Aufgabe 1

a)

Zustand: Menge von Stapeln mit mindestens 1 und höchstens 3 Elementen, jeder Stapel ist als Stack von Kisten organisiert, wobei jede Kiste durch ihre Nummer repräsentiert wird (vorne = oben)
Pop/1 sei eine Funktion, die einen um 1 Element reduzierten Stack zurückliefert
Push/2 liefert einen um das angegebene Element erweiterten Stack zurück
- und + seien Mengendifferenz und Aufnahmen eines Elements in eine Menge

Startzustand: S0={ [5,3,4,1,2,6] }
Zielzustand: SZ={ [1,2,3,4,5,6] }
Aktionen:

Ablegen_1(S): Zn → Zn+1 // 1 Element auf den Boden legen
S ∊ Zn, x=top(S)
S1 = pop(S)
Zn+1 = Zn – S + S1 + [x]

Ablegen_2(S): Zn → Zn+1 // 2 Elemente auf den Boden legen
S ∊ Zn, x=top(S), y=top(pop(S))
S1 = pop(pop(S))
Zn+1 = Zn – S + S1 + [x,y]

Umlegen_1(Sa,Sb): Zn → Zn+1 // 1 Element auf einen anderen Stapel legen
Sa ∊ Zn, Sb ∊ Zn, Sa≠Sb, x=top(Sa)
Sa1 = pop(S), Sb1 = push(x,Sb)
Zn+1 = Zn – Sa + Sa1 – Sb + Sb1

Umlegen_2(Sa,Sb): Zn → Zn+1 // 2 Elemente auf einen anderen Stapel legen
Sa ∊ Zn, Sb ∊ Zn, Sa≠Sb, x=top(Sa), y=top(pop(Sa))
Sa1 = pop(pop(Sa)), Sb1 = push(x, push(y,Sb))
Zn+1 = Zn – Sa + Sa1 – Sb + Sb1

b)

```
      _ {[3,4,1,2,6], [5]}
     |
S0--|
     |_ {[4,1,2,6], [3,5]}
```

(bei anderer als Mengenrepräsentation existieren 4 Folgezustände, abhängig vom Zielort des Teils oder 2er-Stapels)

c)

Die sortierte Liste enthält Pfade, keine Knoten. f=g+h berechnet sich aus der Heuristik plus 1 für jede durchgeführte Aktion (wegen uniformer Kosten).

Iteration	sortierte Liste
0	<{[2,3,1]}>-1
1	<{[2,3,1]},{[2,3],[1]}>-2, <{[2,3,1]},{[3,1],[2]}>-3
2	<{[2,3,1]},{[2,3],[1]},{[1,2,3]}>-2, <{[2,3,1]},{[3,1],[2]}>-3, <{[2,3,1]},{[2,3],[1]},{[1],[2],[3]}>-4

in 2. wird der Pfad <{[2,3,1]},{[2,3],[1],{[2,3,1]}>4 wird entfernt, weil er einen Zyklus darstellt.

Aufgabe 2

a)

1	2 3 4	2 3 4	2 4
2	1 3 4	1 3 4	1 4
4	1 2	1 2	3
3	1 2 4	1 2 4	1 2 4

Durch das Entfernen inkonsistenter Werte sind die Variablen M(1,1) und M(1,2) bereits belegt.

b)

Minimum Remaining Values (aller noch nicht belegten Variablen) in Kombination mit Most Constrained Variable: MRVen sind {M(2,2),M(3,2),M(4,3),M(4,4)} mit jeweils 2-stelligen Wertebereichen. MCV als Tie-Breaker bringt keine eindeutige Entscheidung: MCVen sind {M(2,2),M(3,2),M(4,3),M(4,4)} mit jeweils 4 Constraints.
(Alternative: MCVen sind die 8 Variablen M(2,1), …, M(3,4) mit jeweils 5 Constraints zu noch nicht belegten Variablen. (Die übrigen Variablen haben eine Grad-Heuristik von 4).

c)

1	2 3 4	2 3 4	2 4
2	3 4	1 3 4	1 4
4	1	2	3
3	2 4	1 2 4	1 2 4

Belege M(2,2) mit 1. Forward Checking führt in die rechts abgebildete Situation. M(3,2) ist belegt, {M(2,1),M(2,3),M(2,4)} haben jeweils um den Wert „1“ reduzierte Wertebereiche. Die übrigen Domains werden nicht verändert.

Aufgabe 3

a/1	~ Aufgabenblatt
u/1	~ unlösbar
s/1	~ Student
b/2	~ bekommen
sch/1	~ Schein
l/2	~ lösen
a/2	~ abgeben
k/2	~ Kommilitone
v/3	~ verraten
L	~ diese Lösung
l/1	~ Lösung

a)

$\exists x,y\ a(x) \wedge a(y) \wedge x \neq y \wedge (\forall z\ a(z) \rightarrow z=x \vee z=y)$

b)

$\neg\exists x\ a(x) \wedge u(x)$ oder $\forall x\ a(x) \rightarrow \neg u(x)$

c)

$(\forall x\ s(x) \rightarrow \forall y\ a(y) \rightarrow l(x,y)) \rightarrow \exists z\ sch(z) \wedge b(x,z)$

$(\forall x,y\ s(x) \wedge a(y) \rightarrow l(x,y)) \rightarrow \exists z\ sch(z) \wedge b(x,z)$ (nicht ganz korrekt)

Wenn „genau jeder Student” bzw. „nur Studenten, die“ gemeint ist, dann Äquivalenz verwenden.

d)

$\forall x\ s(x) \rightarrow \exists y,z\ a(y) \wedge a(z) \wedge y{\neq}z \wedge a(s,y) \wedge a(s,z)$

e)

$\neg\exists x\ s(x) \wedge \forall y\ k(x,y) \rightarrow v(x,y,L) \wedge l(L)$

Aufgabe 4

K1 = { $\neg P(x,1,f(x))$, $\neg Q(x,x)$, $\neg R(x,x)$ }
K2 = { $P(y,z,f(z))$, $Q(y,f(y))$, $R(1,y)$ }

1. P gegen ¬P resolvieren:
   {y/x, z/1, x/1} → { $\neg Q(1,1)$, $\neg R(2,1)$, $Q(1,f(1))$, $R(1,1)$ }
2. Q gegen ¬Q resolvieren: geht nicht, da nicht unifizierbar (x und g(x) nicht unifizierbar)
3. R gegen ¬R resolvieren:
   { x/2,y/2 } → { $\neg P(2,1,f(2))$, $\neg Q(2,2)$, $P(2,z,f(z))$, $Q(2,f(2))$ }

Aufgabe 5

a)

1. Konjunktive Normalform der Wissensbasis bilden, 2. Anfrage negieren, 3. Resolventen bilden, bis leere Resolvente ableitbar oder kein neues Wissen generierbar

b)

1. KNF bilden

1.	{ $\neg G(x)$, $\neg S(x)$, $GU(x)$ }	← $\forall x\ G(x) \wedge S(x) \rightarrow GU(x)$
2.	{ $\neg M(x)$, $S(x)$ }	← $\forall x\ M(x) \rightarrow S(x)$
3.	{ $\neg R(x)$, $M(x)$ }	← $\forall x\ R(x) \rightarrow M(x)$
4.	{ $\neg R(x)$, $\neg GF(x)$, $R(x)$ }	← $\forall x\ \forall y\ R(x) \wedge GF(x,y) \rightarrow R(y)$
5.	{ $G(A1)$ }	← $G(A1)$
6.	{ $GF(A1)$ }	← $GF(S1, A1)$
7.	{ $R(S1)$ }	← $R(S1)$

2. Anfrage negieren und in KNF überführen

8.	{ $\neg GU(A1)$ }	← $\neg GU(A1)$

3. Resolventen bilden

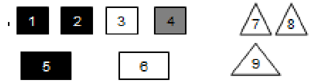
8&1	{x/A1}	→ { $\neg G(A1)$, $\neg S(A1)$ }	(9)
9&5	{}	→ { $\neg S(A1)$ }	(10)
10&2	{x/A1}	→ { $\neg M(A1)$ }	(11)
11&3	{x/A1}	→ { $\neg R(A1)$ }	(12)
12&4	{x/z,y/A1}	→ { $\neg GF(z, A1)$, $\neg R(z)$ }	(13)
13&6	{z/S1}	→ $\neg R(S1)$	(14)

14&7 { } → { }

Aufgabe 6 - Planung

a)

Wir nummerieren die Bausteine folgendermaßen:



Prädikate:

farbe/1	: Typprädikat
größe/1	: Typprädikat
etage/2	: Baustein im Haus mit Etage und Bausteinnummer
höhe/1	: aktuelle Höhe des zu bauenden Hauses
block/3	: Block mit Nummer, Größe und Farbe.
dach/2	: Dach mit Nummer und Größe
n/2	: Nachfolgerprädikat auf natürlichen Zahlen
ge/2	: größer-gleich-Relation (falls mehr als nur 2 Größen vorhanden sind)
v/1	: bezeichnet ein verfügbares Bauteil (eines, das noch nicht verbaut ist)

Bemerkungen: statt v/1 man könnte alternativ auch einen Ort "Kiste" als Konstante definieren und beispielsweise in/2 verwenden. Die Blocknummer ist wichtig, weil mehrere gleiche Blöcke zu unterscheiden sind – dasselbe gilt für Dächer. Dächer haben keine Farbe, weil diese beim Häuserbauen keine Rolle spielt.

Konstanten:

S,G,W	: Farben
1,2	: Größen (klein, groß)
1,...9	: IDs für Bausteine

Bemerkung: 1 und 2 sind sowohl Größen, als auch IDs. Diese „Überladung“ von Konstanten bringt hier keine Probleme. Bei dieser Modellierung von Größen mittels Zahlen in Verbindung mit der größer-gleich-Relation kann man auch weitere Blockgrößen einführen.

b)

S0: { block(1), block(2), ..., block(6), dach(7), dach(8), dach(9),
größe(1,1), größe(2,1), ..., größe(6,2), größe(7,1), größe(8,1), größe(9,2),
farbe(1,S), farbe(2,S), ..., farbe(6,W),
farbe(S), farbe(G), farbe(W), größe(1), größe(2), n(0,1), n(1,2), n(2,3),
höhe(0),
ge(1,1), ge(2,1), ge(2,2)
v(1),v(2),...,v(9)
}

c)

SZ:{etage(1,x),etage(2,y),etage(3,z),dach(z)}
alternativ: {höhe(3),etage(3,x),dach(x)}

d)

ACT stack0(x) // ein beliebiges Bauteil als 1. Etage verbauen
PRE höhe(0), v(x)

ADD höhe(1), etage(1,x),
DEL v(x), höhe(0)

ACT stackb(x) // einen verfügbaren Block auf einen anderen Block legen
PRE höhe(h), etage(h,b), v(x), größe(b,g), farbe(b,f), block(x), größe(x,gx), farbe(x,f), ge(g,gx), n(h,h1)
ADD höhe(h1), etage(h1,x)
DEL höhe(h), v(x)

Bemerkung: die Überprüfung der Farbe stellt sicher, dass b kein Dach sein kann.

ACT stackd(x) // ein verfügbares, passendes Dach auf die aktuelle Etage legen
PRE höhe(h), etage(h,b), block(b), v(x), dach(x), größe(x,g), größe(b,g), n(h,h1)
ADD höhe(h1), etage(h1,x)
DEL höhe(h), v(x)

e)

Lösungsplan: stack0(1),stackb(2),stackd(7)

f)

Relevant und konsistent sind alle Aktionen, die eines der Teilziele erfüllen. dach/1 wird von keiner Aktion generiert (ist bereits erfüllt, geeignete Variablenbelegung vorausgesetzt). Gesucht sind also alle Aktionen, die eine etage(1,x) in ihrer ADD haben: Dies betrifft sowohl stack0 als auch stackb und stackd mit allen möglichen Belegungen für die Variable x (also alle Individuenkonstanten (insgesamt 12 Konstanten) --> 12*3=36 Aktionen