

```
1: 474 Assessed Coursework: Summary for dvn14 of i4
2: -----
3:
4:   Public Tests:
5:     Part 1 (No-loops):          6 / 6
6:     Part 1 (Loops):            6 / 6
7:     Part 2:                    1 / 1
8:     Part 3 (Only Attacks):     1 / 4
9:     Part 3 (Only Supports):    0 / 4
10:    Part 3 (Attacks and Supports): 1 / 4
11:    Part 4 (Arguments):         0 / 3
12:    Part 4 (Attacks):           0 / 2
13:
```

```
1: %add below the definition of grounded/1
2: %(i.e. your answer to Part 1)
3: % do not include any examples
4:
5: :- dynamic(attacks/2).
6:
7: attackers(I,A) :- findall(X,attacks(X,I),A).
8:
9: defence_from_attacks(_,[]) :- !.
10: defence_from_attacks(X,[H|T]) :-
11:     attacks(Y,H),
12:     \+ attacks(Y,X),
13:     grounded(Y),
14:     defence_from_attacks(X,T).
15:
16: %checks_self_attacks
17: grounded(X) :-
18:     attacks(X,X),!,fail.
19:
20: %checks_attacks_between_themselves
21: grounded(X) :-
22:     attacks(X,Y),
23:     attacks(Y,X),!,fail.
24:
25: %checks_unattacked
26: grounded(X) :-
27:     argument(X),
28:     \+ attacks(_,X),!.
29:
30: %checks_defence_against_attacks
31: grounded(X) :-
32:     argument(X),
33:     attackers(X,A),
34:     defence_from_attacks(X,A).
```

```

1: %add below the definition of strength/2
2: %(i.e. your answer to Part 3)
3: % do not include any examples
4:
5: :- dynamic(supports/2).
6: :- dynamic(attacks/2).
7:
8: supporters(I,S) :- findall(X,supports(X,I),S).
9: attackers(I,A) :- findall(X,attacks(X,I),A).
10:
11: base_function(V1,V2,BFV) :-
12:     BFV is V1+V2-V1*V2.
13:
14: combination_function(Vo,Vs,Va,C) :-
15:     Va >= Vs,
16:     C is Vo-Vo*abs(Vs-Va).
17:
18: combination_function(Vo,Vs,Va,C) :-
19:     Va < Vs,
20:     C is Vo+(1-Vo)*abs(Vs-Va).
21:
22: strength_function([],V,K) :- !,
23:     K is V.
24:
25: strength_function([H|T],0,K) :- !,
26:     calc_strength(H,Val),
27:     base_function(0,Val,BFV),
28:     strength_function(T,BFV,K).
29:
30: strength_function([H|T],V,K) :-
31:     calc_strength(H,Val),
32:     base_function(V,Val,BFV),
33:     strength_function(T,BFV,K).
34:
35: calc_strength(I,Val) :-
36:     \+ attacks(_,I),
37:     \+ supports(_,I), !,
38:     base(I,B),
39:     Val is B.
40:
41: calc_strength(I,Val) :-
42:     \+ attacks(_,I),
43:     supporters(I,S), !,
44:     strength_function(S,0,Vs),
45:     base(I,B),
46:     combination_function(B,Vs,0,C),
47:     Val is C.
48:
49: calc_strength(I,Val) :-
50:     \+ supports(_,I),
51:     attackers(I,A), !,
52:     strength_function(A,0,Va),
53:     base(I,B),
54:     combination_function(B,0,Va,C),
55:     Val is C.
56:
57: calc_strength(I,Val) :-
58:     supporters(I,S),
59:     attackers(I,A), !,
60:     strength_function(A,0,Va),
61:     strength_function(S,0,Vs),
62:     base(I,B),
63:     combination_function(B,Vs,Va,C),
64:     Val is C.
65:
66: strength(I,X) :-
67:     argument(I),

```

```

68:     calc_strength(I,Y),
69:     X == Y.

```

```
1: %add below your answer to Part 2
2:
3: %arguemnts
4: argument(a0).
5: argument(a1).
6: argument(a2).
7: argument(a3).
8: argument(a4).
9: argument(a5).
10: argument(a6).
11: argument(a7).
12: argument(a8).
13: argument(a9).
14: argument(a10).
15: argument(a11).
16: argument(a12).
17: argument(a13).
18: argument(a14).
19: argument(a15).
20: argument(a16).
21: argument(a17).
22:
23: %attacks
24: attacks(a1,a0).
25: attacks(a2,a1).
26: attacks(a3,a0).
27: attacks(a3,a2).
28: attacks(a5,a0).
29: attacks(a7,a4).
30: attacks(a7,a6).
31: attacks(a8,a0).
32: attacks(a9,a8).
33: attacks(a13,a0).
34: attacks(a14,a13).
35: attacks(a15,a14).
36: attacks(a17,a3).
37:
38: %supports
39: supports(a2,a0).
40: supports(a2,a4).
41: supports(a3,a1).
42: supports(a6,a4).
43: supports(a8,a1).
44: supports(a10,a1).
45: supports(a10,a3).
46: supports(a11,a10).
47: supports(a12,a1).
48: supports(a13,a1).
49: supports(a15,a13).
50: supports(a17,a0).
51: supports(a17,a16).
52:
```

474 Assessed Coursework/Public Tests**TestLog.txt: 1/3****Devin Nanayakkara - dvn14: i4**

```
1: % compiling /root/labcat/labcat/engines/lib/prolog/automarker_ft.pl... 53: yes
2: % loading /usr/lib/sicstus4.3.5/bin/sp-4.3.5/sicstus-4.3.5/library/timeout.po... 54: yes
3: % module timeout imported into user 55: yes
4: % loading /usr/lib/sicstus4.3.5/bin/sp-4.3.5/sicstus-4.3.5/library/types.po... 56: yes
5: % module types imported into timeout 57: yes
6: % loaded /usr/lib/sicstus4.3.5/bin/sp-4.3.5/sicstus-4.3.5/library/types.po in mo 58: yes
dule types, 0 msec 4112 bytes 59: yes
7: % loading foreign resource /usr/lib/sicstus4.3.5/bin/sp-4.3.5/sicstus-4.3.5/libr 60: yes
ary/x86_64-linux-glibc2.17/timeout.so in module timeout 61: yes
8: % loaded /usr/lib/sicstus4.3.5/bin/sp-4.3.5/sicstus-4.3.5/library/timeout.po in m 62: yes
odule timeout, 0 msec 50464 bytes 63: yes
9: % compiled /root/labcat/labcat/engines/lib/prolog/automarker_ft.pl in module user, 64: yes
180 msec 1044720 bytes 65: yes
10: SICStus 4.3.5 (x86_64-linux-glibc2.17): Tue Dec 6 10:41:06 PST 2016 66: yes
11: Licensed to SP4.3doc.ic.ac.uk 67: yes
12: % compiling /tmp/d20180208-36-33rr5r/src/argumentation_autopatch.pl... 68: yes
13: % loading /usr/lib/sicstus4.3.5/bin/sp-4.3.5/sicstus-4.3.5/library/lists.po... 69: yes
14: % module lists imported into user 70: yes
15: % module types imported into lists 71: yes
16: % loaded /usr/lib/sicstus4.3.5/bin/sp-4.3.5/sicstus-4.3.5/library/lists.po in mod 72: yes
ule lists, 10 msec 126544 bytes 73: yes
17: % compiled /tmp/d20180208-36-33rr5r/src/argumentation_autopatch.pl in module user, 74: yes
10 msec 150416 bytes 75: yes
18: yes 76: yes
19: % compiling /tmp/d20180208-36-33rr5r/src/grounded.pl... 77: yes
20: % compiled /tmp/d20180208-36-33rr5r/src/grounded.pl in module submittedGrounded, 0 78: yes
msec 2816 bytes 79: yes
21: yes 80: yes
22: % compiling /tmp/d20180208-36-33rr5r/src/df_quad.pl... 81: yes
23: % compiled /tmp/d20180208-36-33rr5r/src/df_quad.pl in module submittedDFQUAD, 0 ms 82: yes
ec 10656 bytes 83: yes
24: yes 84: yes
25: % compiling /tmp/d20180208-36-33rr5r/src/angrymen.pl... 85: yes
26: % compiled /tmp/d20180208-36-33rr5r/src/angrymen.pl in module submittedANGRY, 0 ms 86: yes
ec 9152 bytes 87: yes
27: yes 88: yes
28: % compiling /tmp/d20180208-36-33rr5r/src/aba.pl... 89: yes
29: % compiled /tmp/d20180208-36-33rr5r/src/aba.pl in module submittedABA, 0 msec 0 by 90: yes
tes 91: yes
30: yes 92: yes
31: % compiling /tmp/d20180208-36-33rr5r/src/ma_grounded.pl... 93: yes
32: % compiled /tmp/d20180208-36-33rr5r/src/ma_grounded.pl in module modelGrounded, 10 94: yes
msec 4432 bytes 95: yes
33: yes 96: yes
34: % compiling /tmp/d20180208-36-33rr5r/src/ma_df_quad.pl... 97: yes
35: % compiled /tmp/d20180208-36-33rr5r/src/ma_df_quad.pl in module modelDFQUAD, 0 mse 98: yes
c 6256 bytes 99: yes
36: yes 100: yes
37: % compiling /tmp/d20180208-36-33rr5r/src/ma_angrymen.pl... 101: yes
38: % compiled /tmp/d20180208-36-33rr5r/src/ma_angrymen.pl in module modelANGRY, 0 mse 102: yes
c 7056 bytes 103: yes
39: yes 104: yes
40: % compiling /tmp/d20180208-36-33rr5r/src/ma_aba.pl... 105: yes
41: % loading /usr/lib/sicstus4.3.5/bin/sp-4.3.5/sicstus-4.3.5/library/sets.po... 106: yes
42: % module sets imported into modelABA 107: yes
43: % module lists imported into sets 108: yes
44: % loaded /usr/lib/sicstus4.3.5/bin/sp-4.3.5/sicstus-4.3.5/library/sets.po in modu 109: yes
le sets, 0 msec 23456 bytes 110: yes
45: % compiled /tmp/d20180208-36-33rr5r/src/ma_aba.pl in module modelABA, 0 msec 28112 111: yes
bytes 112: yes
46: yes 113: yes
47: yes 114: yes
48: yes 115: yes
49: yes 116: yes
50: yes 117: yes
51: yes 118: yes
52: yes 119: yes
```

474 Assessed Coursework/Public Tests**TestLog.txt: 2/3****Devin Nanayakkara - dvn14: i4**

120: yes
121: yes
122: yes
123: yes
124: yes
125: yes
126: yes
127: yes
128: yes
129: yes
130: yes
131: yes
132: yes
133: yes
134: yes
135: yes
136: yes
137: yes
138: yes
139: yes
140: yes
141: yes
142: yes
143: yes
144: yes
145: yes
146: yes
147: yes
148: yes
149: yes
150: yes
151: yes
152: yes
153: yes
154: yes
155: yes
156: yes
157: yes
158: yes
159: yes
160: yes
161: yes
162: yes
163: yes
164: yes
165: yes
166: yes
167: yes
168: yes
169: yes
170: yes
171: yes
172: yes
173: yes
174: yes
175: yes
176: yes
177: yes
178: yes
179: yes
180: yes
181: yes
182: yes
183: yes
184: yes
185: yes
186: yes

187: yes
188: yes
189: yes
190: yes
191: yes
192: yes
193: yes
194: yes
195: yes
196: yes
197: yes
198: yes
199: yes
200: yes
201: yes
202: yes
203: yes
204: yes
205: yes
206: yes
207: yes
208: yes
209: yes
210: yes
211: yes
212: yes
213: yes
214: yes
215: yes
216: yes
217: yes
218: yes
219: yes
220: yes
221: yes
222: yes
223: yes
224: yes
225: yes
226: yes
227: yes
228: yes
229: yes
230: yes
231: yes
232: yes
233: yes
234: yes
235: yes
236: yes
237: yes
238: yes
239: yes
240: yes
241: yes
242: yes
243: yes
244: yes
245: yes
246: yes
247: yes
248: yes
249: yes
250: yes
251: yes
252: yes
253: yes

254: yes
255: yes
256: yes
257: yes
258: yes
259: yes
260: yes
261: yes
262: yes
263: yes
264: yes
265: yes
266: yes
267: yes
268: yes
269: yes
270: yes
271: yes
272: yes
273: yes
274: ! Existence error in submittedABA:argument/1
275: ! procedure submittedABA:argument/1 does not exist
276: ! goal: submittedABA:argument((a,_383))
277: ! Existence error in submittedABA:argument/1
278: ! procedure submittedABA:argument/1 does not exist
279: ! goal: submittedABA:argument((y,_383))
280: ! Existence error in submittedABA:argument/1
281: ! procedure submittedABA:argument/1 does not exist
282: ! goal: submittedABA:argument((z,_383))
283: yes
284: ! Existence error in submittedABA:attacks/2
285: ! procedure submittedABA:attacks/2 does not exist
286: ! goal: submittedABA:attacks((_383,_385),(x,_391))
287: ! Existence error in submittedABA:attacks/2
288: ! procedure submittedABA:attacks/2 does not exist
289: ! goal: submittedABA:attacks((y,_385),(y,_391))
290: yes

```

1: =====
2:      474 Arg&MAS::Assessed Coursework
3:      Submission: dvn14
4: =====
5:
6: =====
7: Part 1 (No-loops)
8: ----- Test 1 ::a not in GE -----
9:
10: | ? grounded(a).
11: no      %% correct
12:
13: ----- Test 2 ::b not in GE -----
14:
15: | ? grounded(b).
16: no      %% correct
17:
18: ----- Test 3 ::c in GE -----
19:
20: | ? grounded(c).
21: yes     %% correct
22:
23: ----- Test 4 ::d not in GE -----
24:
25: | ? grounded(d).
26: no      %% correct
27:
28: ----- Test 5 ::e in GE -----
29:
30: | ? grounded(e).
31: yes     %% correct
32:
33: ----- Test 6 ::f in GE -----
34:
35: | ? grounded(f).
36: yes     %% correct
37:
38:
39: =====
40: Part 1 (No-loops)
41: TESTS PASSED: 6 / 6 ; MARKS: 6 / 6
42: =====
43:
44: =====
45: Part 1 (Loops)
46: ----- Test 1 ::a not in GE -----
47:
48: | ? grounded(a).
49: no      %% correct
50:
51: ----- Test 2 ::b not in GE -----
52:
53: | ? grounded(b).
54: no      %% correct
55:
56: ----- Test 3 ::c not in GE -----
57:
58: | ? grounded(c).
59: no      %% correct
60:
61: ----- Test 4 ::d not in GE -----
62:
63: | ? grounded(d).
64: no      %% correct
65:
66: ----- Test 5 ::e not in GE -----
67:

```

```

68: | ? grounded(e).
69: no      %% correct
70:
71: ----- Test 6 ::f in GE -----
72:
73: | ? grounded(f).
74: yes     %% correct
75:
76:
77: =====
78: Part 1 (Loops)
79: TESTS PASSED: 6 / 6 ; MARKS: 6 / 6
80: =====
81:
82: =====
83: Part 2
84: ----- Test 1 ::correct set of arguments considered -----
85:
86: | ? arguments_differences.
87:
88: These arguments should not be mined:
89:
90: These arguments are missing:
91:
92: yes
93: These arguments should not be mined:
94:
95: These arguments are missing:
96:
97:      %% correct
98:
99:
100: =====
101: Part 2
102: TESTS PASSED: 1 / 1 ; MARKS: 1 / 1
103: =====
104:
105: =====
106: Part 3 (Only Attacks)
107: ----- Test 1 ::strength of d -----
108:
109: | ? strength(d,_947).
110: no      %% WRONG
111:
112: ----- Test 2 ::strength of c -----
113:
114: | ? strength(c,_947).
115: no      %% WRONG
116:
117: ----- Test 3 ::strength of b -----
118:
119: | ? strength(b,_947).
120: no      %% WRONG
121:
122: ----- Test 4 ::does a have strength 0.234375? -----
123:
124: | ? strength(a,0.234375).
125: yes     %% correct
126:
127:
128: =====
129: Part 3 (Only Attacks)
130: TESTS PASSED: 1 / 4 ; MARKS: 1 / 4
131: =====
132:
133: =====
134: Part 3 (Only Supports)

```



```

135: ----- Test 1 ::strength of d -----
136:
137: | ? strength(d,_947).
138: no          %% WRONG
139:
140: ----- Test 2 ::strength of c -----
141:
142: | ? strength(c,_947).
143: no          %% WRONG
144:
145: ----- Test 3 ::strength of b -----
146:
147: | ? strength(b,_947).
148: no          %% WRONG
149:
150: ----- Test 4 ::strength of a -----
151:
152: | ? strength(a,_947).
153: no          %% WRONG
154:
155:
156: =====
157: Part 3 (Only Supports)
158: TESTS PASSED: 0 / 4 ; MARKS: 0 / 4
159: =====
160:
161: =====
162: Part 3 (Attacks and Supports)
163: ----- Test 1 ::strength of d -----
164:
165: | ? strength(d,_947).
166: no          %% WRONG
167:
168: ----- Test 2 ::strength of c -----
169:
170: | ? strength(c,_947).
171: no          %% WRONG
172:
173: ----- Test 3 ::strength of b -----
174:
175: | ? strength(b,_947).
176: no          %% WRONG
177:
178: ----- Test 4 ::strength of a is 0.390625? -----
179:
180: | ? strength(a,0.390625).
181: yes          %% correct
182:
183:
184: =====
185: Part 3 (Attacks and Supports)
186: TESTS PASSED: 1 / 4 ; MARKS: 1 / 4
187: =====
188:
189: =====
190: Part 4 (Arguments)
191: ----- Test 1 ::argument for single assumptions -----
192:
193: | ? argument((a,[a])).
194: Test 1: FATAL ERROR!!
195:
196: ----- Test 2 ::argument for non-assumptions -----
197:
198: | ? argument((y,[b])).
199: Test 2: FATAL ERROR!!
200:
201: ----- Test 3 ::non- arguments -----

```

```

202:
203: | ? argument((z,_957)).
204:
205: Test 3: FATAL ERROR!!
206:
207: =====
208: Part 4 (Arguments)
209: TESTS PASSED: 0 / 3 (Fatal errors: 3); MARKS: 0 / 3
210: =====
211:
212: =====
213: Part 4 (Attacks)
214: ----- Test 1 ::no attacks against any arguments with claim x -----
215:
216: | ? attacks(_923,(x,_989)).
217: Test 1: FATAL ERROR!!
218:
219: ----- Test 2 ::self-attacking arguments -----
220:
221: | ? attacks((y,[b]),(y,[b])).
222:
223: Test 2: FATAL ERROR!!
224:
225: =====
226: Part 4 (Attacks)
227: TESTS PASSED: 0 / 2 (Fatal errors: 5); MARKS: 0 / 2
228: =====
229:
230:
231: ===== SUMMARY =====
232:
233: TESTS PASSED (dvn14): 15 / 30 (Fatal errors: 8); MARKS: 15 / 30
234:
235: =====

```