**A FIRST TASTE OF F#**

**by Geoff Smith**

// Classic recursion for calculating Fibonacci numbers:

let rec fib1 = function

| 0 -> 0I

| 1 -> 1I

| n -> fib1(n-1) + fib1(n-2)

// The following code corrects the displaying of bigints under .NET 4.0:

fsi.AddPrinter (fun (x:bigint) -> string x + "I")

/// polymorphic exponentiation

let mk\_expon times one =

let rec expon x n =

if n = 0 then one

elif n%2 = 0 then expon (times x x) (n/2)

else times x (expon x (n-1))

expon

// Note that mk\_expon returns an exponentiation function.

/// int exponentiation

let expon1 = mk\_expon (\*) 1

/// string exponentiation

let expon2 = mk\_expon (+) ""

let matmult (a : bigint, b, c, d) (e, f, g, h) =

(a\*e+b\*g, a\*f+b\*h, c\*e+d\*g, c\*f+d\*h)

let identity = (1I,0I,0I,1I)

/// 2-by-2 matrix exponentiation

let expon3 = mk\_expon matmult identity

// It turns out that Fibonacci numbers can be

// computed via matrix exponentiation!

//

// / 1 1 \ n / fib(n+1) fib n \

// \ 1 0 / = \ fib n fib(n-1) /

let fib2 n =

let (a, b, c, d) = expon3 (1I, 1I, 1I, 0I) n

c

// Here's a taste of F#'s libraries:

open System.Windows.Forms

let form = new Form(Visible=true, TopMost=true, Text="Welcome to F#")

let textB = new RichTextBox(Dock=DockStyle.Fill, Text="Here is some initial text")

form.Controls.Add(textB)

open System.IO

open System.Net

let http (url:string) =

let req = System.Net.WebRequest.Create(url)

let resp = req.GetResponse()

let stream = resp.GetResponseStream()

let reader = new StreamReader(stream)

let html = reader.ReadToEnd()

resp.Close()

html

let geoff = http "http://www.cis.fiu.edu/~smithg"

textB.Text <- geoff