```python
from collections import deque

x = int(input("Enter capacity of Jug 1: "))
y = int(input("Enter capacity of Jug 2: "))
target = int(input("Enter Target to achieve: "))

def gcd(a, b):
    while b:
        a, b = b, a % b
    return a

if target > max(x, y):
    print("No solution possible: Target exceeds capacity of both jugs")
elif target % gcd(x, y) != 0:
    print("No solution possible: Target cannot be measured with these jug sizes")
else:

    visited = set()
    queue = deque([(0, 0, [])])
    solution_found = False

    while queue and not solution_found:
        jug1, jug2, steps = queue.popleft()

        if jug1 == target or jug2 == target:
            print("Solution found:")
            for i, step in enumerate(steps, 1):
                print(f"{i}. {step}")
            solution_found = True

        if (jug1, jug2) in visited:
            continue

        visited.add((jug1, jug2))

        if jug1 < x:
            queue.append((x, jug2, steps + [f"Fill jug 1: ({x}, {jug2})"]))

        if jug2 < y:
            queue.append((jug1, y, steps + [f"Fill jug 2: ({jug1}, {y})"]))

        if jug1 > 0:
            queue.append((0, jug2, steps + [f"Empty jug 1: (0, {jug2})"]))

        if jug2 > 0:
            queue.append((jug1, 0, steps + [f"Empty jug 2: ({jug1}, 0)"]))

        if jug1 > 0 and jug2 < y:
            pour = min(jug1, y - jug2)
            queue.append((jug1 - pour, jug2 + pour, steps + [f"Pour jug 1 to jug 2: ({jug1 - pour}, {jug2 + pour})"]))
```

```python
            if jug2 > 0 and jug1 < x:
                pour = min(jug2, x - jug1)
                queue.append((jug1 + pour, jug2 - pour, steps + [f"Pour jug 2 to jug 1:
    ({jug1 + pour}, {jug2 - pour})"]))

    if not solution_found:
        print("No solution found")
```

```
pwsh  Python_LocalVC  master ≡ ?1  56s 844ms
>> python -u "d:\SelfRepoClone\Python_LocalVC\AI Lab\Water_Jug_Problem.py"
Enter capacity of Jug 1: 5
Enter capacity of Jug 2: 3
Enter Target to achieve: 4
Solution found:
1. Fill jug 1: (5, 0)
2. Pour jug 1 to jug 2: (2, 3)
3. Empty jug 2: (2, 0)
4. Pour jug 1 to jug 2: (0, 2)
5. Fill jug 1: (5, 2)
6. Pour jug 1 to jug 2: (4, 3)
```