

## CS 405/505 ASSIGNMENT #5

Due Thursday, October 18, 2007

1. Roman numerals use the following symbols:

I (one)  
V (five)  
X (ten)  
L (fifty)  
C (one hundred)  
D (five hundred)  
M (one thousand)

These are usually written in linear order (e.g., I, II, III) with smaller numerals always following larger numerals except when they are to be subtracted (e.g., IV means 5 - 1). Subtractions are used instead of writing four consecutive occurrences of the same symbol (i.e., IV should be used instead of IIII, MCM instead of MDCCCC, etc.). Only symbols which are powers of 10 may be used in subtractions (i.e., I, X, C, and M). Furthermore, subtraction rules require that a symbol representing  $10^x$  may not precede any symbol larger than  $10^{x+1}$  (e.g., IC is not permitted to represent 99 but XC is 90). Write a LISP function (`romanToDecimal RomanNumeral`) which takes a Roman numeral as a list of digits and returns the decimal equivalent if the Roman numeral is well formed and `error` otherwise. Test your function as follows:

- (a) (`romanToDecimal '(C D V)`)  $\Rightarrow$  405
- (b) (`romanToDecimal '(M M V I I)`)  $\Rightarrow$  2007
- (c) (`romanToDecimal '(M M M C M X C I X)`)  $\Rightarrow$  3999
- (d) (`romanToDecimal '(M I M)`)  $\Rightarrow$  `error`

2. DNA is a two-stranded molecule. Each strand is a polynucleotide composed of A (adenosine), T (thymidine), C (cytidine), and G (guanosine) residues. The two strands of DNA run antiparallel (i.e., at each nucleotide residue along the double-stranded DNA molecule, the nucleotides are complementary). That is, A forms two hydrogen-bonds with T; C forms three hydrogen bonds with G. In most cases the two-stranded, antiparallel, complementary DNA molecule folds to form a helical structure, which resembles a spiral staircase. This is the reason why DNA has been referred to as the “Double Helix.” An example of two complementary strands of DNA would be:

```
ATGGAATTCTCGCTC  
TACCTTAAGAGCGAG
```

This genetic sequencing problem is to find complementary matching strands of DNA. Write a LISP function (`matchDNA strand1 strand2`) that takes two strands as lists of nucleotides, and returns: 1) the beginning part of the second strand which does not match, 2) the part of the second strand which the first string matches, and 3) the remaining part of the second strand which does not match. Note that if there is no match, the second and third components will be null. If there are multiple matching strings, returning only the first match is sufficient. Test your function as follows:

- (a)  $(\text{matchDNA } '(A\ T\ G\ C) \ '(A\ T\ G\ C\ A\ T\ A\ C\ G\ A)) \Rightarrow ((A\ T\ G\ C\ A) (T\ A\ C\ G) (A))$
- (b)  $(\text{matchDNA } '(A) \ '(A\ T\ G\ C\ A\ T)) \Rightarrow ((A) (T) (G\ C\ A\ T))$
- (c)  $(\text{matchDNA } '(C\ G) \ '(A\ T\ G\ C)) \Rightarrow ((A\ T) (G\ C) \text{ nil})$
- (d)  $(\text{matchDNA } '(G\ A\ A\ T\ T\ C) \ '(T\ G\ G\ A\ A\ T\ T\ C\ T)) \Rightarrow$   
 $((T\ G\ G\ A\ A\ T\ T\ C\ T) \text{ nil nil})$

3. Consider the Core program below:

```
input a; input b;
k := 1;
while (b > 0) loop
  b := b - 1;
  k := k * a;
end loop;
output k;
```

Assuming that the input file contains the integers 4 and 3 (i.e., the list  $\langle 4, 3 \rangle$ ), use the denotational semantics of Core to trace the interpretation of the program.