McMaster University
Department of Computing and Software
Dr. W. Kahl

CAS 781 — Functional Programming

2003-01-23

Understand the following functions:

```
data Tree a = ET | Br (Tree a) a (Tree a) deriving Show

mktree bs = foldt Br ET bs

foldt :: (a -> b -> a -> a) -> a -> [b] -> a
foldt f a bs = h (repeat a,bs)
 where
  h (a:_ , [] ) = a
  h xs = h (fold xs)
  fold (a1:a2:as , b1:b2:bs) = pupd ((f a1 b1 a2) :) (b2 :) (fold (as,bs))
  fold (a1:a2:as , [b]     ) = (f a1 b a2 : as, [])
  fold p = p

pupd f g (x,y) = (f x, g y)

-- specialised instance:

mkT    :: [b] -> Tree b
mkT bs = mkT1 (repeat ET, bs)

mkT1 (a:_ , [] )          = a
mkT1 p                    = mkT1 (collect p)

-- collect (a1:a2:as , b:bs) = pupd1 ((Br a1 b a2) :) (shift (as,bs))
collect (a1:a2:as , b1:b2:bs) = pupd ((Br a1 b1 a2) :) (b2 :) (collect (as,bs))
collect (a1:a2:as , [b]     ) = (Br a1 b a2 : as, [])
collect p                 = p

shift (as , b:bs)         = pupd2 (b:) (collect (as, bs))
shift p                   = p

pupd1 f (x,y) = (f x, y)
pupd2 g (x,y) = (x, g y)
```

- As a start, simulate evaluation of `mkT [1,2,3,4,5]`.
- Write a “normal” `showTree` function (with linear complexity).
- Write a function to display trees on the screen via indentation.
- Use `mkT` to implement `mergesort`.