fmal v10

hir12

November 2022

```haskell
{-
Notkun:    partition a b x
Fyrir:    a og b eru samanburðarföll.
        x er listi af gildum af sömu týpu.1111
Gildi:  3 listar sem að innihalda skiptu gildi x.
        listarnir eru skiptir í hluta eftir gildi a og b.
        fyrsti listinn inniheldur allt vinstra megin við a.
        annar listinn inniheldur allt á milli a og b.
        þriðji listinn inniheldur allt hægra megin við b.
-}
partition :: (a -> Bool) -> (a -> Bool) -> [a] -> ([a],[a],[a])
partition a b [] = ([],[],[])
partition a b (x:xs) =
    let
        (p1,p2,p3) = partition a b xs
    in
        if a x then
            (x:p1,p2,p3)
        else if b x then
            (p1,p2,x:p3)
        else
            (p1,x:p2,p3)

{-
Notkun: quicksort comp x
Fyrir:  comp er samanburðarvirki sem fallið notar
            (<, <=) er til þess að fá lista af vaxandi gildum, (1,2,3,4,...,n-1,n)
            (>, >=) er til þess að fá lista af minnkandi gildum, (n,n-1,...,4,3,2,1)
        x er listi sem á að raða.
Gildi:  Skilar röðuðum lista sem er annaðhvort vaxandi eða minnkandi.
-}
quicksort :: (a -> a -> Bool) -> [a] -> [a]
quicksort (<) [] = []
quicksort (<) [x] = [x]
quicksort (<) (x:xs) =
```

```
    let
        (p1,p2,p3) = partition (<x) (x<) xs
    in
        quicksort (<) p1 ++ x:p2 ++ quicksort (<) p3

{-
Notkun: split x
Fyrir:  x er listi talna sem á að skipta í tvennt.
Gildi:  Skilar tveimur listum sem að voru upprunanlega einn listi.
-}
split :: [a] -> ([a],[a])
split [] = ([],[])
split [x] = ([x],[])
split (x1:x2:xs) =
    let
        (p1,p2) = split xs
    in
        (x1:p1,x2:p2)

{-
Notkun: merge comp x y
Fyrir:  x og y eru listar af tölum sem geta verið mislangir.
        comp er samanburðarvirki sem að fallið notar.
            (<), (<=), (>=), (>) samanburðarvirkjar sem að sjá um hvernig er sameinað.
Gildi:  Skilar sameinuðum lista x, y þar sem comp stjórnar hvernig er sameinað.
-}
merge :: (a -> a -> Bool) -> [a] -> [a] -> [a]
merge (<) [] x = x
merge (<) x [] = x
merge (<) (x:xs) (y:ys) =
    if x<y then
        x : (merge (<) xs (y:ys))
    else
        y : (merge (<) (x:xs) ys)

{-
Notkun: mergesort comp x
Fyrir:  comp er samanburðarvirki, x er listi.
            (<), (<=), (>=), (>) samanburðarvirkjar sem að ráða hver röðin á listanum er.
Gildi:  Skilar sortuðum lista þar sem comp segir hvort listinn sé vaxandi eða minnkandi.
-}
mergesort :: (a -> a -> Bool) -> [a] -> [a]
mergesort (<) [] = []
mergesort (<) [x] = [x]
mergesort (<) xs =
    let
```

```
        (p1,p2) = split xs
    in
       merge (<) (mergesort (<) p1) (mergesort (<) p2)

{-
Notkun: repeatList n x
Fyrir:  x is some list of type [a],
        n is an Int, n>=0.
Gildi:  The list that results from concatenating
        x with itself n times.
Note:   Do not change this implementation.
-}
repeatList :: Int -> [a] -> [a]
repeatList 0 x = []
repeatList n x = x++(repeatList (n-1) x)
{-
Notkun: main
Fyrir:  Nothing.
Post:   Some test results have been written
        to stdout.
Note:   You may modify this function if you
        wish to do more tests.
-}
main :: IO ()
main = do
    let s = repeatList 10 "abcde"
    print s
    print (quicksort (>) s)
    print (mergesort (>) s)
```

```
PS C:\Users\Hákon Ingi\VS code\haskellur> .\e10.exe
"abcdeabcdeabcdeabcdeabcdeabcdeabcdeabcdeabcdeabcde"
"eeeeeeeeeeddddddddddccccccccccbbbbbbbbbbaaaaaaaaaa"
"eeeeeeeeeeddddddddddccccccccccbbbbbbbbbbaaaaaaaaaa"
PS C:\Users\Hákon Ingi\VS code\haskellur>
```