**Assignment 3**

1. Part 1: Theoretical Questions
   1. Answer:
      1. Let's find the MGUs for the type expressions
         1. {T2 = Number, T1 = [T3->T4], T5 = [T3->T4]}
         2. No unifier exists, because T1 needs to be a Number and a Symbol.
         3. {T1=T2}
         4. {}, an empty MGU
      2. following typing statements true or false?
         1. True, because it demonstrates composition. The g function gets a T1 (that could be anything including Boolean), then returns a T2 type argument and this argument is the input for f function and the f function returns a T3 as required.
         2. False, because f function except to get T2 argument and x is a T1 argument.
   2. Answer:  
      A program that produces a chain of type variables of length 5 is:  
      (lambda (x) (x (x (x (x x))))).  
      When passed to the type inference algorithm, the system infers a type for x that depends on its own application multiple times. Each nested application introduces a new unification constraint of the form TVar\_i = [TVar\_{i+1} -> TVar\_{i+2}], leading to a chain like:  
      TVar1 → TVar2 → TVar3 → TVar4 → TVar5