```fsharp
1  (* Solution to Problem 1 from exam winter 2013 in 02157     Michael R. Hansen
2                                                                    07-10-2015   ⮡
                      *)
3
4  type Multiset<'a when 'a : equality> = ('a * int) list;;
5
6  (* Q1 *)
7  let rec inv = function
8                  | [] -> true
9                  | (e,k)::m -> k>0 && List.forall (fun (e',_) -> e <> e') m && inv   ⮡
                    m;;
10
11 let rec inv' es = function
12                     | []                                         -> true
13                     | (e,k)::_ when k<=0 || Set.contains e es  -> false
14                     | (e,_)::ms                                 -> inv' (Set.add e   ⮡
                       es) ms;;
15
16 let inv1 ms = inv' Set.empty ms;;
17
18 let ms1 = [("b",3); ("a",5); ("d",1)];;
19
20
21 (* Q2 *)
22 let rec insert e k m =
23    if k <= 0 then failwith "multiset: argument error"
24    else match m with
25        | []                    -> [(e,k)]
26        | (e',k')::m' when e=e' -> (e,k+k')::m'
27        | (e',k')::m'           -> (e',k')::insert e k m';;
28
29
30 (* Q3 *)
31 let rec numberOf e = function
32                        | []                  -> 0
33                        | (e',k)::m when e=e' -> k
34                        | _::m'               -> numberOf e m';;
35
36 (* Q4 *)
37 let rec delete e = function
38                      | [] -> []
39                      | (e',1)::m when e=e' -> m
40                      | (e',k)::m when e=e' -> (e,k-1)::m
41                      | (e',k)::m           -> (e',k)::delete e m;;
42
43
44 (* Q5 *)
45
46 let rec union(m, m2) =
47    match m with
48    | []        -> m2
49    | (e,k)::m1 -> union(m1, insert e k m2);;
```

```
50
51  let union1(m1, m2) = List.foldBack (fun (e,k) m -> insert e k m)  m1 m2;;
52
53  let mu =  union ([("b",3); ("a",5); ("d",1)], [("a",3); ("b",4); ("c",2)]);;
54
55  (* Q6 *)
56
57  type MultisetMap<'a when 'a : comparison> = Map<'a,int>;;
58
59  let inv2 m = Map.forall (fun _ n -> n>0) m;;
60
61  let insert1 e k m =
62      if k<=0 then failwith "insert1: argument error"
63      else match Map.tryFind e m with
64          | None    -> Map.add e k m
65          | Some k' -> Map.add e (k+k') m;;
66
67  let union2(m1,m2) = Map.foldBack insert1 m1 m2;;
68
69  let mu1 =  union2(Map.ofList[("b",3); ("a",5); ("d",1)], Map.ofList [("a",3);     ↵
        ("b",4); ("c",2)]);;
70
71
```