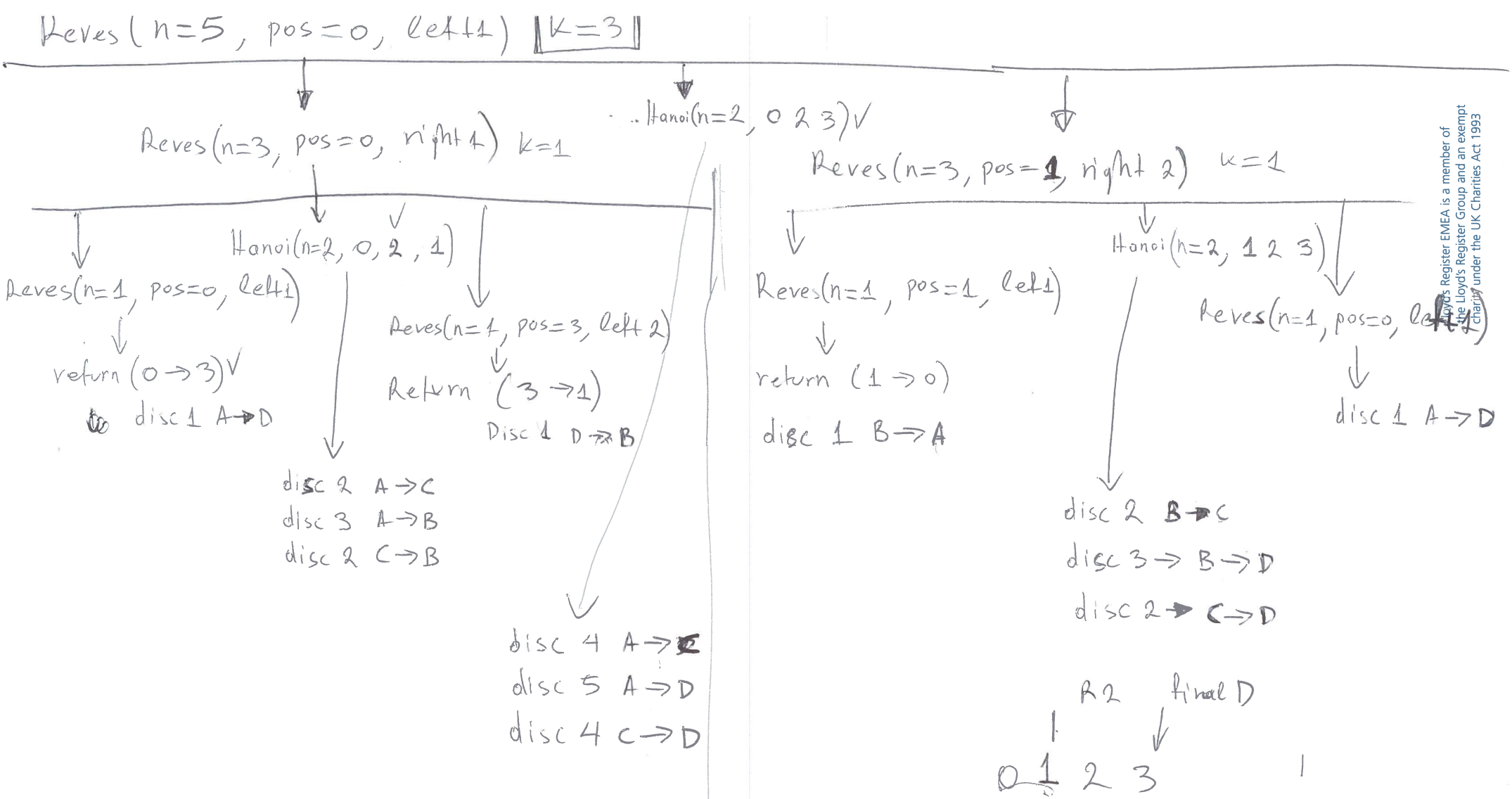
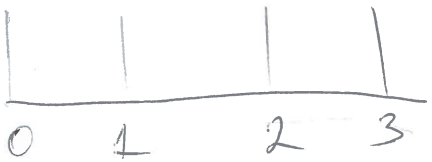


Algo: Right → Left → Right, where { Right is R1, R2 ; Left is L1, L2 } →

R1 → L1 L2
R2 → L1 L1
L1 ⇒ R1 R2
L2 → R1 R1



Starting call: (n, position=0, Right/Left)

If n==1: Reduces to Hanoi ✓

If n==2:

Reves(2, pos=0, Left1)
Reves(1, pos=0, Right1) → return(0→1)
Hanoi(n=1, 0 2 3) → ✓
Reves(1, pos=1, Right2) → return(1→3) ✓

(n=2, pos=0, Right)
↓ Reves(1, pos=0, Left1) → return(0→3) ✓
Hanoi1(0 2 1) ✓
Reves(1, pos=3, Left2) → return(3→1) ✓

Reves(n=3, pos=0, Left1)
K=1
→ Reves(1, pos=0, Right1) → returns(0→1) ✓
→ Hanoi(n=2, 0 2 3) ✓
→ Reves(1, pos=1, Right2) → returns(1→3) ✓

Reves(n=4, pos=0, Left) K=2
→ Reves(2, pos=0, Right1)
  Reves(1, pos=0, Left1) returns(0→3)
  Hanoi(0 2 1) ✓
  Reves(1, pos=3, Left2) returns(3→1)
→ Hanoi(n=2, 0 2 3) ✓
→ Reves(2, pos=1, Right2) ✓
  Reves(1, pos=1, Left1) returns(1→0)
  Hanoi(1 2 3) ✓
  Reves(1, pos=0, Left1) returns(0→3) ✓