The following exercises are designed to test your understanding of recursion. The functions are defined using a variant of LISP known as meta-LISP. In order to aid your understanding, the function defined in problem 1 is identical to the one below:

\[
drop(x) = \begin{cases} 
nil & \text{if } \text{null}(x) \\
(car \ x) \text{ cons } \text{drop}(cdr \ x) & \text{otherwise}
\end{cases}
\]

The idea is that
\[
a \ x = \text{car}(x)
\]
\[
d \ x = \text{cdr}(x)
\]
\[
n \ x = \text{null}(x)
\]
\[
at \ x = \text{atom}(x)
\]
\[
a. b = a \text{ cons } b
\]
\[
\langle a \rangle = a \text{ cons } \text{nil} = a \text{ list whose single element is } a
\]
\[
a*b = \text{concatenate lists } a \text{ and } b \text{ (i.e. append list } b \text{ to list } a)
\]
\[
\text{reverse}[x] = \text{reverses the top level list } x. \text{ For example } \text{reverse}[(A \ B \ C)] = (C \ B \ A). \text{ But } \text{reverse}[(\langle A \ B \ C \rangle) (D \ E)] = ((D \ E) (A \ B \ C)).
\]

1. Consider the function drop defined by
\[
drop[x] \leftarrow \text{if } \text{null}(x) \text{ then } \text{nil} \text{ else } [a \ x]. \text{drop}[d \ x].
\]
Compute (by hand) drop[(A B C)]. What does drop do to lists in general?

2. What does the function
\[
r2[x] \leftarrow \text{if } \text{null}(x) \text{ then } \text{nil} \text{ else } \text{reverse}[a \ x]. \text{r2}[d \ x]
\]
do to lists of lists? How about
\[
r3[x] \leftarrow \text{if } \text{atom}(x) \text{ then } x \text{ else } \text{reverse}[r4[x]]
\]
\[
r4[x] \leftarrow \text{if } \text{null}(x) \text{ then } \text{nil} \text{ else } r3[a \ x]. \text{r4}[d \ x]?\]

3. Compare the following function with the function r3 of the preceding example:
\[
r3'[x] \leftarrow \text{if } \text{atom}(x) \text{ then } x \text{ else } r3'[d \ x] * \langle r3'[a \ x]\rangle
\]

4. Consider r5 defined by
\[
r5[x] \leftarrow \text{if } \text{null}(x) \lor \text{null}(d \ x) \text{ then } x \\
\text{else } [a \ r5[d \ x]] . \ r5[a \ x . \ r5[d \ r5[d \ x]]].
\]
Compute r5[(A B C D)]. What does r5 do in general. Needless to say, this is not a good way of computing this function even though it involves no auxiliary functions.