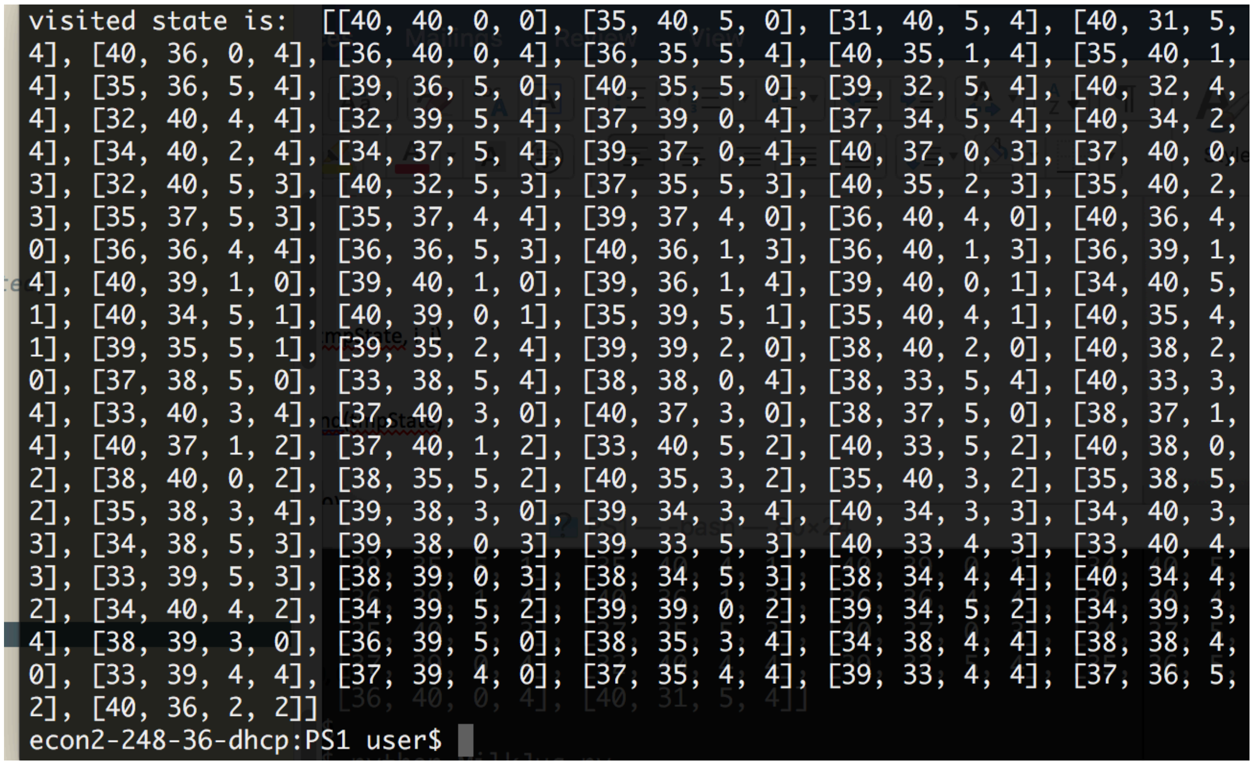
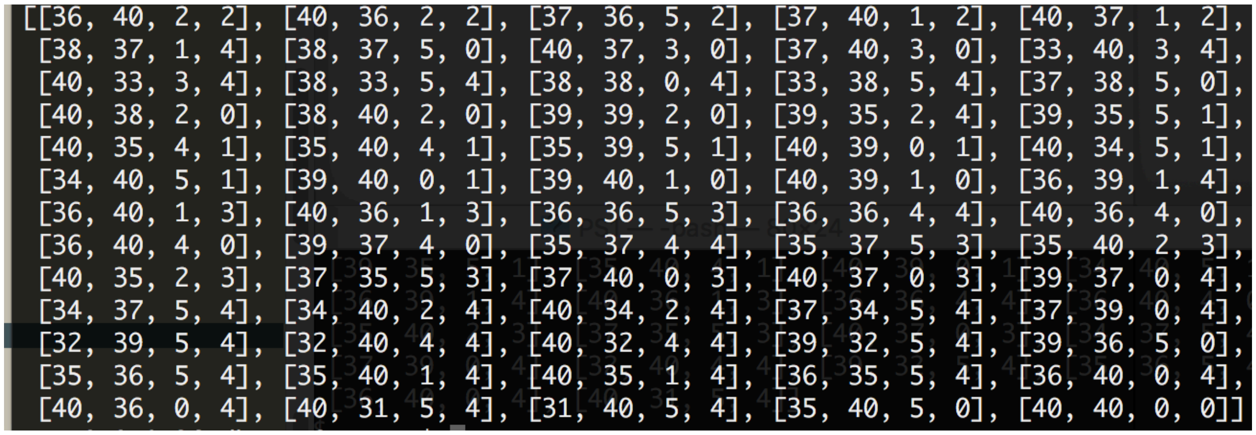
The total size of the problem space is 105 states



```
visited state is:  [[40, 40, 0, 0], [35, 40, 5, 0], [31, 40, 5, 4], [40, 31, 5,
4], [40, 36, 0, 4], [36, 40, 0, 4], [36, 35, 5, 4], [40, 35, 1, 4], [35, 40, 1,
4], [35, 36, 5, 4], [39, 36, 5, 0], [40, 35, 5, 0], [39, 32, 5, 4], [40, 32, 4,
4], [32, 40, 4, 4], [32, 39, 5, 4], [37, 39, 0, 4], [37, 34, 5, 4], [40, 34, 2,
4], [34, 40, 2, 4], [34, 37, 5, 4], [39, 37, 0, 4], [40, 37, 0, 3], [37, 40, 0,
3], [32, 40, 5, 3], [40, 32, 5, 3], [37, 35, 5, 3], [40, 35, 2, 3], [35, 40, 2,
3], [35, 37, 5, 3], [35, 37, 4, 4], [39, 37, 4, 0], [36, 40, 4, 0], [40, 36, 4,
0], [36, 36, 4, 4], [36, 36, 5, 3], [40, 36, 1, 3], [36, 40, 1, 3], [36, 39, 1,
4], [40, 39, 1, 0], [39, 40, 1, 0], [39, 36, 1, 4], [39, 40, 0, 1], [34, 40, 5,
1], [40, 34, 5, 1], [40, 39, 0, 1], [35, 39, 5, 1], [35, 40, 4, 1], [40, 35, 4,
1], [39, 35, 5, 1], [39, 35, 2, 4], [39, 39, 2, 0], [38, 40, 2, 0], [40, 38, 2,
0], [37, 38, 5, 0], [33, 38, 5, 4], [38, 38, 0, 4], [38, 33, 5, 4], [40, 33, 3,
4], [33, 40, 3, 4], [37, 40, 3, 0], [40, 37, 3, 0], [38, 37, 5, 0], [38, 37, 1,
4], [40, 37, 1, 2], [37, 40, 1, 2], [33, 40, 5, 2], [40, 33, 5, 2], [40, 38, 0,
2], [38, 40, 0, 2], [38, 35, 5, 2], [40, 35, 3, 2], [35, 40, 3, 2], [35, 38, 5,
2], [35, 38, 3, 4], [39, 38, 3, 0], [39, 34, 3, 4], [40, 34, 3, 3], [34, 40, 3,
3], [34, 38, 5, 3], [39, 38, 0, 3], [39, 33, 5, 3], [40, 33, 4, 3], [33, 40, 4,
3], [33, 39, 5, 3], [38, 39, 0, 3], [38, 34, 5, 3], [38, 34, 4, 4], [40, 34, 4,
2], [34, 40, 4, 2], [34, 39, 5, 2], [39, 39, 0, 2], [39, 34, 5, 2], [34, 39, 3,
4], [38, 39, 3, 0], [36, 39, 5, 0], [38, 35, 3, 4], [34, 38, 4, 4], [38, 38, 4,
0], [33, 39, 4, 4], [37, 39, 4, 0], [37, 35, 4, 4], [39, 33, 4, 4], [37, 36, 5,
2], [40, 36, 2, 2]]
econ2-248-36-dhcp:PS1 user$
```

The correct solution of total state is 65 states.



```
[[36, 40, 2, 2], [40, 36, 2, 2], [37, 36, 5, 2], [37, 40, 1, 2], [40, 37, 1, 2],
 [38, 37, 1, 4], [38, 37, 5, 0], [40, 37, 3, 0], [37, 40, 3, 0], [33, 40, 3, 4],
 [40, 33, 3, 4], [38, 33, 5, 4], [38, 38, 0, 4], [33, 38, 5, 4], [37, 38, 5, 0],
 [40, 38, 2, 0], [38, 40, 2, 0], [39, 39, 2, 0], [39, 35, 2, 4], [39, 35, 5, 1],
 [40, 35, 4, 1], [35, 40, 4, 1], [35, 39, 5, 1], [40, 39, 0, 1], [40, 34, 5, 1],
 [34, 40, 5, 1], [39, 40, 0, 1], [39, 40, 1, 0], [40, 39, 1, 0], [36, 39, 1, 4],
 [36, 40, 1, 3], [40, 36, 1, 3], [36, 36, 5, 3], [36, 36, 4, 4], [40, 36, 4, 0],
 [36, 40, 4, 0], [39, 37, 4, 0], [35, 37, 4, 4], [35, 37, 5, 3], [35, 40, 2, 3],
 [40, 35, 2, 3], [37, 35, 5, 3], [37, 40, 0, 3], [40, 37, 0, 3], [39, 37, 0, 4],
 [34, 37, 5, 4], [34, 40, 2, 4], [40, 34, 2, 4], [37, 34, 5, 4], [37, 39, 0, 4],
 [32, 39, 5, 4], [32, 40, 4, 4], [40, 32, 4, 4], [39, 32, 5, 4], [39, 36, 5, 0],
 [35, 36, 5, 4], [35, 40, 1, 4], [40, 35, 1, 4], [36, 35, 5, 4], [36, 40, 0, 4],
 [40, 36, 0, 4], [40, 31, 5, 4], [31, 40, 5, 4], [35, 40, 5, 0], [40, 40, 0, 0]]
```

#!/bin/bash python

# 4 milk cans capacity -> (x,y,z,w) where (x = y > z > w)
# initial_state = (40,40,0,0)
# final state = (40,36,2,2) or (36,40,2,2)

```
# mark visited state
visited = []
# final solution
solution = []

def getState(state, From, toThere):
   global count

   a = state[0]
   b = state[1]
   c = state[2]
   d = state[3]

   tmpState = list(state)
   #print tmpState

   if toThere == 0:
      Max = Capacity[0]
   elif toThere == 1:
      Max = Capacity[1]
   elif toThere == 2:
      Max = Capacity[2]
   elif toThere == 3:
      Max = Capacity[3]

   # Check how much you can pour to
   pour_amount = Max - state[toThere]

   if From != toThere:
      if tmpState[From] <= pour_amount:
         tmpState[toThere] += tmpState[From]
         tmpState[From] = 0
      else:
         tmpState[From] -= pour_amount
         tmpState[toThere] += pour_amount

   if tmpState in visited:
      #print("State had been visited: " , visited)
      return False
   elif state[2] == 2 and state[3] == 2:
      solution.append(tmpState)
      return True
   else:
      visited.append(tmpState)
```

```
        print visited

    for i in range(0, 4):
        for j in range(0, 4):
            goal = getState(tmpState, i, j)

            if goal == True:
                solution.append(tmpState)
                return True

initail_state = (40,40,0,0)
Capacity = (40,40,5,4)  #a, b, c ,d

numberOfState = 1
print ("Start...\n")

getState(initail_state, 0, 0)
#print solution
#print solution.reverse()
```