Homework 4
COSE212, Fall 2017

Hakjoo Oh

Due: 11/26, 24:00

Problem 1 Consider the language:

```
type exp =
  | CONST of int
  | VAR of var
  | ADD of exp * exp
  | SUB of exp * exp
  | MUL of exp * exp
  | DIV of exp * exp
  | READ
  | ISZERO of exp
  | IF of exp * exp * exp
  | LET of var * exp * exp
  | LETREC of var * var * exp * exp
  | PROC of var * exp
  | CALL of exp * exp
and var = string
```

Types for the language are defined as follows:

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

Implement the following type-inference function:

```
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`.

Examples:

- The program

```
PROC ("f",
 PROC ("x", SUB (CALL (VAR "f", CONST 3),
                 CALL (VAR "f", VAR "x"))))
```

has type `TyFun (TyFun (TyInt, TyInt), TyFun (TyInt, TyInt))`.

- The program

  ```
  PROC ("f", CALL (VAR "f", CONST 11))
  ```

  has type `TyFun (TyFun (TyInt, TyVar "t"), TyVar "t")`, where `t` can be any type variable.

- The program

  ```
  LET ("x", CONST 1,
    IF (VAR "x", SUB (VAR "x", CONST 1), CONST 0))
  ```

  is ill-typed, so `typeof` should raise an exception `TypeError`.

As discussed in class, `typeof` is defined with two functions: one for generating type equations and the other for solving the equations. Complete the implementation of these two functions:

```
gen_equations  :  TEnv.t -> exp -> typ -> typ_eqn
        solve  :  typ_eqn -> Subst.t
```

Modules for type environments (`TEnv`) and substitutions (`Subst`), as well as the operations of applying substitutions to types (`Subst.apply`) and extending substitutions (`Subst.extend`), are provided.