

```
1 open List;;
2
3 (* implementazione del tipo di dato 't seq *)
4 type 't seq = 't list;; (* sequenze polimorfe implementate con liste *)
5
6 (* operazioni di 't seq richieste in QuickSort *)
7 let (size:'t seq ->int) =
8     (* Out: numero di elementi 't contenuti in seq *)
9     (* effect: nessuna modifica *)
10     fun x -> length x
11 ;;
12 let (pivot:'t seq -> ('t * 't seq)) =
13     (* Out: coppia (ElementoSelezionato,SequenzaRimanente) per seq *)
14     (* effect: nessuna modifica *)
15     fun x -> (hd x,tl x)
16 ;;
17 let (dividi: 't seq -> ('t -> 't -> bool) -> 't -> ('t seq * 't seq)) =
18     (* Out: partiziona seq (rispetto a pv, con ordine op) in una *)
19     (* coppia (MinoriUguali,Maggiori) *)
20     (* effect: nessuna modifica *)
21     fun seq op pv -> partition (fun x -> (op x pv)or(x = pv)) seq
22 ;;
23 let (concatena: 't seq -> 't seq -> 't seq) =
24     (* Out: concatena seq1 con seq2 *)
25     (* effect: nessuna modifica *)
26     fun x y -> x@y
27 ;;
28 let (cons: 't -> 't seq -> 't seq) =
29     (* Out: aggiunge x in testa ad xs *)
30     (* effect: nessuna modifica *)
31     fun x xs -> x::xs
32 ;;
33
34
```