Guideline

Due Date: Thursday, 2023-09-28, by 23:59.

Upload your answers as a singular PDF to Brightspace.

If you're writing by hand, please ensure your handwriting is legible.

Multiple submissions are possible before the due time; the last submission will be graded.

Exercise 1 (points = 40)

Check if the following is alpha equivalent

- (λx.xy) vs. (λx.yx)
- (λx.xu) vs. (λx.xv)
- (λa.λb.abc) vs. (λx.λy.xyz)
- (λx.λy.xyz) vs. (λy.λx.yxz)

Exercise 2 (points = 10)

True or False?

- The lambda expression ((λx. λy.y) y) (λx.x z) reduces to z as the normal form.

If false, provide the correct normal form.

Exercise 3 (points = 40)

Reduce the lambda term below until you get a normal form

- (λx.λy.xyz)y x
- (λx.λf.λy.fxy)(fy)
- (λx.λy.yx)(λx.xx)y
- (λx.λy.xyy)(λy.y)z

Exercise 4 (points = 10)

Let f be a function of the type `A -> B`, and g be a function of the type `B -> C`. The composition of f and g, will be a function of the type `A -> C`. This composition is read as “g of f”. For example, if we let f denote the function that increments an integer by 1, and g denote division by 2, then the composition g of f applied to 41 will yield 21 (i.e., 41+1, then divided by 2).

Find out the lambda expression "compose" that takes g and f as input, and produce the composition “g of f” as explained above. In other words, you need construct a lambda term, denoted by `compose` such that `compose`

`g f x` will reduce to `g (f x)`.