In [10]:
```
# Solve(position) → { win, tie, lose, draw }
def Solve(position):
    if PrimitiveValue(position) == 'not_primitive':
        for move in GenerateMoves(position):
            if Solve(DoMove(position, move)) == 'lose':
                return 'win'
        return 'lose'
    return PrimitiveValue(position)
```

In [ ]:
```
```

In [1]:
```
%run Solver.ipynb
```

In [2]:
```
GAMENAME = '10-to-0-by-1-or-2'
STARTING_POS = 10
POSSIBLE_MOVES = {1, 2}
PRIMITIVE_POS = 0
```

In [3]:
```
def DoMove(position, move):
    return position - move
```

In [4]:
```
def GenerateMoves(position):
    return [move for move in POSSIBLE_MOVES if isLegal(position, move)]
```

In [5]:
```
def PrimitiveValue(position):
    if not isinstance(position, int) or position > STARTING_POS:
        print('Please enter a valid number!')
        return
    if position == PRIMITIVE_POS:
        return 'lose'
    else:
        return 'not_primitive'
```

In [6]:
```
def isLegal(position, move):
    return position - move >= PRIMITIVE_POS
```

In [9]:
```
%%capture cap
for i in range(STARTING_POS, -1, -1):
    print(i,  ": ",  Solve(i))
with open(GAMENAME + '_output.txt', 'w') as f:
    f.write(cap.stdout)
```

In [ ]:
```
```

```
10 :  win
9 :  lose
8 :  win
7 :  win
6 :  lose
5 :  win
4 :  win
3 :  lose
2 :  win
1 :  win
0 :  lose
```

```
In [1]: %run Solver.ipynb
```

```
In [2]: GAMENAME = '25-to-0-by-1-or-3-or-4'
        STARTING_POS = 25
        POSSIBLE_MOVES = {1, 3, 4}
        PRIMITIVE_POS = 0
```

```
In [3]: def DoMove(position, move):
            return position - move
```

```
In [4]: def GenerateMoves(position):
            return [move for move in POSSIBLE_MOVES if isLegal(position, move)]
```

```
In [5]: def PrimitiveValue(position):
            if not isinstance(position, int) or position > STARTING_POS:
                print('Please enter a valid number!')
            if position == PRIMITIVE_POS:
                return 'lose'
            else:
                return 'not_primitive'
```

```
In [6]: def isLegal(position, move):
            return position - move >= PRIMITIVE_POS
```

```
In [8]: %%capture cap
        for i in range(STARTING_POS, -1, -1):
            print(i,  ": ",  Solve(i))
        with open(GAMENAME + '_output.txt', 'w') as f:
            f.write(cap.stdout)
```

```
In [ ]:
```

```
25 :  win
24 :  win
23 :  lose
22 :  win
21 :  lose
20 :  win
19 :  win
18 :  win
17 :  win
16 :  lose
15 :  win
14 :  lose
13 :  win
12 :  win
11 :  win
10 :  win
9 :  lose
8 :  win
7 :  lose
6 :  win
5 :  win
4 :  win
3 :  win
2 :  lose
1 :  win
0 :  lose
```