4. 1° The basic case: consider the smallest non-empty set \( S_1 = \{a_1\} \). Its only partial order \( R_1 = \{(a_1, a_1)\} \). (A partial order is a reflexive, anti-symmetric, and transitive binary relation.)

An element \( a \in S \) is a minimum whenever it doesn’t occur as the second element of a pair in the relation (except that the reflexive self-loop is allowed). Formally, \( a \) is a minimum iff:

\[
\forall a, b \in S : (b, a) \in R \Rightarrow a = b,
\]

The element \( a_1 \) fulfills this condition in \( R_1 \) so it is a minimum.

2° Induction hypothesis: Suppose that there exists a natural number \( n \) such that all partial orders on a set \( S \) have a minimum always when \( |S| < n \).

3° Induction step: Let \( S_n = \{a_1, \ldots, a_n\} \) be a set with \( n \) elements and \( R_n \) be an arbitrary partial order on \( S_n \). Choose now an arbitrary element \( a_i \in S_n \), remove it from \( S_n \) as well as all pairs that refer to it from \( R_n \):

\[
S'_n = S_n - \{a_i\}
\]

\[
R'_n = \{(a, b) \in R_n : a \neq a_i \land b \neq a_i\}
\]

Now \( R'_n \) is a partial order (prove this to yourself formally, it follows from transitivity of \( R_n \)). Since \( |S'_n| = n - 1 < n \), by induction hypothesis \( R'_n \) has at least one minimum that we now denote by \( a_{\text{min}} \).

Consider again \( R_n \). Now there are two possibilities:

i) If \((a_i, a_{\text{min}}) \notin R_n\), is \( a_{\text{min}} \) also a minimum of \( R_n \).

ii) If \((a_i, a_{\text{min}}) \in R_n\), then \( a_{\text{min}} \) can’t be a minimum. However, since \( a_{\text{min}} \) is the minimum of the partial order \( R'_n \) and a partial order is always transitive, there may not be a pair \((b, a_i), b \neq a_i\) in the relation. Thus, \( a_i \) is a minimum of \( R_n \) and the induction is complete.

5. Suppose that there are \( n \) persons in the party. We try to give every one a different number of acquaintances.

<table>
<thead>
<tr>
<th>Person</th>
<th>Acquaintances</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>\ldots</td>
<td>\ldots</td>
</tr>
<tr>
<td>( n )</td>
<td>( n - 1 )</td>
</tr>
</tbody>
</table>

We notice that the last person knows everybody but the first person doesn’t know anybody. These two cases are in
conflict, so only \(n - 1\) different numbers are possible. Now by the pigeon-hole principle we know that it is not possible to allocate \(n\) persons into \(n - 1\) slots without having at least two persons in at least one slot so it is not possible for all persons to have a different number of acquaintances.

6. We can define the concatenation \(v \circ w\) of strings \(v\) and \(w\) \((v, w \in \Sigma^*)\) as follows:

1° If \(|v| = 0\), then \(v \circ w = w\).

2° If \(|v| = n + 1 > 0\), we can write \(v\) in a form \(v = ua, u \in \Sigma^*, a \in \Sigma\).

Now we define \(v \circ w = u \circ aw\).

For example, \(\Sigma = \{a, b\}\), \(v = aba, w = bba\):

\[
v \circ w = aba \circ bba = ab \circ abba = a \circ babba = e \circ ababba = ababba
\]

7. We have to prove that if we reverse a string twice, we get the original string. The simplest way to do it is by induction. To simplify the proof we will use the identity \((wx)^R = x^Rw^R\) that is proved in the textbook.

1° The basic case: \(|w| = 0\), \((e^R)^R = e\) (by definition \(e^R = e\)).

2° Induction hypothesis: Suppose that the claim holds for all \(|w| \leq n, n > 0\).

3° Induction step: Let \(|w| = n + 1\). Now \(w\) can be written as \(w = ua, a \in \Sigma, u \in \Sigma^*, |u| = n\).

\[
(w^R)^R = ((ua)^R)^R
= (au^R)^R
= (u^R)^R(a^R)
= (u^R)^R(ea)^R
= (u^R)^R(e^R)
= (u^R)^Rw
= ua = w
\]

by induction hypothesis

8. A formal alphabet is a finite set of symbols. For example, the common alphabet \(\{a, b, \ldots, z\}\) and the binary alphabet \(\{0, 1\}\) are both also formal alphabets. Most often we use letters and numbers in alphabets, but we may also use any other symbols if necessary.

The notation \(\Sigma^*\) denotes all strings that can be formed using the symbols in \(\Sigma\) including the empty string \(e\). For example, if \(\Sigma = \{a, b\}\), then \(\Sigma^* = \{e, a, b, aa, ab, ba, bb, \ldots\}\). If \(\Sigma\) is not empty, \(\Sigma^*\) is necessarily infinite.

A formal language \(L\) is some subset \(L \subseteq \Sigma^*\). The most common notation in use is \(L = \{w \in \Sigma^* \mid w\text{ fulfills the property } P\}\). That is, \(w\) is in the language if it satisfies some property \(P\).
a) The set $L = \{ w \mid \text{for some } u \in \Sigma, w = uu^R u \}$ contains all six letter long words where the first two letters are equal to the last two letters and the middle part contains the same string reversed. The notation $u \in \Sigma$ denotes all two-letter words.

For example, the words $abbaab \ (u = ab)$ and $aaaaaa \ (u = aa)$ belong to $L$. On the other hand, $w = abbbba \notin L$. Since there are only a finite number of two-letter words, $L$ too is finite.

b) The language $L = \{ w \mid ww = www \}$ contains only the empty word $e$. By the condition $2|w| = 3|w|$ that is only possible when $|w| = 0$ and $w = e$.

c) The language $L = \{ w \mid \text{for some } u, v \in \Sigma^*, uvw = wvu \}$ contains all words ($L = \Sigma^*$). We see that if we choose $u = v = e$, then $e \circ e \circ w = w = w \circ e \circ e$ and the condition is fulfilled.

d) The language $L = \{ w \mid \text{for some } u \in \Sigma^*, wuw = uu \}$ contains for example $aa \ (u = aaa)$ and $aaaa \ (u = aaaaaa)$. The condition is that $w$ is either all $a$ or all $b$ and $3 \cdot |w|$ has to be divisible by two. The string $ab$ does not belong to the language.