```haskell
-------------------
--- Selection Sort

delete y [] = []
delete y l = [x|x<-l,x /= y]

minimo [x] = x
minimo (x:y:t) = if x < y then minimo (x:t)
                 else minimo (y:t)

ssort [] = []
ssort [x] = [x]
ssort l   =  m:ssort l'
             where
                m = minimo l
                l'= delete m l
----------------------------------------
--- Insertion Sort

insert x [] = [x]
insert x (y:t)= if x < y then (x:y:t)
                else y:insert x t

isort [] = []
isort [x] = [x]
isort (x:t) = insert x (isort t)

----------------------------------------
--- Quick Sort

qsort [] = []
qsort [x] = [x]
qsort (x:t) = qsort menores ++ [x] ++ qsort mayores
              where
                   menores = [j | j<-t, j < x]
                   mayores = [j | j<-t, j >= x]
--- Version 2
qsort2 [] = []
qsort2 [x] = [x]
qsort2 (x:t) = let
                  menores = [j | j<-t, j < x]
                  mayores = [j | j<-t, j >= x]
               in
                  qsort2 menores ++ [x] ++ qsort2 mayores

----------------------------------------
--- Merge Sort

split [] = ([],[])
split [a] = ([a], [])
split (a:b:t)= let
                    (m,n) = split t
               in
                    (a:m, b:n)
---------
split2 [] = ([],[])
split2 [a] = ([a], [])
split2 (a:b:t)= (a:m, b:n)
                where
                (m,n) = split2 t
---------

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

msort []= []
msort [x] = [x]
msort lista =
```

```
74              let
75                  (i,j) = split lista;
76                   i' = msort i;
77                   j' = msort j
78              in
79                  merge i' j'
80  --------------------------------
81  --- Previamente ver el funcionamiento de
82  --- La funcion de orden superior "foldr"
83
84  --- Observar que fold necesita una funcion de dos variables
85  --- ej.: suma x y = x+y
86  ----foldr suma 0 [2,3,6,8,67] ..2+(3+(6+(8+(67+0))))
87
88  isort2 [] = []
89  isort2 [x] = [x]
90  isort2 (l) = foldr insert [] l
91
92  ---------
93  msort2 []= []
94  msort2 [x] = [x]
95  msort2 l = foldr merge [] [[x]|x<-l]
96
97
```