```
;; PROBLEM 1 [1:45]
;;
;; Here are just the first two local function definitions from the maze solver
;; in the last lecture:

(define (solvable? m)
  ;; base case test: solved? is is pos lower right
  ;; reduction: down and right but within maze and no walls
  ;; argument: maze is finite, so moving only down or right
  ;;           must run into walls, edge of maze, or solved
  (local [(define (fn-for-pos p)
            (cond [(solved? p) true]
                  [else
                   (fn-for-lop (valid-next-positions p))]))

          (define (fn-for-lop lop)
            (cond [(empty? lop) false]
                  [else
                   (local [(define try (fn-for-pos (first lop)))]
                     (if (not (false? try))
                         try
                         (fn-for-lop (rest lop))))]))
          ...
          ]))

;; What is the most correct template tag for this code:

;; A. (@template encapsulated Pos (listof Pos))
;; B. (@template encapsulated Pos (listof Pos) try-catch)
;; C. (@template encapsulated genrec Pos (listof Pos) try-catch)
;; D. (@template encapsulated genrec arb-tree try-catch)
;; E. (@template encapsulated genrec arb-tree Pos try-catch)
```

```
(define (solvable? m)
  (local [(define (fn-for-pos p)
            (cond [(solved? p) true]
                  [else
                   (fn-for-lop (valid-next-positions p))]))

          (define (fn-for-lop lop)
            (cond [(empty? lop) false]
                  [else
                   (local [(define try (fn-for-pos (first lop)))]
                     (if (not (false? try))
                         try
                         (fn-for-lop (rest lop))))]))
          ...
          ]))
```

Which is the correct assignment of template origins to colors?

A	B	C	D	E
arb-tree	arb-tree	arb-tree	arb-tree	arb-tree
genrec	genrec	genrec	genrec	genrec
try-catch	try-catch	try-catch	try-catch	try-catch