

```
man = 0
```

```
lion = 1
```

```
goat = 2
```

```
grass = 3
```

```
def check_constraints(state):
```

```
    if state[lion] == state[goat] and state[man] != state[lion]:
```

```
        return False
```

```
    if state[goat] == state[grass] and state[man] != state[goat]:
```

```
        return False
```

```
    return True
```

```
def print_state(state):
```

```
    print("Starting Side: {} {} {} {}".format("Man " if not state[man] else "", "Lion " if not  
state[lion] else "", "Goat " if not state[goat] else "", "Grass " if not state[grass] else ""))
```

```
    print("Target Side: {} {} {} {}".format("Man " if state[man] else "", "Lion " if state[lion] else "",  
"Goat " if state[goat] else "", "Grass " if state[grass] else ""))
```

```
    print()
```

```
example_1 = (False, False, False, False)
```

```
example_2 = (False, True, True, False)
```

```
example_3 = (True, True, False, False)
```

```
print("Valid: Everyone is on the left side: {}".format(check_constraints(example_1)))
```

```
print_state(example_1)
```

```
print()
```

```
print("Invalid: The lion and goat are left without the farmer:
```

```
{}".format(check_constraints(example_2)))
```

```
print_state(example_2)
```

```
print()
```

```
print("Invalid: The goat and grass are left without the farmer:
```

```
{}".format(check_constraints(example_3)))
```

```
print_state(example_3)
```

```
start = (False, False, False, False)
```

```
goal = (True, True, True, True)
```

```
def next_states(state):
```

```
    moves = []
```

```
    potential_state = (not state[man], state[lion], state[goat], state[grass])
```

```
    if check_constraints(potential_state):
```

```
        moves.append(potential_state)
```

```
if state[man] == state[lion]:

    potential_state = (not state[man], not state[lion], state[goat], state[grass])

    if check_constraints(potential_state):

        moves.append(potential_state)


if state[man] == state[goat]:

    potential_state = (not state[man], state[lion], not state[goat], state[grass])

    if check_constraints(potential_state):

        moves.append(potential_state)


if state[man] == state[grass]:

    potential_state = (not state[man], state[lion], state[goat], not state[grass])

    if check_constraints(potential_state):

        moves.append(potential_state)


return moves
```

```
def dfs(start, goal):

    stack = [[start]]

    visited = set([start])

    while stack:

        path = stack.pop()
```

```
state = path[-1]
```

```
if state == goal:
```

```
    return path
```

```
for next_state in next_states(state):
```

```
    if next_state not in visited:
```

```
        visited.add(next_state)
```

```
        next_path = path.copy()
```

```
        next_path.append(next_state)
```

```
        stack.append(next_path)
```

```
return None
```

```
solution = dfs(start, goal)
```

```
if solution:
```

```
    for state in solution:
```

```
        print_state(state)
```

```
else:
```

```
    print("No solution!")
```