Exercise 1 isList

Define a predicate argIsList/1 that returns true if its argument is a list (including the empty list) and false otherwise. Hint: use the fact that any list (except the empty list) has both a head and a tail.

isList([]).
isList([_|_]).
Exercise 2 prepend

Define a predicate prepend/3, which takes any term as its first argument, a list as its second argument and returns as its third argument a list in with the first argument as its head and the second argument as its tail.

prepend(X, L, [X|L]) :- isList(L).
Define a predicate remove/3 which takes any term as its first argument, a list as its second argument and returns as its third argument a list equal to its second argument but with the first occurrence of the first argument removed (if it exists).

\[
\text{remove}(\_, [], []). \\
\text{remove}(X, [X|T], T). \\
\text{remove}(X, [Y|T], L) :- X \neq Y, \text{prepend}(Y, Z, L), \text{remove}(X, T, Z).
\]
Exercise 4 filter

Define a predicate filter/3 which behaves like remove but eliminates all occurrences of the first argument.

\[
\begin{align*}
\text{filter}(_, [], []). \\
\text{filter}(X, [X|T], L) & : \neg \text{filter}(X, T, L). \\
\text{filter}(X, [Y|T], L) & : X \neq Y \\
& \quad, \text{prepend}(Y, Z, L) \\
& \quad, \text{filter}(X, T, Z).
\end{align*}
\]
Exercise 5 frenchFlag

Define a predicate frenchFlag that takes a list as first argument and returns a list where all the ‘b’ atoms are grouped on the left, all the ‘w’ atoms are grouped in the middle, all the ‘r’ atoms are grouped on the right and all other atoms are eliminated.

frenchFlag(X, Y) :- sieve(b, X, B), sieve(w, X, W), sieve(r, X, R), append(B, W, BW), append(BW, R, Y).

sieve(_, [], []). sieve(X, [X | T], [X | Y]) :- sieve(X, T, Y).
sieve(X, [H | T], Y) :- X ≠ H, sieve(X, T, Y).
Define a predicate \texttt{frenchFlag} that takes a list as first argument and returns a list where all the ‘b’ atoms are group on the left, all the ‘w’ atoms are grouped in the middle, all the ‘r’ atoms are grouped on the right and all other atoms are eliminated.

\texttt{frenchFlag(X, Y) :- triage(X, [], [], [], Y).}

\texttt{triage([], B, W, R, Y) :- append(W,R,WR), append(B, WR, Y).}

\texttt{triage([b|T], B, W, R, Y) :- triage(T, [b|B], W, R, Y).}

\texttt{triage([w|T], B, W, R, Y) :- triage(T, B, [w|W], R, Y).}

\texttt{triage([r|T], B, W, R, Y) :- triage(T, B, W, [r|R], Y).}

\texttt{triage([H|T], B, W, R, Y) :- H\=b, H\=w, H\=r, triage(T, B, W, R, Y).}