

## lib/main/set\_unittest.ath

```

1  load "set"
2  load "basic/testing"
3
4  open Set
5  open Testing
6
7  (Testing   "(eval 23 in [1 5 23 98])"
8    lambda () (eval 23 in [1 5 23 98])
9    [expecting
10     true
11    ])
12
13 (Testing   "(eval 23 in [1 5 98])"
14   lambda () (eval 23 in [1 5 98])
15   [expecting
16    false
17   ])
18
19 (Testing   "(eval 5 in [])"
20   lambda () (eval 5 in [])
21   [expecting
22    false
23   ])
24
25 (Testing   "(eval 5 in [5])"
26   lambda () (eval 5 in [5])
27   [expecting
28    true
29   ])
30
31 (Testing   "(eval [1 2] subset [3 2 4 1 5])"
32   lambda () (eval [1 2] subset [3 2 4 1 5])
33   [expecting
34    true
35   ])
36
37 (Testing   "(eval [1 2] subset [3 2])"
38   lambda () (eval [1 2] subset [3 2])
39   [expecting
40    false
41   ])
42
43 (Testing   "(eval [] subset [])"
44   lambda () (eval [] subset [])
45   [expecting
46    true
47   ])
48
49 (Testing   "(eval 1 ++ 2 ++ [] = 2 ++ 1 ++ [])"
50   lambda () (eval 1 ++ 2 ++ [] = 2 ++ 1 ++ [])
51   [expecting
52    true
53   ])
54
55 (Testing   "(eval 1 ++ 2 ++ 3 ++ 4 ++ [] = 3 ++ 2 ++ 1 ++ [])"
56   lambda () (eval 1 ++ 2 ++ 3 ++ 4 ++ [] = 3 ++ 2 ++ 1 ++ [])
57   [expecting
58    false
59   ])
60
61 (Testing   "(eval [1 2] proper-subset [2 3 1])"
62   lambda () (eval [1 2] proper-subset [2 3 1])
63   [expecting
64    true
65   ])
66
67 (Testing   "(eval [1 2] proper-subset [2 1])"
68   lambda () (eval [1 2] proper-subset [2 1])

```

```

69   [expecting
70     false
71   ])
72
73 (Testing  "(eval [1 2 3 2 5] - 2)"
74   lambda () (eval [1 2 3 2 5] - 2)
75   [expecting
76     [1 3 5]
77   ])
78
79 (Testing  "(eval [1 2 3] \\/ [4 5 6])"
80   lambda () (eval [1 2 3] \\/ [4 5 6])
81   [expecting
82     [1 2 3 4 5 6]
83   ])
84
85 (Testing  "(eval [1 2] \\/ [1 2])"
86   lambda () (eval [1 2] \\/ [1 2])
87   [expecting
88     [1 2]
89   ])
90
91 (Testing  "(eval [1 2 1] /\ [5 1 3])"
92   lambda () (eval [1 2 1] /\ [5 1 3])
93   [expecting
94     [1]
95   ])
96
97 (Testing  "(eval [1 2 1] /\ [5])"
98   lambda () (eval [1 2 1] /\ [5])
99   [expecting
100     []
101   ])
102
103 (Testing  "(eval 3 paired-with [2 8])"
104   lambda () (eval 3 paired-with [2 8])
105   [expecting
106     [(pair 3 2) (pair 3 8)]
107   ])
108
109 (Testing  "(eval [1 2] X ['foo 'bar 'car])"
110   lambda () (eval [1 2] X ['foo 'bar 'car])
111   [expecting
112     [(pair 1 'foo) (pair 1 'bar) (pair 1 'car)
113      (pair 2 'foo) (pair 2 'bar) (pair 2 'car)]
114   ])
115
116 (Testing  "(eval dom [('a @ 1) ('b @ 2) ('c @ 98)])"
117   lambda () (eval dom [('a @ 1) ('b @ 2) ('c @ 98)])
118   [expecting
119     ['a 'b 'c]
120   ])
121
122 (Testing  "(eval range [('a @ 1) ('b @ 2) ('c @ 98)])"
123   lambda () (eval range [('a @ 1) ('b @ 2) ('c @ 98)])
124   [expecting
125     [1 2 98]
126   ])
127
128 (Testing  "(eval 1 @ 2 composed-with [(2 @ 5) (7 @ 8) (2 @ 3)])"
129   lambda () (eval 1 @ 2 composed-with [(2 @ 5) (7 @ 8) (2 @ 3)])
130   [expecting
131     [(pair 1 5) (pair 1 3)]
132   ])
133
134 (Testing  "(eval 1 @ 2 composed-with [(7 @ 8) (9 @ 10)])"
135   lambda () (eval 1 @ 2 composed-with [(7 @ 8) (9 @ 10)])
136   [expecting
137     []
138   ])

```

```

139
140 (Testing  "(eval 1 @ 2 composed-with [])"
141   lambda () (eval 1 @ 2 composed-with [])
142   [expecting
143    []
144   ])
145
146 (Testing  "(eval [(('nyc @ 'boston) ('houston @ 'dallas) ('austin @ 'dc)) o
147                [(('boston @ 'montreal) ('dallas @ 'chicago) ('dc @ 'nyc)) o
148                [(('chicago @ 'seattle) ('montreal @ 'london))]"
149   lambda () (eval [(('nyc @ 'boston) ('houston @ 'dallas) ('austin @ 'dc)) o
150                [(('boston @ 'montreal) ('dallas @ 'chicago) ('dc @ 'nyc)) o
151                [(('chicago @ 'seattle) ('montreal @ 'london))
152   [expecting
153    [(pair 'nyc 'london) (pair 'houston 'seattle)]
154   ])
155
156 (Testing  "let {R1 := [(('nyc @ 'boston) ('austin @ 'dc));
157             R2 := [(('boston @ 'montreal) ('dc @ 'chicago) ('chicago @ 'seattle))}
158             (eval R1 o R2)"
159   lambda () let {R1 := [(('nyc @ 'boston) ('austin @ 'dc));
160             R2 := [(('boston @ 'montreal) ('dc @ 'chicago) ('chicago @ 'seattle))}
161             (eval R1 o R2)
162   [expecting
163    [(pair 'nyc 'montreal) (pair 'austin 'chicago)]
164   ])
165
166 (Testing  "(eval [(1 @ 'foo) (2 @ 'b) (1 @ 'bar)] restrict1 1)"
167   lambda () (eval [(1 @ 'foo) (2 @ 'b) (1 @ 'bar)] restrict1 1)
168   [expecting
169    [(pair 1 'foo) (pair 1 'bar)]
170   ])
171
172 (Testing  "(eval [(1 @ 'foo) (2 @ 'b) (3 @ 'c) (4 @ 'd) (1 @ 'bar)] ^ [1 2])"
173   lambda () (eval [(1 @ 'foo) (2 @ 'b) (3 @ 'c) (4 @ 'd) (1 @ 'bar)] ^ [1 2])
174   [expecting
175    [(pair 1 'foo) (pair 1 'bar) (pair 2 'b)]
176   ])
177
178 (Testing  "(eval [(1 @ 'a) (2 @ 'b) (3 @ 'c)] ** [1 3])"
179   lambda () (eval [(1 @ 'a) (2 @ 'b) (3 @ 'c)] ** [1 3])
180   [expecting
181    ['a 'c]
182   ])
183
184 (Testing  "(eval card [1 2 3] \\/ [4 7 8])"
185   lambda () (eval card [1 2 3] \\/ [4 7 8])
186   [expecting
187    6
188   ])

```