Homework 7

Please email your solutions to Rongdi Huang (rh1424@nyu.edu). Solutions to programming exercises must be submitted electronically as plain text files. No exotic formats, please!

The deadline for Homework 7 is November 21.

For the following problems, make sure your code runs under SML/NJ.

Problem 1  Big Natural Numbers (10 Points)

On the course website you find a file BIG_NAT.sig declaring a signature BIG_NAT for manipulating arbitrarily large natural numbers. You further find a file BigNat.sml declaring a structure BigNat with an incomplete implementation of the signature BIG_NAT.

(a) Understand the implementation of big natural numbers in the structure BigNat and complete the missing functions sub (subtraction with carry) and exp (exponentiation). When you implement sub, pay attention to the elimination of leading zeros to preserve the uniqueness of the representation. (6 Points)

(b) Load the signature and structure into the interpreter. Write a function fact: bignat -> bignat that computes the factorial function on big natural numbers using BigNat. Further, write a function factDigits: int -> int that for a given n ∈ N, computes the number of decimal digits of n!. How many decimal digits has the factorial of 333? (4 Points)

Problem 2  Big Integers (10 Points)

Write a module implementing the type bigint representing arbitrarily large integers. The signature of the module should contain the following operations:

```
  eqtype bigint

  val fromInt : int -> bitint
  val toInt : bigint -> int  (* [Overflow] *)
  val toString : bigint -> string

  val < : bigint * bigint -> bool
  val <= : bigint * bigint -> bool
  val isEven : bigint -> bool
  val isNegative : bigint -> bool

  val ~ : bigint -> bigint
  val + : bigint * bigint -> bigint
  val - : bigint * bigint -> bigint
  val * : bigint * bigint -> bigint
```
val div : bigint * bigint -> bigint (* [Div] *)
val mod : bigint * bigint -> bigint (* [Div] *)
val abs : bigint -> bigint (* absolute value *)
val exp : bigint * bigint -> bigint (* [Domain] *)

Implement the type bigint as follows:

type bigint = bool * BigNat.bignat

where a boolean marking true indicates that the represented number is negative. The function toString should use the symbol ~ as the sign for negative numbers, according to the SML syntax. Hint: Be careful to ensure that the representation of numbers is unique so that the meaning of equality of two bigint values is correct. In particular, the number 0 must be represented uniquely.

Problem 3  Multisets (10 Points)

Multisets are sets in which elements can appear multiple times. Formally, a multiset over a set $X$ is a function $X \to \mathbb{N}$. We consider the following operations on multisets:

$MS(X) = X \to \mathbb{N}$

$\text{count} : MS(X) \times X \to \mathbb{N}$

$\text{count}(m, x) = m(x)$

empty : MS(X)

empty = $\lambda x \in X.0$

insert : MS(X) × X → MS(X)

insert($m, x'$) = $\lambda x \in X.\text{if } x = x' \text{ then } m(x) + 1 \text{ else } m(x)$

union : MS(X) × MS(X) → MS(X)

union($m_1, m_2$) = $\lambda x \in X. m_1(x) + m_2(x)$

(a) Declare a signature MULTISET that describes multisets using a type

\textbf{type } ''a mset

and the above operations. (2 Points)

(b) Declare a structure Multiset1 with signature MULTISET that implements multisets using the type

\textbf{type } ''a mset = ''a -> int

(3 Points)

(c) Declare a structure Multiset2 with signature MULTISET that implements multisets with the type

\textbf{type } ''a mset = (''a * int) list

(5 Points)

Note that in SML a double-quoted name ''a is a type variable that stands for an arbitrary type equipped with an equality predicate $=$ : ''a * ''a -> bool.