Foundations of Functional Programming

Explain why the Church–Rosser theorem implies that there are \( \lambda \)-terms that are not equal to each other. \hspace{1cm} [2 marks]

Suppose the following reduction rule is added to the \( \lambda \)-calculus:

\[
\lambda xy.x \rightarrow \lambda xy.y
\]

Show that in the resulting calculus, all terms are equal. \hspace{1cm} [3 marks]

Let \( A = \lambda xy.y(xy) \) and \( \Theta = AA \). Show that \( \Theta \) is a fixed-point combinator. \hspace{1cm} [3 marks]

Assume an encoding of lists where \([a_1, \ldots, a_m]\) is represented by

\[
\lambda fx.f(a_1)(f(a_2)(\cdots(f(a_mx)\cdots))
\]

Use the fixed-point combinator \( \Theta \) to obtain a \( \lambda \)-term \( \text{rev} \) such that:

\[
\text{rev}[a_1, \ldots, a_m] = [a_m, \ldots, a_1]
\]

You may assume a \( \lambda \)-term representation of the booleans and of ordered pairs, but you should define any other terms you require. \hspace{1cm} [12 marks]