99 questions/Solutions/84

From HaskellWiki

< 99 questions | Solutions

Create an undirected-graph:

```
  graph = mkGraph False (1,5)
            [(1,2,12),(1,3,34),(1,5,78),(2,4,55),
             (2,5,32),(3,4,61),(3,5,44),(4,5,93)]
```

False means undirected

Use prim algorithm to find the minimal spanning tree:

```
  prim graph
```

Output:

```
  [(55,2,4),(34,1,3),(32,2,5),(12,1,2)]
```

```haskell
module Prim where

import Data.List
import Array

type Graph n w = Array n [(n,w)]

mkGraph dir bnds es =
    accumArray (\xs x -> x:xs) [] bnds
               ([(x1,(x2,w)) | (x1,x2,w) <- es] ++
               if dir then []
               else [(x2,(x1,w)) | (x1,x2,w) <- es, x1 /= x2])

adjacent g v = map fst (g!v)

nodes g = indices g

edgeIn g (x,y) = elem y (adjacent g x)

weight x y g = head [c | (a,c) <- g!x, a == y]

edgesD g = [(v1,v2,w) | v1 <- nodes g, (v2,w) <- g!v1]
edgesU g = [(v1,v2,w) | v1 <- nodes g, (v2,w) <- g!v1, v1 < v2]

prim g = prim' [n] ns []
    where (n:ns) = nodes g
          es = edgesU g
```

```
        prim' t [] mst = mst
        prim' t r mst = let e@(c,u',v') = minimum
                                          [(c,u,v) | (u,v,c) <- es,
                                                     elem u t,
                                                     elem v r]
                        in  prim' (v':t) (delete v' r) (e:mst)
```
