Daniel Leeds, 15-212 R12, November 14, 2007

Definitions provided here are incomplete! Have pages 12 and 13 of assn5.pdf in front of you while you code.

Parsed Input:

Expressions and Declarations:

```
datatype Exp   = Int of int
               | Bool of bool
               | Unit
               | Var of ident
               | Tuple of Exp list
               | App of Exp * Exp
               | IfThenElse of Exp * Exp * Exp
datatype Decl  = Val of Pattern * Exp
               | Fun of ident * (Pattern * Type) * Type * Exp
```

The structures above get evaluated to:

Types and Values:

```
datatype Type  = TUnit
               | Tint
               | TTuple of Type list
               | TArrow of Type * Type
datatype Value = VInt of int
               | VBool of bool
               | VUnit
               | VFn of ValueCtx * Pattern * Type * Exp
```

Note: I put “variables” in quotes below, as I use it somewhat loosely. A more strict definition may appear in the assignment.

Type-checking

The *context* Γ defines a set of bindings between “variables” and types. Given a new expression, *e*, we determine its type based on the types of its variables, as defined by Γ, and by the rules of evaluation given in section 4 of the assignment.

Extending context

You can add new variable-type bindings to a context Γ, generating a new context Γ’.

Evaluating

The *closure* Σ defines a set of bindings between “variables” and the values to which they evaluate. It basically works the same way as type-checking.

Matching

You can add new variable-value bindings to a closure Σ---that is, match them together---generating a new closure Σ’.