Exercise 1 (Reduction Relation with Closures)

For the evaluation of lambda terms that is closer to evaluation of programs in functional programming languages, one usually replaces textual substitution $t[v/x]$ with a more lazy approach that records the binding $x \mapsto v$ in an environment. These bindings are used whenever we need the value of a variable $v$.

In this approach abstractions $\lambda x. t$ do not evaluate to themselves, but to a pair $(\lambda x. t)[e]$, where $e$ is the current environment. We call such pairs function closures.

a) Define a big-step reduction relation for the lambda calculus with function closures and environments.

b) Add explicit error handling for the case where the binding of a free variable $v$ cannot be found in the environment. Introduce an explicit value `abort` to indicate such an exception in the relation.

Exercise 2 (Better Translation Algorithm)

Give a variant of the translation algorithm that produces shorter terms. More specifically, define a variant of $\lambda^*x. t$ that analyzes more precisely where $x$ actually appears in $t$. 

Homework 3 (Proofs with Small-steps and Big-steps)

Let \( \omega := \lambda x. x x \) and
\[
  t := (\lambda x. (\lambda y. z) y) (\omega \omega ((\lambda x y. x) y)).
\]
Prove the following:

a) \( t \Rightarrow_n z \)
b) \( t \rightarrow_3^{cbv} t \)
c) \( t \not\rightarrow_3^{cbn} t \)

Homework 4 (More Combinators)

Find combinators \( O \) and \( W \) such that:
\[
  O \rightarrow^+ O \\
  W X Y \rightarrow^* X Y Y
\]

Homework 5 (Mocking Birds)

Consider a combinatory logic that only provides the basic combinators \( B \) and \( M \) (the “mocking bird”) where:
\[
  B X Y Z \rightarrow^* X (Y Z) \\
  M X \rightarrow^* X X
\]
Prove the following properties of this logic:

a) For every combinator \( X \), there is a combinator \( Y \) such that \( Y \rightarrow^* X Y \).

b) For all combinators \( U \) and \( W \), there exist combinators \( X \) and \( Y \) such that \( Y \rightarrow^* U X \) and \( X \rightarrow^* W Y \).

Homework 6 (Correctness of the Translation Algorithm)

Show that the translation algorithm given in the tutorial is correct. That is, show that it fulfills the following property:
\[
  (\lambda^* x. X) Y \rightarrow^* X[Y/x]
\]