Syntactic Transformation To Monadic Form

- ***Expressions:***

```
------ exp -------
```

$desugar_{<exp>} :: Exp \rightarrow Exp$

```
desugar<exp> exp = desugar<lexp> exp >>= \h -> return h
```

```
------- lexp -------
```

$desugar_{<lexp>} :: Exp \rightarrow Exp$

```
-------lexp: fexp -------
```

```
desugar<lexp> fexp = desugar<fexp> fexp
```

```
------- fexp -------
```

$desugar_{<fexp>}$ aexp = $desugar_{<aexp>}$ aexp

$desugar_{<fexp>}$ (fexp literal) = $desugar_{<lexp>}$ fexp literal >>= $\backslash g_i \rightarrow g_i$

$desugar_{<fexp>}$ (fexp qvar) = $desugar_{<lexp>}$ fexp qvar >>= $\backslash g_i \rightarrow g_i$

$desugar_{<fexp>}$ (fexp gcon) = $desugar_{<lexp>}$ fexp gcon >>= $\backslash g_i \rightarrow g_i$

$desugar_{<fexp>}$ (fexp (exp))= $desugar_{<lexp>}$ fexp exp >>= $\backslash ex_i \rightarrow ex_i$

$desugar_{<fexp>}$ (fexp (exp1,…, exp_k))= $desugar_{<lexp>}$ fexp (exp1,…, exp_k) >>=
$\backslash tuple \rightarrow$ tuple

$desugar_{<fexp>}$ (fexp [exp1,…, exp_k])= $desugar_{<lexp>}$ fexp [exp1,…, exp_k] >>=
$\backslash list \rightarrow$ list

```
------- aexp -------
```

$desugar_{<aexp>}$ literal = literal

$desugar_{<aexp>}$ qvar = qvar

$desugar_{<aexp>}$ gcon = gcon

$desugar_{<aexp>}$ (exp) = ($desugar_{<lexp>}$ exp)

$desugar_{<aexp>}$ (exp1,…, exp_k) = ($desugar_{<lexp>}$ exp_1, … , $desugar_{<lexp>}$ exp_k)

$desugar_{<aexp>}$ [exp1,…, exp_k] = [$desugar_{<lexp>}$ exp_1, … , $desugar_{<lexp>}$ exp_k]

```
-------lexp: let decls in exp -------
```

desugar_{<lexp>} (let decls in exp) = desugar_{<dclrs>} decls exp

- ***Declarations***

```
-------  dclrs -------
```

desugar_{<dclr> ::} Dclrs → Exp

desugar_{<dclrs>} (dclr1; … ; dclrn) = desugar_{<dclr>} declr1 …   desugar_{<dclr>} declrn
| (;) = _ ->

```
-------  dclr -------
```

desugar_{<dclr>} (funlhs | pat) (= exp) = exp >>= \(funlhs | pat) ->