Suppose that \( take \) and \( drop \) are ML functions such that \( take(n, s) \) returns the first \( n \) elements of the list \( s \), while \( drop(n, s) \) returns all but the first \( n \) elements of \( s \). Let \( length(s) \) be the function to compute the length of the list \( s \). Consider the following ML function

\[
\begin{align*}
\text{fun front } s & = \text{take(length } s \div 2, s) ; \\
\text{fun back } s & = \text{drop(length } s \div 2, s) ;
\end{align*}
\]

\[
\begin{align*}
\text{fun bsum } [ & ] = 0.0 \\
& | \text{bsum } [x] = x \\
& | \text{bsum } s = \text{bsum front } s + \text{bsum back } s ;
\end{align*}
\]

\[
\begin{align*}
\text{fun sum } [ & ] = 0.0 \\
& | \text{sum } (x :: s) = x + \text{sum } s ;
\end{align*}
\]

Give a formal proof that \( \text{sum(front } s) + \text{sum(back } s) = \text{sum(s)} \) for all lists \( s \), explaining what properties of arithmetic you are assuming. [9 marks]

Describe a proof of \( \text{bsum(s)} = \text{sum(s)} \) for all \( s \) using the lemma that you have just established. Do not give a detailed proof but instead outline the main argument. State any additional lemmas required and indicate how they might be proved. [6 marks]

Does proving \( \text{bsum(s)} = \text{sum(s)} \) for all \( s \) in this way ensure that \( \text{bsum} \) and \( \text{sum} \) are completely interchangeable in ML programs? Discuss. [5 marks]