HW #4

Complete By: Wednesday Feb 5th @ 9:00pm
Policy: Individual work only, late work not accepted
Submission: answers to exercises, hand-written or typed

Learning F#

The concepts of functional programming are discussed in chapter 10 of the text. However, we are going to program in F#, which we'll discuss in class to some degree. However, you expected to fill in the gaps on your own. Microsoft's main site for learning F# is here:

http://www.tryfsharp.org/Learn

Interactive tutorials are also available here: http://fsharp.org/about/learning.html. For starters, I recommend we all use http://ideone.com/, though you can also use Visual Studio and create a new Visual F# project (select *Windows*, then *F# Application*); Visual Studio will offer intellisense and type feedback that http://ideone.com is unable to do.

Assignment

Your assignment is to write the following functions. These functions should work on lists of any type, but in some cases F# may not be able to infer the type[1]. You will know that F# is having problems when you get the error message "*Lookup on object of indeterminate type … A type annotation may be needed …*". The solution is to add the required type annotation, which for a generic list will be **'a list**. For example, suppose you are trying to define a recursive function F that takes a list as a parameter, and you need to add a type annotation to define that the list is a generic list. Then the definition of F will look like this:

```
let rec F(L:'a list) =
```

If F takes additional parameters, they follow the type annotation like this:

```
let rec F(L:'a list, N) =
```

Add additional type annotations as necessary, but usually the first annotation is sufficient for F# to automatically figure out the rest. Here is the list of functions to write for the homework:

- **4.1**: write a generic, tail-recursive **length** function.
- **4.2**: Write a function **median(L)** that takes a sorted list of integers and returns the median value for that list. If the list is empty, return 0. If the list contains an odd number of elements, return the middle

[1] For more info type inference in F#, see http://fsharpforfunandprofit.com/posts/type-inference/ .

element. If the list contains an even number of elements, return the average of the middle 2 elements.

- **4.3**: Write a function **equal(L1, L2)** that returns true if two lists L1 and L2 are equivalent, i.e. contain the same elements in the same sequence. Otherwise it returns false. Your function should work safely even if L1 and L2 contain different numbers of element. Note: even though the "=" operator in F# can be used to compare lists directly, write your functions recursively, using "=" on an element by element basis. Good practice. [*Hint: in F# you are allowed to match multiple conditions at the same time, i.e. match L1, L2 with …*]

- **4.4**: Write a function called **vectorMultiply(L1, L2)** that multiplies the corresponding elements of two lists (vectors). Assume the lists contain integers, and are of the same length. For example, **vectorMultiply**([1; 2; 3], [4; 5; 6]) returns [4; 10; 18]. An empty list * an empty list is the empty list.

Each function should be written from scratch, do not use F# existing functions other than the necessary list operations we have been using in class: *Head*, *Tail*, concatenation, etc. However, feel free to write any helper functions you might need; be sure to include those functions in your electronic submission.

Electronic Submission

Using Blackboard (http://uic.blackboard.com/), submit your F# functions under the assignment "HW4". Submit a text file; if you worked on http://ideone.com, please copy-paste your code into a text file — as text, not as HTML — and submit that file. Please make sure your submission contains a header comment with your name and such:

```
//
// F# functions for <<fill in the blanks>>
//
// <<YOUR NAME HERE>>
// U. of Illinois, Chicago
// CS341, Spring 2014
// Homework 4
//
```

You may submit as many times as you want before the due date, but we grade the last version submitted.

Policy

Late work is not accepted. All work is to be done individually — group work is not allowed. Academic dishonesty is unacceptable, and all parties involved will be immediately subject to the official academic integrity review process. The University's policy is quite clear, and can be read here: http://www.uic.edu/depts/dos/studentconduct.html. In particular, note that you are guilty of academic dishonesty if you extend or receive any kind of unauthorized assistance. Examples of academic dishonesty include emailing your program to another student, copying-pasting code from the internet, working in a group on a homework assignment, and allowing a tutor, TA, or another individual to write an answer for you.