1.

Der Fehler liegt in der Abschätzung

$$2^{n-1} + 2 \leq 2^{n-1} + 2^{n-1},$$

da nach Induktionsbehauptung $n \geq 1$ gelten muss (sonst kein Induktionsschluss möglich). Für $n = 1$ gilt allerdings obige Abschätzung nicht, da

$$2 \not\leq 2^{n-1} = 1.$$

2.

SATZ: $\forall n > 2 : n^3 > 3n + 3$.

BEWEIS:

Induktionsanfang: Sei $n = 3$, so gilt $n^3 = 27 > 3n + 3 = 12$

Induktionsvoraussetzung: $n^3 > 3n + 3$

Induktionsbehauptung: $n^3 > 3n + 3 \Rightarrow (n+1)^3 > 3(n+1) + 3$

$$\begin{aligned} (n+1)^3 &> 3(n+1) + 3 \\ n^3 + 3n^2 + 3n + 1 &> 3n + 3 + 3 \\ (n^3) + 3n^2 + 3n + 1 &> (3n + 3) + 3 \end{aligned}$$

Hieraus ist dann leicht ersichtlich, dass, da $n^3 > 3n + 3$ lt. Voraussetzung gilt, obige Ungleichung ebenfalls erfüllt ist. Die rechte Seite wird jeweils um 3 größer und die linke um $3n^2 + 3n + 1$. Da die linke Seite monoton wächst und die rechte konstant bleibt, wird diese Ungleichung immer erfüllt sein (sie ist es ja schon für $n = 3$). ■

3.

SATZ: $\forall n \in \mathbb{N} : 2^{n+2} + 3^{2n+1} = [0]_7$, wobei hier mit den eckigen Klammern und dem Index gemeint ist, dass das Ergebnis innerhalb der Restklasse modulo 7 zu sehen ist.

BEWEIS:

Induktionsanfang: Sei $n = 0$, so gilt $2^{n+2} + 3^{2n+1} = 2^2 + 3 = 7 = [0]_7$.

Induktionsvoraussetzung: $2^{n+2} + 3^{2n+1} = [0]_7$

Induktionsbehauptung: $2^{n+2} + 3^{2n+1} = [0]_7 \Rightarrow 2^{(n+1)+2} + 3^{2(n+1)+1} = [0]_7$

$$\begin{aligned} 2^{(n+1)+2} + 3^{2(n+1)+1} &= 2 \cdot 2^{n+2} + 3^2 \cdot 3^{2n+1} \\ &= [2]_7 \cdot 2^{n+2} + [2]_7 \cdot 3^{2n+1} \\ &= [2]_7 \cdot (2^{n+2} + 3^{2n+1}) \end{aligned}$$

Hier haben wir (lt. Induktionsvoraussetzung) ein Vielfaches einer durch 7 teilbaren Zahl, welche wieder durch 7 teilbar ist. ■

4.

Der Fehler liegt hier in der Anwendung des Induktionsschrittes, da wir nur dann zwei *verschiedene* Pferde entfernen können, wenn unsere Menge mindestens zwei Elemente hat. Damit funktioniert dieser Induktionsschritt nicht für alle n, sondern nur für diejenigen $n \geq 2$.

5.

SATZ: „Jeder Boolesche Ausdruck der Sprache W, in dem keine Vergleiche, keine Konstanten und kein Negationsoperator vorkommen, liefert in einem Zustand, der jeder Booleschen Variablen den Wert true zuweist, immer den Wert true."

BEWEIS:

Induktionsanfang:

$$\begin{aligned} a &= true \\ a \text{ AND } b &= true \\ a \text{ OR } b &= true \end{aligned}$$

Hier seien a bzw. b beliebige Variable. Diese Gleichungen folgen alle aus den Voraussetzungen des Satzes.

Induktionsvoraussetzung: Obiger Satz gilt für alle booleschen Ausdrücke der Spielsprache W.

Induktionsbehauptungen: Wenn für beliebige boolesche Ausdrücke $\mathcal{A}$ und $\mathcal{B}$ gilt:

$$\begin{aligned} \mathcal{A} \text{ AND } \mathcal{B} &= true \\ \mathcal{A} \text{ OR } \mathcal{B} &= true \end{aligned}$$

So gilt auch

$$\begin{aligned} (\mathcal{A} \text{ AND } \mathcal{B}) \text{ AND } c &= true \\ (\mathcal{A} \text{ OR } \mathcal{B}) \text{ AND } c &= true \\ (\mathcal{A} \text{ AND } \mathcal{B}) \text{ OR } c &= true \\ (\mathcal{A} \text{ OR } \mathcal{B}) \text{ OR } c &= true \end{aligned}$$

Für beliebige boolesche Variablen c:

$$\begin{aligned} (\mathcal{A} \text{ AND } \mathcal{B}) \text{ AND } c &\underset{\text{n. Vor.}}{=} true \text{ AND } c \\ &\underset{\text{n. Vor.}}{=} true \end{aligned}$$

Analog gilt dies auch für die anderen oben aufgeführten Terme. Im Grunde: Da jeder Ausdruck *true* wird und jede Variable, die wir damit verknüpfen können, ebenfalls *true* ist, wird auch die Verknüpfung wieder *true* liefern (notfalls aufgrund Aussagenlogik bzw. der für W geltenden Axiome).