TÖL304G
Forritunarmál
Hópverkefni 7

1. Program the function mapreduce in Morpho. The function must be tail recursive (or use a loop) and should compute from left to right.

```
;;; Use:     mapreduce(f,op,u,x)
;;; Pre:     f is a function 'a -> 'b
;;;          op is a function 'c -> 'b -> 'c
;;;          x=[x1;x2;...;xN] is a list of values of type 'a
;;;          u is a value of type 'c
;;; Value:   u+f(x1)+f(x2)+...+f(xN)
;;;          computed from left to right,
;;;          where p+q = (op p q).
;;;          This is a value of type 'c.
rec fun mapreduce(f,op,u,x)
{
    if(x==[])
    {
        return u;
    }
    else
    {
        return mapreduce(f, op, (op(u,f(head(x)))), tail(x));
    }
};

writeln(mapreduce(fun(x){1+x},fun(a,b){a+b},0,[1,2,3]));
writeln(mapreduce(fun(x){x*x},fun(a,b){a*b},1,[1,2,3]));
9
36
```

2. Program the function fromTo in Morpho.

```
;;; Use:     fromTo(i,j)
;;; Pre:     i and j are integers, i<=j
;;; Value:   The list [i,i+1,...,j-1].
rec fun fromTo(i,j)
{
    if(i==j)
    {
        return [];
    }
    else
    {
        return i : (fromTo((i+1),j));
    }
};

writeln(fromTo(3,6));
[3,4,5]
```

3. Program the function insertAt in Morpho.

```
;;; Use:     insertAt(x,i,z)
;;; Pre:     x=[x1,x2,...,xN] is a list of values
;;;          of some type 'a. z is a value of type 'a.
;;;          i is an integer, 0 <= i <= N,
;;;          where N is the length of the list x.
;;; Value:   The list [x1,x2,...,x_i,z,x_i+1,...,xN],
;;;          i.e. the list that results from
;;;          inserting the value z into the
;;;          list x right after the first i values.
rec fun insertAt(x,i,z)
{
    if(i<1)
    {
        return z:x;
    }
    else
    {
        return head(x) : insertAt(tail(x),(i-1),z);
    }
};

writeln(insertAt([1,2,3,4,5,6,7,8], 4, 9));
[1,2,3,4,9,5,6,7,8]
```

4. Program the function permutations in Morpho.

First we need the function extendPermutation:

```
;;; Use:     extendPermutation(n,z)
;;; Pre:     n >= 0 is an integer.
;;;          z is some permutation of [1,2,...,n-1].
;;; Value:   The list of all the lists that result from
;;;          inserting the number n into the list z at
;;;          some position, from start to end.
rec fun extendPermutation(n,z)
{
    return map((fun(x){insertAt(z,x,n)}),(fromTo(0,n)));
};

writeln(extendPermutation(3,[1,2,3]));
[[3,1,2,3],[1,3,2,3],[1,2,3,3]]
```

Then we can do permutations:

```
;;; Use:     permutations(n)
;;; Pre:     n>=0 is an integer.
;;; Value:   The list of all permutations of the list
;;;          [1,2,...,n].
rec fun permutations(n)
{
    if(n==0)
    {
        return [[]];
    }
    else
    {
        return mapreduce(
            fun(z){extendPermutation(n,z)},
            fun(a,b){append(a,b)},
            [],
            permutations(n-1));
    }
};

writeln(permutations(3));
[[3,2,1],[2,3,1],[2,1,3],[3,1,2],[1,3,2],[1,2,3]]
```