list -> 'a list list ,
```

die zu einer gegebenen Liste l die Liste aller Permutationen von l berechnet. Sie können davon ausgehen, dass alle Elemente in l verschieden sind.

Hinweis: Dies können Sie folgendermassen erreichen. Schreiben Sie zuerst eine Funktion

```
einfuegen : 'a -> 'a list -> 'a list list ,
```

die ein x und eine Liste l nimmt, und x an jeder möglichen Stelle in l einfügt.

– bitte wenden –

Z.B. ist

```
# einfuegen 1 [2;3] ;;
- : int list list = [[1; 2; 3]; [2; 1; 3]; [2; 3; 1]]
```

Schreiben Sie dann eine Funktion

```
listen_einfuegen : 'a -> 'a list list -> 'a list list ,
```

die ein x und eine Liste von Listen l nimmt und x mittels `einfuegen` in jedes Element von l einfügt. Die Ergebnisse werden dann durch @ aneinandergehängt.

Programmieraufgabe P-42 (`goldbach.ml`): **4 Punkte**

Nachdem Fermats letzter Satz vor einigen Jahren bewiesen wurde, ist die Goldbach-Vermutung eines der größten verbleibenden Probleme der Mathematik. Christian Goldbach äußerte 1742 in einem Brief an den großen Mathematiker Leonhard Euler die Annahme, dass sich jede gerade Zahl, die größer als zwei ist, als Summe zweier Primzahlen darstellen lässt, z.B. $10 = 3 + 7$ oder $98 = 19 + 79$.

Schreiben Sie eine Funktion `goldbach : int -> int * int`, die ein n nimmt und

- eine Exception `Zu_klein` auslöst, falls n kleiner als 4 ist,
- eine Exception `Ungerade` auslöst, falls n ungerade ist,
- ein Paar von ungeraden Primzahlen (x, y) liefert, so dass $n = x + y$ ist, und
- eine Exception `Goldbach_hatte_Unrecht` zusammen mit dem gefundenen Gegenbeispiel n auslöst, falls Goldbach mit seiner Vermutung falsch lag.