**Principles of Programming Languages**

Assignments 3

1. As we know the *let* expression is a syntactic abbreviation and can be considered as a special form in L3, the interpreter of L3 has a special rule of evaluation, let is converted into AppExp, and we know that some of the compound *let* expression are not evaluated at all.
2. The role of the function *valueToLitExp* is to substitute value to their Exp (number to *NumExp*, boolean to *BoolExp* and etc.) to create a valid AST.
3. The function *valueToLitExp* is not needed in the normal evaluation strategy because in normal evaluation we make substitution before the evaluation of the arguments, and *valueToLitExp* is needed when variables are substituted with values.
4. The function *valueToLitExp* is not needed in the environment-model interpreter there is no need to substitute variables with values (as in normal evaluation). In environment-model interpreter we use *varRef* and new environments in computations.
5. Switching between applicative order to normal order:
   1. The function *valueToLitExp* is not needed.
   2. Avoid unnecessary calculations (in some cases).
   3. Can finish the evaluations when an applicative order can throw an error.
6. Switch between normal order to applicative order:
   1. Avoid recalculation of the same expression (in some cases).
7. Only the Devil knows

P2: (f, g)

B2: (lambda (x)

(if (= x 0)

x

(g x)))

GE

a: 2

goo:

foo:

P1: (x)

B1: (lambda (y) (/ x y))

x:0

E3

E4

f:

g:

E1

P3: (y)

B3: (lambda (y) (/ x y))

x:2

E2

P4: (x)

B4: (f x)

x:0

P5: (x)

B5: (g)

0

0