Due: 2022/4/23

Homework 10 - Inference2

∗ If there is any problem, please contact TA.

Name:_________ Student ID:_________ Email: ____________

Problem 1. Prove the Lemma: If $(S, q) \to (S', q')$ then:

- T is complete for (S, q) iff T is complete for (S', q')
- T is principal for (S, q) iff T is principal for (S', q')

Problem 2. Given the following variant of untyped lambda calculus:

```
e::=
x (variables)
| c (constants)
| \x.e
| e1 e2
| e1 bop e2 (binary op)
| uop e (unary op)
| let x = e1 in e2
| if e1 then e2 else e3
| letfun f(x) = e1 in e2  (defining a recursive function f(x) for use in e2)
| {e1, e2}
| e.1
| e.2
| inl e
| inr e
| case e1 of inl x => e2 | inr x => e3
| nil
| e1 :: e2
| case e1 of nil => e2 | x1 :: x2 => e3
| (e)
```

(a) Inductively define the constraint generation judgement:

```
G |- u ==> e:t, q
```

(b) Give the detailed derivation of the following expressions and obtain the set of equations, then solve these equations by unification algorithm to get the principle solution and give the universal polymorphic types:

```
letfun sum(l) = case l of nil => 0 | x1 :: x2 => x1 + sum(x2)
in sum(12::10::0::nil)
```

Problem 3. Show why type checking let expression using [t-LetPoly] is exponential in time and give an amortised linear implementation of let polymorphism instead.

Remark: You just need to send your .pdf file to likaijian@sjtu.edu.cn. Email Subject line Format(also the pdf file name): HW_X_Name_StudentID