Let \( n \) be a natural constant. Let \( S: n^*\text{nat} \) be an implementer’s variable. It is being reimplemented by \( R: n^*\text{nat} \) representing the same \( n \) naturals but in the reverse order.

§ It is convenient to define the “reverse” operator \( \leftarrow \). For any string \( S \), define \( \leftarrow S \) to be the reverse of \( S \). Formally, if \( n = \leftrightarrow S \),

\[
\forall j: 0..n \cdot S_j = (\leftarrow S)_{n-j-1}
\]

and equivalently

\[
\forall j: 0..n \cdot (\leftarrow S)_j = S_{n-j-1}
\]

and we can prove \( \leftarrow \) is its own inverse

\[
\leftarrow \leftarrow = S
\]

(a) What is the data transformer?

§

\[
D = R = \leftarrow S
\]

or equivalently

\[
D = S = \leftarrow R
\]

(b) A user has variable \( i: \text{nat} \) and the operation

\[
\text{get} = i := S_i
\]

Use your transformer from part (a) to transform \( \text{get} \).

§

\[
\forall S \cdot D \Rightarrow \exists S' \cdot D' \land \text{get}
\]

\[
= \forall S \cdot S = \leftarrow R \Rightarrow \exists S' \cdot S' = \leftarrow R' \land (i := S_i) \quad \text{expand assignment in the old variable}
\]

\[
= \forall S \cdot S = \leftarrow R \Rightarrow \exists S' \cdot S' = \leftarrow R' \land i' = S_i \land S' = S \quad \text{one-point}
\]

\[
= \forall S \cdot S = \leftarrow R \Rightarrow i' = S_i \land \leftarrow R' = S \quad \text{one-point}
\]

\[
= i' = (\leftarrow R)_i \land \leftarrow R' = \leftarrow R \quad \leftarrow \text{ is self-inverse}
\]

\[
= i' = R_{n-i-1} \land R' = R \quad \text{contract assignment in the new variable}
\]

\[
= i := R_{n-i-1}
\]