Daniel Leeds, 15-212 R01, August 29, 2007

Administrative Stuff:

Textbooks: from bookstore, barnesandnoble.com, amazon.com
AFS directory: /afs/andrew/course/15/212/*username/assn#/assn#.sml*
Web site: http://www.cmu.edu/blackboard

Running Standard ML:

Log into Andrew (*e.g.*, ssh username@linux.andrew.cmu.edu)

```
rlwrap sml       OR      /usr/contributed/bin/rlwrap /usr/local/bin/sml
```

(Ctrl-D to exit)

emacs file.sml -> Ctrl-C Ctrl-B sml

Basis Library:

http://www.smlnj.org/doc/basis/pages/sml-std-basis.html

Some ML code:

```
val x = 4;                (* binds the value 4 to x, sets the type of x to int *)
val z = 4.3 : real;       (* binds the value 4.3 to z, we explicitly specify type of z to be real *)
fun times2 y = y*2;       (* binds the function to times2, sets the type of times2 to int -> int *)
times2 x;                 (* applies times2 to value of x, binds int result to it *)
use "file.sml";           (* runs file.sml *)
fun mystery 0 = 1         (* you tell me… *)
    | mystery n = n * mystery(n-1);
```

Sample error messages:

```
stdIn:10.1-10.9 Error: operator and operand don't agree [tycon mismatch]
  operator domain: int
  operand:         real
  in expression:
    times2 z;
```

```
stdIn:8.4-8.19 Error: expression doesn't match constraint [literal]
    expression:    int
    constraint:    real
    in expression:
        42: real
```

```
stdIn:11.1-11.7 Error: unbound variable or constructor: Times2 z;
```

```
stdIn:2.2-8.20 Error: match redundant
              n => …
    -->       0 => …
```

```
stdIn:2.2-6.3 Warning:match nonexhaustive
              0 => …
val mystery2 = fn : int -> int
```

```
uncaught exception Math [nonexhaustive match failure]
  raised at: stdIn:6.3
```

Polymorphism:

```
    List.length : 'a list -> int
    val xs = [1, 2, 3, 4] : int list;
    val ys = [1.1, 2.3, 3.14159] : real list;
```

List.length will work on both lists! Type 'a list indicates a list of any type.

Exercises:

Implement factorial.

Implement Fibonacci ($F_n = F_{n-1} + F_{n-2}$)

Implement Euclid's GCD: (assuming m<n, gcd(0,n) = n and gcd(m,n) = gcd(n mod m, n))

Implement List.length

(Good and bad) ML expressions:

```
val x = 4;
val q = “hello”;
x+q;
val y = 5.6;
x+y;
y+2.4;
x+3;
it*3;
mystery(4);
mystery(4.5);
mystery(-3);
mystery(~3);
```

```
val goose = SOME 4.5;
```