Proving that a language $L$ is undecidable by reduction requires several steps:

- Choose a language $L'$ that you already know is undecidable. Typical choices for $L'$ include:

$$\begin{aligned}
\textsc{Accept} &:= \left\{ \langle M, w\rangle \;\middle|\; M \text{ accepts } w \right\} \\
\textsc{Reject} &:= \left\{ \langle M, w\rangle \;\middle|\; M \text{ rejects } w \right\} \\
\textsc{Halt} &:= \left\{ \langle M, w\rangle \;\middle|\; M \text{ halts on } w \right\} \\
\textsc{Diverge} &:= \left\{ \langle M, w\rangle \;\middle|\; M \text{ diverges on } w \right\} \\
\textsc{NeverAccept} &:= \left\{ \langle M\rangle \mid \textsc{Accept}(M) = \varnothing \right\} \\
\textsc{NeverReject} &:= \left\{ \langle M\rangle \mid \textsc{Reject}(M) = \varnothing \right\} \\
\textsc{NeverHalt} &:= \left\{ \langle M\rangle \mid \textsc{Halt}(M) = \varnothing \right\} \\
\textsc{NeverDiverge} &:= \left\{ \langle M\rangle \mid \textsc{Diverge}(M) = \varnothing \right\}
\end{aligned}$$

- Describe an algorithm (really a Turing machine) $M'$ that decides $L'$, using a Turing machine $M$ that decides $L$ as a black box. Typically this algorithm has the following form:

  Given a string $w$, transform it into another string $x$,
  such that $M$ accepts $x$ if and only if $w \in L'$.

- Prove that your Turing machine is correct. This almost always requires two separate steps:
  - Prove that if $M$ accepts $w$ then $w \in L'$.
  - Prove that if $M$ rejects $w$ then $w \notin L'$.

---

Prove that the following languages are undecidable:

1. $\textsc{AcceptIllini} := \left\{ \langle M\rangle \;\middle|\; M \text{ accepts the string } \texttt{ILLINI} \right\}$
2. $\textsc{AcceptThree} := \left\{ \langle M\rangle \;\middle|\; M \text{ accepts exactly three strings} \right\}$
3. $\textsc{AcceptPalindrome} := \left\{ \langle M\rangle \;\middle|\; M \text{ accepts at least one palindrome} \right\}$