Introduction to Functional Programming 2004
Paper 13 Question 10 (GMB)

(*a*)

```
datatype 'a tree  = Lf
                  | Node of 'a * 'a tree * 'a tree;

fun walk ([],[]) = []
  | walk (x,[]) = x
  | walk ([],x) = x
  | walk (x::xs,y::ys) = (x@y)::walk(xs,ys);

fun flatten [] = []
  | flatten (x::xs) = x@(flatten(xs));

fun tbreadth Lf = []
  | tbreadth (Node(n,l,r)) = let val l1 = tbreadth l
                                 val r1 = tbreadth r
                             in
                                 [n] :: walk (l1,r1)
                             end;

fun breadth t = flatten(tbreadth(t));
```

(*c*)

```
datatype 'a seq   = Nil
                  | Cons of 'a * (unit -> 'a seq);

datatype 'a ltree = LLf
                  | LNode of 'a * (unit -> 'a ltree)
                                * (unit -> 'a ltree);
fun lappend (Nil,s) = s
  | lappend (Cons(h,t),s) = Cons(h,
                                 fn ()=>lappend(t(),s));

fun linorder  LLf           = Nil
  | linorder (LNode(n,l,r)) = lappend(linorder(l()),
                                 Cons(n,fn ()=>linorder(r())));
```