Homework 2
COSE212, Fall 2015

Hakjoo Oh

Due: 10/09, 24:00

Problem 1 Write a function `filter`

```
filter : ('a -> bool) -> 'a list -> 'a list
```

Given a predicate `p` and a list `l`, `filter p l` returns all the elements of the list l that satisfy the predicate p. The order of the elements in the input list is preserved. For example,

```
# filter (fun x -> x mod 2 = 0) [1;2;3;4;5];;
- : int list = [2; 4]
# filter (fun x -> x > 0) [5;-1;0;2;-9];;
- : int list = [5; 2]
# filter (fun x -> x * x > 25) [1;2;3;4;5;6;7;8];;
- : int list = [6; 7; 8]
```

Problem 2 Write a function

```
zipper: int list * int list -> int list
```

which receives two lists a and b as arguments and combines the two lists by inserting the ith element of a before the ith element of b. If b does not have an ith element, append the excess elements of a in order. For example,

```
# zipper ([1;3;5],[2;4;6]);;
- : int list = [1; 2; 3; 4; 5; 6]
# zipper ([1;3],[2;4;6;8]);;
- : int list = [1; 2; 3; 4; 6; 8]
# zipper ([1;3;5;7],[2;4]);;
- : int list = [1; 2; 3; 4; 5; 7]
```

Problem 3 Define the function `iter`:

```
iter : int * (int -> int) -> (int -> int)
```

such that

$$\texttt{iter}(n, f) = \underbrace{f \circ \cdots \circ f}_{n}.$$

When $n = 0$, `iter(`n`, f)` is defined to be the identity function. When $n > 0$, `iter(`n`, f)` is the function that applies f repeatedly n times. For instance,

```
iter(n, fun x -> 2+x) 0
```

evaluates to $2 \times n$.

Problem 4 Write a function

```
diff : aexp * string -> aexp
```

that differentiates the given algebraic expression with respect to the variable given as the second argument. The algebraic expression `aexp` is defined as follows:

```
type aexp =
  | Const of int
  | Var of string
  | Power of string * int
  | Times of aexp list
  | Sum of aexp list
```

For example, $x^2 + 2x + 1$ is represented by

```
Sum [Power ("x", 2); Times [Const 2; Var "x"]; Const 1]
```

and differentiating it (w.r.t. "x") gives $2x + 2$, which can be represented by

```
Sum [Times [Const 2; Var "x"]; Const 2]
```

Note that the representation of $2x + 2$ in `aexp` is not unique. For instance, the following also represents $2x + 2$:

```
Sum
 [Times [Const 2; Power ("x", 1)];
  Sum
   [Times [Const 0; Var "x"];
    Times [Const 2; Sum [Times [Const 1]; Times [Var "x"; Const 0]]]];
  Const 0]
```

Problem 5 Consider the following expressions:

```
type exp = X
         | INT of int
         | ADD of exp * exp
         | SUB of exp * exp
         | MUL of exp * exp
         | DIV of exp * exp
         | SIGMA of exp * exp * exp
```

Implement a calculator for the expressions:

```
calculator : exp -> int
```

For instance,

$$\sum_{x=1}^{10}(x * x - 1)$$

is represented by

```
SIGMA(INT 1, INT 10, SUB(MUL(X, X), INT 1))
```

and evaluating it should give 375.

How to submit

1. Download the homework 2 template file (`hw2.ml`) from the course webpage: `http://prl.korea.ac.kr/~hakjoo/home/courses/cose212/2015`
2. Replace all `(* TODO *)` in `hw2.ml` by your own code. You can define any helper functions in `hw2.ml`.
3. Submit the single file `hw2.ml` via Blackboard.