(a) Writing as usual → for 1-step reduction (i.e. β-η-reduction with α-conversion only used to avoid name clashes) and →* for its reflexive-transitive closure, indicate giving reasons (you may merely claim well-known results) whether the following statements are always, never or sometimes true of pure λ-terms L, M and N.

(i) if $M = N$ then $M \rightarrow N$ or $N \rightarrow M$.

(ii) if $M \rightarrow M$ then $M$ is in normal form.

(iii) if $L \rightarrow M$ and $L \rightarrow N$ then there exists $L'$ such that $M \rightarrow L'$ and $N \rightarrow L'$

(iv) if $L \rightarrow M$ and $L \rightarrow N$ then there exists $L'$ such that $M \rightarrow L'$ and $N \rightarrow L'$

[2 marks each]

(b) Define λ-terms if, true and false that satisfy that if true $M \; N = M$ and if false $M \; N = N$. [2 marks]

(c) Given your definitions in part (b) above, indicate giving reasons whether it is always, never or sometimes true that:

(i) if true $M \; N \rightarrow^e M$ where $\rightarrow^e$ represents eager evaluation

(ii) if true $M \; N \rightarrow^\ell M$ where $\rightarrow^\ell$ represents lazy evaluation

[3 marks each]

(d) Explain why the β-reduction rule tends not to be used literally for implementing functional programming languages, indicating two alternatives. [4 marks]