CLASSIC FUNCTIONS OVER LISTS (40/40 points)

In this exercise, we implement the classic functions over lists.

1. Write a function `mem : int -> int list -> bool` such that `mem x l` is true if and only if `x` occurs in `l`.
2. Write a function `append : int list -> int list -> int list` such that `append l1 l2` is the concatenation of `l1` and `l2`.
3. Write a function `combine : int list -> int list -> (int * int) list` such that `combine l1 l2` is the list of pairs obtained by joining the elements of `l1` and `l2`. This function assumes that `l1` and `l2` have the same length. For instance, `combine [1;2] [3;4] = [(1, 3); (2, 4)]`.
4. Write a function `assoc : (string * int) list -> string -> int option` such that `assoc l k = Some x` if `(k, x)` is the first pair of `l` whose first component is `k`. If no such pair exists, `assoc l k = None`.

YOUR OCAML ENVIRONMENT

```
let rec mem x l = match l with
  | [] -> false
  | y ::  ys -> if x = y then true else mem x ys
;;

let rec append l1 l2 = match l2 with
  | [] -> l1
  | x :: xs -> append (l1 @ [x]) xs
;;

let rec combine l1 l2 = match l1, l2 with
  | [], [] -> []
  | (x :: xs), (y :: ys) -> [(x, y)] @ (combine xs ys)
;;

let rec assoc l k = match l with
  | [] -> None
  | y :: ys -> match y  with
    | (a, b) ->  if a = k then Some b else assoc ys k
;;
```

Evaluate >
Switch >>
Typechecl
Reset Templ
Full-screen |
Check & Sa

Exercise complete (click for details) 40 pts

v Exercise 1: mem Completed, 10 pts

```
Found mem with compatible type.
Computing mem 6 []
Correct value false                                          1 pt
Computing mem 7 [12; 22; 10; 8; 18; 6]
Correct value false                                          1 pt
Computing mem 25 [11; 15; 21; 25; 19]
Correct value true                                           1 pt
Computing mem 23 [14; 4; 8; 24; 2; 18; 20; 0; 10; 22]
Correct value false                                          1 pt
Computing mem 9 [19; 7; 13; 15; 17; 21; 5; 9; 1; 25]
Correct value true                                           1 pt
Computing mem 15 [25; 9; 3; 21; 7; 17; 15; 19; 23]
Correct value true                                           1 pt
Computing mem 6 [3; 1; 7]
Correct value false                                          1 pt
Computing mem 21 [17; 3; 19; 1; 11; 5; 7; 13; 21]
Correct value true                                           1 pt
Computing mem 22 [19; 5; 23; 9]
```

```
v Exercise 2: append                                              Completed, 10 pts
  Found append with compatible type.
  Computing append [57; 65] [2; 46; 65]
  Correct value [57; 65; 2; 46; 65]                                           1 pt
  Computing append [29] [47; 20; 20]
  Correct value [29; 47; 20; 20]                                              1 pt
  Computing append [42; 71; 71; 77] [47; 7; 14; 5]
  Correct value [42; 71; 71; 77; 47; 7; 14; 5]                                1 pt
  Computing append [14; 8; 13; 50] [79; 33; 77; 56; 33]
  Correct value [14; 8; 13; 50; 79; 33; 77; 56; 33]                           1 pt
  Computing append [2; 75; 29; 67; 73] [54]
  Correct value [2; 75; 29; 67; 73; 54]                                       1 pt
  Computing append [70; 74] [62; 40]
  Correct value [70; 74; 62; 40]                                              1 pt
  Computing append [10; 38; 25] [28]
  Correct value [10; 38; 25; 28]                                              1 pt
  Computing append [51] [77]
  Correct value [51; 77]                                                      1 pt
  Computing append [5; 33; 30] [3; 65; 31]
  Correct value [5; 33; 30; 3; 65; 31]                                        1 pt
  Computing append [12] [19]
  Correct value [12; 19]                                                      1 pt
v Exercise 3: combine                                             Completed, 10 pts
  Found combine with compatible type.
  Computing combine [] []
  Correct value []                                                            1 pt
  Computing combine [1] [2]
  Correct value [(1, 2)]                                                      1 pt
  Computing combine [1; 2; 3] [0; 0; 0]
  Correct value [(1, 0); (2, 0); (3, 0)]                                      1 pt
  Computing combine [21; 39; 1; 70; 21] [48; 13; 38; 0; 43]
  Correct value [(21, 48); (39, 13); (1, 38); (70, 0); (21, 43)]              1 pt
  Computing combine [65; 40; 67; 64; 79] [37; 65; 67; 68; 76]
  Correct value [(65, 37); (40, 65); (67, 67); (64, 68); (79, 76)]            1 pt
  Computing combine [52; 37; 68; 32; 47] [22; 14; 79; 25; 11]
  Correct value [(52, 22); (37, 14); (68, 79); (32, 25); (47, 11)]            1 pt
  Computing combine [5; 40; 34; 25; 10] [70; 40; 41; 68; 3]
  Correct value [(5, 70); (40, 40); (34, 41); (25, 68); (10, 3)]              1 pt
  Computing combine [31; 61; 33; 33; 58] [47; 1; 14; 71; 16]
  Correct value [(31, 47); (61, 1); (33, 14); (33, 71); (58, 16)]             1 pt
  Computing combine [34; 34; 17; 60; 47] [63; 7; 67; 39; 74]
  Correct value [(34, 63); (34, 7); (17, 67); (60, 39); (47, 74)]             1 pt
  Computing combine [74; 41; 30; 14; 2] [45; 17; 46; 28; 68]
  Correct value [(74, 45); (41, 17); (30, 46); (14, 28); (2, 68)]             1 pt
v Exercise 4: assoc                                               Completed, 10 pts
  Found assoc with compatible type.
  Computing
    assoc
      [("sig", 8); ("as", 30); ("begin", 46); ("ocp", 21); ("match", 69)]
      "mutable"
  Correct value None                                                          1 pt
  Computing assoc [("object", 56); ("rec", 72)] "object"
  Correct value (Some 56)                                                     1 pt
  Computing
  assoc [("if", 71); ("when", 35); ("for", 1); ("mod", 39); ("done", 38)] "if"
  Correct value (Some 71)                                                     1 pt
  Computing
    assoc
      [("do", 2); ("struct", 66); ("and", 18); ("module", 3); ("let", 40);
       ("object", 45)]
      "and"
  Correct value (Some 18)                                                     1 pt
  Computing assoc [("match", 74); ("as", 21)] "match"
  Correct value (Some 74)                                                     1 pt
  Computing assoc [("when", 72); ("let", 41); ("if", 71)] "when"
Computing assoc [("sig", 65); ("do", 41)] "sig"
Correct value (Some 65)                                                         1 pt
Computing
  assoc
    [("ocp", 14); ("struct", 66); ("begin", 71); ("module", 47); ("mod", 20);
     ("rec", 57)]
    "rec"
Correct value (Some 57)                                                         1 pt
Computing assoc [("begin", 29); ("mod", 29); ("and", 25); ("if", 23)] "with"
Correct value None                                                              1 pt
```