CS 337 **Solution to Test 3** **5/12/08**

1. (Recursion and Induction)

(a)
```
take                    :: Int -> [a] -> [a]
take 0 xs                =  []
take n []                =  []
take n (x:xs)            =  x : take (n-1) xs
```

(b)
```
drop                    :: Int -> [a] -> [a]
drop 0 xs               =  xs
drop n []               =  []
drop n (x:xs)           =  drop (n-1) xs
```

(c)
```
suml xs = foldr (+) 0 xs  -- suml computes the sum of a list
rowsum xss = map suml xss -- compute row sum of the matrix
matsum xss = suml (rowsum xss) -- compute sum of the whole matrix

listsum [] [] = [] -- listsum sums two lists element by element
listsum (x:xs) (y:ys) = (x+y): (listsum xs ys)

colsum [xs] = xs -- compute column sum of the matrix
colsum (xs: xss) = listsum xs (colsum xss)
```

(d)
```
rank 1 [x] = x
rank i (x:xs)
 | i <= n    = rank i lh
 | i == n+1   = x
 | i> n+1    = rank (i-n-1) rh
               where
                (lh,rh) = partition x xs
                n       = length(lh)
```

(e)
```
ac     :: [String] -> [String]
ac [] = []
ac xss = acc [] xss
acc ys [] = []
acc ys (xs:xss) = (zs): (acc zs xss)
                  where zs = ys ++ xs
```

(f)

```
-- (dist xs ys) is  True iff Hamming distance between xs and ys is 1.
dist [] [] = False
dist (x:xs) (y:ys)
 | x == y = dist xs ys
 | x /= y = xs == ys

-- hamming xss yss is True iff
-- the hamming distance between each pair of
-- corresponding elements of xss and yss is 1.
hamming [] [] = True
hamming (xs:xss) (ys:yss) = (dist xs ys) && (hamming xss yss)

-- adj xss is True iff all adjacent pairs of strings in xss
-- have Hamming distance of 1.
adj     :: [String] -> Bool
adj xss  = hamming xss (right_rotate xss)
```

2. (String Searching; 20 points)

(a) We have to show that $c^j(Z_n 0) = Z_{n-j}0$. Proof is by induction on j.

• $j = 1$: We have to show that $c(Z_n 0) = Z_{n-1}0$, for $n \geq 1$. This follows from: (1) $Z_{n-1}0$ is both a prefix and a suffix of $Z_n 0$, and (2) the only longer proper prefix of $Z_n 0$ is Z_n, which is not a suffix.

• $j + 1$: Assume $c^j(Z_n 0) = Z_{n-j}0$, where $j < n$. We show that $c^{j+1}(Z_n 0) = Z_{n-j-1}0$.

$$
\begin{array}{ll}
 & c^{j+1}(Z_n 0) \\
= & \{\text{definition of } c^{j+1}\} \\
 & c(c^j(Z_n 0)) \\
= & \{\text{induction hypothesis}\} \\
 & c(Z_{n-j}0) \\
= & \{\text{from the first proof}\} \\
 & Z_{n-j-1}0
\end{array}
$$

(b) Suppose $p[0..k]$ is the core. From the definition of core,

$p[0..k] = p[12 - k..12]$. Hence,
$p[i] = p[12 - k + i]$

Letting $i = 3$ and $12 - k + i = 5$, we get $k = 10$. That is, if $k = 10$, $p[3] = p[5]$. Since $p[3] \neq p[5]$, $k \neq 10$. Similarly, using the fact that $p[3] \neq p[9]$, we get $k \neq 6$. And, from $p[5] \neq p[9]$, we get $k \neq 8$. Thus, the length of the core is not 6, 8 or 10.

(c) Choose q and make one pass over the genome sequence to compute $val(p) = p \bmod q$ for all substrings p of length 20 and less. Enter each $val(p)$ in a hash table along with a pointer to the part of the genome sequence where p is a substring. To match a pattern r of length less than or equal to 20, compute $val(r)$ and look it up in the hash table. For all occurrences of this value, match the pattern against the corresponding substring in the genome sequence.

For a pattern of length more than 20, let r be its prefix of length 20. Follow the same steps, as above.

3. (Relational Algebra; 10 points)

   (a) The names of theatres which are showing PG movies in which Will Smith is acting, is given by

   $$\pi_{Theatre}(\sigma_{p \wedge q}\langle R \bowtie T\rangle)$$

   where

   p is $Actor =$ Will Smith
   q is $Rating =$ PG

   (b) Consider relations R and S in Table 1. Each relation has two attributes, and just one tuple.

	Title	Actor
R	Men in Black	Will Smith
S	Men in Black	Tommy Lee Jones

Table 1: Two relations

Take attribute a to be Title. Now, $R \cap S$ is empty, so $\pi_a(R \cap S)$ is also empty. However, $\pi_a(R)$ and $\pi_a(S)$ both have a single row (and single column) with the entry “Men in Black”. So, $\pi_a(R) \cap \pi_a(S)$ has one row.