```
insert :: Ord a => a -> [a] -> [a]
insert x [] = [x]
insert x (y:ys) = if x <= y then x: (y:ys)  else y : insert x ys

isort :: [Int]-> [Int]
isort [] = []
isort (x:xs) = insert (x) (isort xs)

qsort :: [Int] -> [Int]
qsort [] = []
qsort (x:xs) = qsort [y | y <- xs, y <= x]
             ++ [x]
             ++ qsort [y | y <- xs, y > x]

merge :: [Int] -> [Int] -> [Int]
merge [] ys = ys
merge xs [] = xs
merge (x:xs) (y:ys) = if x <=y then x: (merge (xs) (y:ys))
                    else y : (merge (x:xs) ys)

mSort :: [Int]-> [Int]
mSort [] = []
mSort [x] = [x]
mSort  xs = merge (mSort left) (mSort right)
          where (left, right) = splitAt ((length xs) ‘div‘ 2) (xs)
```

other stuff

```
N = a ’div’ length xs
    where
         a = 10
          xs =  [1,2,3,4,5]
```

```
grid :: Int -> Int -> [(Int, Int)]
```