Assignment 4

Ahiad Shaiker & Nitzan Guetta

1. A. {T1=[T3->T4], T5=[T3->T4], T2=Number}

   B. No unifier

   C. {T1=T2}

   D. { }

2. When evaluating letrec expression, we need to evaluate the body of closure (we will name it as 'a'), which may include expression (we will name it 'b') that was defined in the next binding. To evaluate expression 'b', it needs a binding of 'b' to it's value, which defined after that, and that's the problem. Alternatively, when type checking, even though it does not have values of 'b' yet, It needs to compare it with type annotations, that are already given to us. as a result, no evaluation of the body needed, and there aren't any recursive environment problems.
3. ((lambda (a)

   ((lambda (b)

   ((lambda (c)

   ((lambda (d)

   #f)

   c)

)b)

)a)

)1)

Test: (typeOfExp (parseL5 (lambda (a) (lambda (b) (lambda (c) (lambda (d) #f)))))

(make-empty-tenv)) => #f