Homework 4
COSE212, Fall 2025

Hakjoo Oh

Problem 1 Consider the language ML^- from HW3:

P	$\to$	E	
E	$\to$	`()`	unit
	$\mid$	`true` $\mid$ `false`	booleans
	$\mid$	n	integers
	$\mid$	x	variables
	$\mid$	E `+` $E \mid E$ `-` $E \mid E$ `*` $E \mid E$ `/` E	arithmetic
	$\mid$	E `=` $E \mid E$ `<` E	comparison
	$\mid$	`not` E	negation
	$\mid$	`nil`	empty list
	$\mid$	E `::` E	list cons
	$\mid$	E `@` E	list append
	$\mid$	`head` E	list head
	$\mid$	`tail` E	list tail
	$\mid$	`isnil` E	checking empty list
	$\mid$	`if` E `then` E `else` E	if
	$\mid$	`let` x `=` E `in` E	let
	$\mid$	`letrec` $f(x)$ `=` E `in` E	recursion
	$\mid$	`letrec` $f(x_1) = E_1$ and $g(x_2) = E_2$ `in` E	mutual recursion
	$\mid$	`proc` x E	function definition
	$\mid$	E E	function application
	$\mid$	`print` E	print
	$\mid$	$E; E$	sequence

In OCaml datatype:

```
type program = exp
and exp =
  | UNIT
  | TRUE
  | FALSE
  | CONST of int
  | VAR of var
```

```
  | ADD of exp * exp
  | SUB of exp * exp
  | MUL of exp * exp
  | DIV of exp * exp
  | EQUAL of exp * exp
  | LESS of exp * exp
  | NOT of exp
  | NIL
  | CONS of exp * exp
  | APPEND of exp * exp
  | HEAD of exp
  | TAIL of exp
  | ISNIL of exp
  | IF of exp * exp * exp
  | LET of var * exp * exp
  | LETREC of var * var * exp * exp
  | LETMREC of (var * var * exp) * (var * var * exp) * exp
  | PROC of var * exp
  | CALL of exp * exp
  | PRINT of exp
  | SEQ of exp * exp
and var = string
```

Types for the language are defined as follows:

```
type typ =
    TyUnit
  | TyInt
  | TyBool
  | TyFun of typ * typ
  | TyList of typ
  | TyVar of tyvar
and tyvar = string
```

Implement a sound type checker, `typeof`, for the language (the notion of soundness is defined with respect to the dynamic semantics of the language defined in HW3):

```
typeof : exp -> typ
```

which takes a program and returns its type if the program is well-typed. When the program is ill-typed, `typeof` should raise an exception `TypeError`.