COMP 302: Assignment 1, Summer 2008.
SML and lists

Due Date: 15th May, in class.

Question 1: Here is a warm-up question.

1.a) (8 points) Write an SML function `duplicates` of type `int list -> bool`, which takes as input a list of integers and returns `true` if the list contains a duplicate entry, and `false` otherwise.

1.b) (2 points) The obvious algorithm for `duplicates` has a running time of $\Theta(n^2)$. Describe, briefly, how to implement `duplicates` so that it has a running time of $O(n \log n)$.

Question 2: In this question, we will be using ordered lists of `ints` to represent finite sets of integers. For instance, the set $\{1, 2, 3, 4, 5\}$ will be represented by the list `1::2::3::4::5::nil`, and the set $\{-3, 0, 3, 300\}$ is represented by the list `~3::0::3::300::nil`.

2.a) (10 points) Write an SML function called `contains`, which takes as input an `int list` and an `int`, which returns `true` if the integer occurs within the list, and `false` otherwise.

2.b) (15 points) Write an SML function `union` which takes as input two sets represented by ordered `int lists`, and outputs the union of two sets. Your algorithm should run in $O(n)$ time.

2.c) (15 points) Write an SML function `inter` to compute the intersection of two sets. Your algorithm should run in $O(n)$ time.

Question 3: Sometimes, we would like to perform arithmetic operations on numbers which are larger than what can be handled by the microprocessor's built-in hardware. To implement RSA, for example, we are required to compute addition, multiplication and modulus of large numbers.

In this question, you will implement a system to perform arithmetic operations on arbitrarily large positive numbers. An n-digit decimal number

$d_1 \ldots d_n$ will be represented by the list `dn::...::(d2::(d1::nil))`, so that the top element of the list is the least significant digit. You may want to use the following functions to convert between strings and bignums.

```
fun btos(L)=
    let
        fun listtostring(nil) = ""
            | listtostring(x::xt) =
                    Int.toString(x) ^ listtostring(xt);
    in
        listtostring(rev(L))
    end;

fun stob(s:string) =
    let
        fun f(nil: char list):int list = nil
           |f(x::xt) = (ord(x) - ord(#"0"))::f(xt);
    in
        f(rev(explode(s)))
    end;

printBignum(L) = print(btos(L));
```

3.a) (25 points) Write a function `add` which takes as input two bignums and computes their sum (essentially, you want to implement the grade-school algorithm for addition).

3.b) (25 points) Using `add`, write a function `mult` which takes as input two bignums and computes their product.

You should be able to write `add` and `sum` with about 10-12 lines of SML code per function.