Dictionaries and Functional Arrays

1 Dictionaries

A *dictionary* associates some unique *keys* with corresponding *values*. In OCaml, these keys and their values are stored in a list of *pairs*. A pair is a special case of a *tuple*, which may contain two or more objects (not necessarily of the same type). For example, the tuple `(1, false, 'a')` has type `int` × `bool` × `char`. We now present some useful functions when working with dictionaries.

```
add : 'a -> 'b -> ('a * 'b) list
mklists : ('a * 'b) list -> 'a list * 'b list

let rec add k v d =
  match d with
    [] -> [(k, v)]
  | (k', v')::t ->
      if k = k'
        then (k, v) :: t
        else (k', v') :: add k v t

let rec mklists l =
  match l with
    [] -> ([], [])
  | (k, v)::t ->
      let (kt, vt) = mklists t in
        (k :: ks, k :: vs)
```

Note that the add function replaces a duplicate key with the newer key value pair. The mklists (make lists) function decomposes a dictionary into a tuple of lists, one containing the keys, the other containing the values. These can be extracted by pattern matching in the usual way. Alternatively, the following syntax may be used:

```
let first (x, _) = x
let second (_, y) = y
```