Berthold Hoffmann
Markus Roggenbach

Übungen "Programmiersprachen"
Besprechung 27.1.03
Thema "Funktionale Sprachen"

Aufgabe 1

Schreibe ML-Funkionen `sum` und `product`, die alle Elemente einer Zahlenliste aufsummie-renkl bzw. aufmultiplizieren.
Tipp: Benutze dazu die ML Funktion

```
foldl : ('a * 'b -> b') -> 'b -> 'a list -> 'b

fun foldl f e []      = e
  | foldl f e (x::xs) = foldl f (f(x,e)) xs
```

Aufgabe 2

Schreibe eine ML-Funktion `primes`, die beliebig viele Primzahlen nach dem Algorithmus 'Sieb des Erathostenes' aufzählt.
Tipp: Schreibe dazu einen ML-Datentyp `sequence` zur Repräsentation unendlicher Listen (der ML-Typ `unit` repräseniert die einelementige Menge bestehend aus dem 0-Tupel '()'):

```
- exception Empty;
exception Empty
- datatype 'a seq = Nil | Cons of 'a * (unit -> 'a seq);
datatype 'a seq = Cons of 'a * (unit -> 'a seq) | Nil
- fun cons (x,xq) = Cons(x, fn() => xq );
val cons = fn : 'a * 'a seq -> 'a seq
- fun from k = Cons(k, fn()=> from(k+1));
val from = fn : int -> int seq
- from 1;
GC #0.0.0.0.1.28:   (0 ms)
val it = Cons (1,fn) : int seq
- fun take (xq,0) = [] | take(Nil, n) = raise Empty | take
(Cons(x,xf),n) = x ::take(xf(),n-1);
val take = fn : 'a seq * int -> 'a list
- take (from 30, 7);
val it = [30,31,32,33,34,35,36] : int list
```