Homework 13: Due Friday, April 21

Problem 1: Suppose that we code rationals numbers in ML as ordered pairs or arbitrary precision integers. That is, suppose that we introduce the type synonym

$$\mathsf{type\ rational} = \mathsf{IntInf.int} * \mathsf{IntInf.int}.$$

Thus, we view $(1, 2)$ as coding the fraction $\frac{1}{2}$. Write an ML function addRat that takes two two inputs, assumed to be of type rational, and outputs the rational that is the sum of the two inputs. Your answer should be in lowest terms, i.e. you should not be able to simply the fraction further by dividing out a common value. For example, $\mathsf{addRat}((1, 6), (1, 10))$ should produce $(4, 15)$. Moreover, your function should operate very quickly on large inputs.

Problem 2: Recall from class that for all $a, b \in \mathbb{N}$ with $b \neq 0$, there exist $q, r \in \mathbb{N}$ with $a = qb + r$ and $r < b$. In ML, given $a, b \in \mathbb{N}$ with $b \neq 0$, the value q is computed by a div b, and the value r is computed by a mod b. In this problem, you will write two versions of div and mod (packaged together) using recursion. That is, you should write two versions of a function that takes two inputs a and b of type int, where $a \geq 0$ and $b > 0$, and produces the ordered pair $(q, r) \in \mathbb{N} \times \mathbb{N}$, with $r < b$, and $a = qb + r$.
a. Using only basic arithmetic operations on integers (so neither div nor mod), write an implementation of the above function that makes a recursive call to $a - 1$.
b. Using only basic arithmetic operations on integers (so neither div nor mod), write an implementation of the above function that makes a recursive call to $a - b$.
Note: The first function corresponds to regular mathematical induction, while the second corresponds to strong induction.

Problem 3: Using the built-in addition function on IntInf.int, write a multiplication function multiply that works on two nonnegative values from IntInf.int. Your function should operate very quickly on large inputs (so you should not just use repeated addition). You may use both div and mod by 2 in your solution, as these can be performed quickly if your numbers are stored in binary. Of course, you should not use the built-in multiplication function.

Problem 4: Recall the following ML function from class:

$$\begin{aligned} &\mathsf{fun}\ \ \mathsf{h}(0, \mathsf{a}, \mathsf{b}) = \mathsf{a} \\ &\quad\ \ | \ \mathsf{h}(\mathsf{n}, \mathsf{a}, \mathsf{b}) = \mathsf{h}(\mathsf{n} - 1, \mathsf{b}, \mathsf{a} + \mathsf{b}). \end{aligned}$$

Show that for all $\mathsf{k}, \mathsf{n} \in \mathbb{N}$, we have that $\mathsf{h}(\mathsf{n}, \mathsf{f_k}, \mathsf{f_{k+1}}) = \mathsf{f_{k+n}}$. Using the special case where $\mathsf{k} = 0$, it follows that $\mathsf{h}(\mathsf{n}, \mathsf{f_0}, \mathsf{f_1}) = \mathsf{f_n}$ for all $\mathsf{n} \in \mathbb{N}$.
Hint: Let $P(\mathsf{n})$ be the predicate "For all $\mathsf{k} \in \mathbb{N}$, we have $\mathsf{h}(\mathsf{n}, \mathsf{f_k}, \mathsf{f_{k+1}}) = \mathsf{f_{n+k}}$", and prove "For all $\mathsf{n} \in \mathbb{N}$, we have $P(\mathsf{n})$" by induction.

Interlude: We saw how fast the Euclidean Algorithm ran in class. However, it is not obvious why the algorithm terminates so quickly. In the next problem, we work to understand the theory behind the speed. Let $a, b \in \mathbb{N}$ with $b \neq 0$. Fix $q, r \in \mathbb{N}$ with $a = qb + r$ and $r < b$. Notice that after one step of the algorithm, the new second argument r may not be much smaller than the original second argument b. For example, if $a = 77$ and $b = 26$, then we have $q = 2$ and $r = 25$. However, it turns out that after *two* steps of the Euclidean Algorithm, the new second argument will be at most half the size of the original second argument. This is what we will prove.

Problem 5: Let $a, b \in \mathbb{N}$ with $b \neq 0$. In the first step of the algorithm, we write $a = qb + r$ where $q, r \in \mathbb{N}$ and $0 < r < b$ (we can assume that $r \neq 0$ because otherwise the algorithm stops at the next step). In the next step of the algorithm, we write $b = pr + s$ where $s, r \in \mathbb{N}$ and $s < r$. Show that $s < \frac{b}{2}$. Thus, the second argument is at least cut in half after two steps of the Euclidean algorithm, which is just like our function power from class.
Hint: It might help to break the problem into cases based on how large r is.

Problem 6: In class, we discussed the fact that for all $a, b \in \mathbb{N}$, there exist $k, \ell \in \mathbb{Z}$ with $ka + \ell b = \gcd(a, b)$. In this problem, you will write an ML function that calculates such a pair (k, ℓ) on input (a, b).
a. Let $a, q, b, r \in \mathbb{N}$ with $a = qb + r$ and $r < b$. Let $k, \ell \in \mathbb{Z}$ be such that $kb + \ell r = \gcd(b, r)$. Show that $\ell a + (k - q\ell)b = \gcd(a, b)$.
b. Using part a, write a recursive ML function that takes two inputs $a, b \in \mathbb{N}$ (each assumed to be of type IntInf.int), and produces an ordered pair $(k, \ell) \in \mathbb{Z}^2$ (of type IntInf.int*IntInf.int) such that $ka + \ell b = \gcd(a, b)$. For example, with inputs 525 and 182, your function could produce $(-9, 26)$. Your program should work extremely quickly on very large inputs.