```
qsort [] = []
qsort [x] = [x]
qsort (x:t) = qsort menores ++ [x] ++ qsort mayores
              where
                    menores = [y | y <- t, y <= x]
                    mayores = [y | y <- t, y > x]
---- [3,1]

----  pivot = 3 , menores = [1], mayores= []
----     qsort ([1]) ++ [3] ++ qsort ([])
----     [1] ++ [3] ++ [] = [1,3]

---- t=1---- n/2
---- t=2 --- n/4
---  t=3 --- n/8
----- t ---- (n/2^t) --- n/2^t = 1 --- n = 2^t---- t = log2 n

--- Qsort ---->O (n log2 n)


particion [] p = ([], [])
particion (x:t) p = if x <= p then (x:men, may) else (men,x:may)
                    where (men,may) = particion t p









split []     = ([],[])
split [x]    = ([x],[])
split(x:y:t) = (x:m1,y:m2)
                where (m1,m2) = split t


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 (l) = let
                (m1,m2) = split l;
                 m1p = msort m1;
                 m2p = msort m2
            in
                 merge m1p m2p

--- l = [3,1]
```

```
--- m1 = [3], m2 = [1]
--- m1p = [3], m2p = [1]
--- merge (3:[]) (1:[]) = 1: (merge [3] []) = 1:[3] = [1,3]
```