Wright State University
Department of Computer Science and Engineering

CS 784 Spring 2011 Prasad

Midterm (30 pts)

1 Recursive Definition (6 pts)

Define a function `countSymbols` that takes a nested list of numbers and symbols as input, and returns the count of all symbols in the input list.

```
(countSymbols '(a)) returns 1
(countSymbols '(2 56 x (1 y))) returns 2
(countSymbols '(((a)) -2 (2 (ab b) (-1 0 1))))) returns 3
```

2 Higher-Order Function (5 + 1 pts)

```
(define (reduce f id lis)
        (if (null? lis) id
            (f (car lis) (reduce f id (cdr lis)))))
```

The length of a list can be defined in terms of `reduce` (as opposed to using a recursive definition from scratch) as

`(define (mylength lis) (reduce  (lambda (x n) (+ 1 n)) 0 lis)).`

Define the list function `mymap` (similar to map) that takes a unary function `uf` and a list `lis` in terms of `reduce`, that is, by determining the corresponding f and id in

`(mymap uf lis) = (reduce` f id `lis)`,

Recall that mymap returns a list resulting from calling the function on every element in the input list such as `(mymap (lambda(x) (* 2 x)) '(1 2 3)) = (2 4 6)`.

3 Metaprogramming (4 pts)

Explain why is it not possible to simulate **if-then-else** construct as a function in an interpreter that supports function application, to justify the need for a special form in the interpreter to deal with it.

4 ADT Specification (4 + 1 + 3*3 pts)

A *sequence* is an ordered collection of values of the same type, possibly with duplicates. You are required to specify the generic ADT **Seq** that supports the following operations: `empty`, `insert`, `isEmpty`, `odd`, `even`, and `alternate`. Informally,

- `empty`: the empty sequence.
- `insert`: Takes a value and a sequence as input, and yields the sequence resulting from introducing one occurrence of the value at the beginning of the sequence.
- `isEmpty`: Checks to see if a sequence is empty.
- `odd`: Takes a sequence as input, and yields the sequence resulting from retaining only odd-positioned values from the sequence. (That is, odd([]) = [], odd([1,11,2,22,3,33]) = [11,22,33], odd([a,b,c]) = [b], etc.)
- `even`: Takes a sequence as input, and yields the sequence resulting from retaining only even-positioned values from the sequence. (That is, even([]) = [], even([1,11,2,22,3,33]) = [1,2,3], even([a,b,c]) = [a,c], etc.)
- `alternate`: Takes two sequences as input, and yields the sequence resulting from interleaving them with first sequence values occupying even-positions and second sequence values occupying odd-positions. If the inputs are of unequal length, ignore the extra suffix. (That is, alternate([1,2,3], [11,22,33]) = [1,11,2,22,3,33], alternate([a,b,c], [aa]) = [a,aa], alternate([a,b,c], []) = [], etc.)

1. Specify the signatures and classify the aforementioned operations on ADT **Seq**.
2. Give an algebraic specification of the semantics of ADT **Seq** clarifying any ambiguity in the informal description.