Job No: 08

Job Name: **Write a program for Water Jug Problem.**

Theory: The Water Jug Problem is a puzzle where we have two jugs, one with a capacity of 4L and the other with a capacity of 3L. Our objective is to use the 3L jug to measure and transfer exactly 2L of water into the 4L jug.

Code:

```
a = int(input("Enter Jug A Capacity: "))
b = int(input("Enter Jug B Capacity: "))
ai = int(input("Initially Water in Jug A: "))
bi = int(input("Initially Water in Jug B: "))
af = int(input("Final State of Jug A: "))
bf = int(input("Final State of Jug B: "))
count = 0
print("Operations:\n 1.Fill A\n 2.Fill B\n 3.Empty A\n 4.Empty B\n 5.Pour A=>B\n 6.Pour B=>A")
while ai != af or bi != bf:
    op = int(input("Enter the Operation: "))
    count += 1
    if op == 1:
        ai = a
    elif op == 2:
        bi = b
    elif op == 3:
        ai = 0
    elif op == 4:
        bi = 0
    elif op == 5:
        ai, bi = ai - min(ai, b - bi), bi + min(ai, b - bi)
    elif op == 6:
        bi, ai = bi - min(bi, a - ai), ai + min(bi, a - ai)
    else:
        print("Invalid operation. Please enter a valid operation number.")

    print(f"Jug A: {ai}L, Jug B: {bi}L")
```

```
print(f"You have tried {count} times to reach the goal")
print("Desired state reached!")
```

Input/Output:

```
Enter Jug A Capacity: 4
Enter Jug B Capacity: 3
Initially Water in Jug A: 0
Initially Water in Jug B: 0
Final State of Jug A: 2
Final State of Jug B: 0
Operations:
 1.Fill A
 2.Fill B
 3.Empty A
 4.Empty B
 5.Pour A=>B
 6.Pour B=>A
Enter the Operation: 1
Jug A: 4L, Jug B: 0L
Enter the Operation: 5
Jug A: 1L, Jug B: 3L
Enter the Operation: 4
Jug A: 1L, Jug B: 0L
Enter the Operation: 5
Jug A: 0L, Jug B: 1L
Enter the Operation: 1
Jug A: 4L, Jug B: 1L
Enter the Operation: 5
Jug A: 2L, Jug B: 3L
Enter the Operation: 4
Jug A: 2L, Jug B: 0L
You have tried 7 times to reach the goal
Desired state reached!
```

Job No: 09

Job Name: **Write a program for Tower Of Hanoi Of Hanoi Problem.**

Theory: The Tower of Hanoi problem is a classic puzzle where you have three pegs and a stack of disks on one peg. The goal is to move the entire stack to another peg, following rules like moving one disk at a time and never placing a larger disk on top of a smaller one.

Code:

```
def tower_of_hanoi(n, source, auxiliary, destination):
    if n == 1:
        print(f"Move disk 1 from {source} to {destination}")
        return
    tower_of_hanoi(n-1, source, destination, auxiliary)
    print(f"Move disk {n} from {source} to {destination}")
    tower_of_hanoi(n-1, auxiliary, source, destination)

num_disks = 3
tower_of_hanoi(num_disks, 'A', 'B', 'C')
```

Input/Output:

```
Move disk 1 from A to C
Move disk 2 from A to B
Move disk 1 from C to B
Move disk 3 from A to C
Move disk 1 from B to A
Move disk 2 from B to C
Move disk 1 from A to C
```