Test Case for HW7Q2

Some notes:

- All doors are bidirectional, but the fact will only provide one direction for each pair.
- There could be loops of room design, but each room would access at most once in a path.
- Please include all the following tests in your report.
- To ease grading, we will use letters to represent room names in test cases.
- Three categories of tests:
    - given R1, R2, infer Path
    - given R1, infer R2 and Path
    - given R2, infer R1 and Path

KB (facts):

```
door_between(a,b).
door_between(a,c).
door_between(a,d).
door_between(b,c).
door_between(b,e).
door_between(d,c).
door_between(d,e).
door_between(e,c).
```

looks like:

```
a --- b
|\   /|
|  c  |
|/   \|
d --- e
```

Queries:

- Given two rooms, look for all possible routes:

```
?- path_from(a,a,R). % keep it simple, same room return no path.
  R = [];
  false.

 ?- path_from(a,e,R).
  R = [a, b, c, d, e];
  R = [a, b, c, e];
  R = [a, b, e];
  R = [a, c, b, e];
  R = [a, c, d, e];
```

```
   R = [a, c, e];
   R = [a, d, c, b, e];
   R = [a, d, c, e];
   R = [a, d, e];
   false.
```

- Given a room for arrival, find all where and how you can get there:

```
?- path_from(X, a, R).
   R = [],
   X = a;
   R = [b, c, d, a],
   X = b;
   R = [b, c, e, d, a],
   X = b;
   R = [b, c, a],
   X = b;
   R = [b, e, c, d, a],
   X = b;
   R = [b, e, c, a],
   X = b;
   R = [b, e, d, c, a],
   X = b;
   R = [b, e, d, a],
   X = b;
   R = [d, c, b, a],
   X = d;
   R = [d, c, e, b, a],
   X = d;
   R = [d;, c, a],
   X = d;
   R = [d, e, c, b, a],
   X = d;
   R = [d, e, c, a],
   X = d;
   R = [d, e, b, c, a],
   X = d;
   R = [d, e, b, a],
   X = d;
   R = [e, c, b, a],
   X = e;
   R = [e, c, d, a],
   X = e;
   R = [e, c, a],
   X = e;
   R = [c, b, e, d, a],
   X = c;
```

```
   R = [c, b, a],
   X = c;
   R = [e, b, c, d, a],
   X = e;
   R = [e, b, c, a],
   X = e;
   R = [e, b, a],
   X = e;
   R = [c, d, e, b, a],
   X = c;
   R = [c, d, a],
   X = c;
   R = [e, d, c, b, a],
   X = e;
   R = [e, d, c, a],
   X = e;
   R = [e, d, a],
   X = e;
   R = [c, e, b, a],
   X = c;
   R = [c, e, d, a],
   X = c;
   R = [b, a],
   X = b;
   R = [c, a],
   X = c;
   R = [d, a],
   X = d;
   false.
```

- Given a room for departure, see where and how you can go:

```
?- path_from(a, X, R).
   R = [],
   X = a;
   R = [a, b, c, d, e],
   X = e;
   R = [a, b, c, e, d],
   X = d;
   R = [a, b, c, d],
   X = d;
   R = [a, b, c, e],
   X = e;
   R = [a, b, e, c, d],
   X = d;
   R = [a, b, e, d, c],
   X = c;
```

```
R = [a, b, e, c],
X = c;
R = [a, b, e, d],
X = d;
R = [a, b, c],
X = c;
R = [a, b, e],
X = e;
R = [a, c, b, e, d],
X = d;
R = [a, c, b, e],
X = e;
R = [a, c, d, e, b],
X = b;
R = [a, c, d, e],
X = e;
R = [a, c, e, b],
X = b;
R = [a, c, e, d],
X = d;
R = [a, c, b],
X = b;
R = [a, c, d],
X = d;
R = [a, c, e],
X = e;
R = [a, d, c, b, e],
X = e;
R = [a, d, c, e, b],
X = b;
R = [a, d, c, b],
X = b;
R = [a, d, c, e],
X = e;
R = [a, d, e, c, b],
X = b;
R = [a, d, e, b, c],
X = c;
R = [a, d, e, c],
X = c;
R = [a, d, e, b],
X = b;
R = [a, d, c],
X = c;
R = [a, d, e],
X = e;
R = [a, b],
X = b;
```

```
R = [a, c],
X = c;
R = [a, d],
X = d;
false.
```