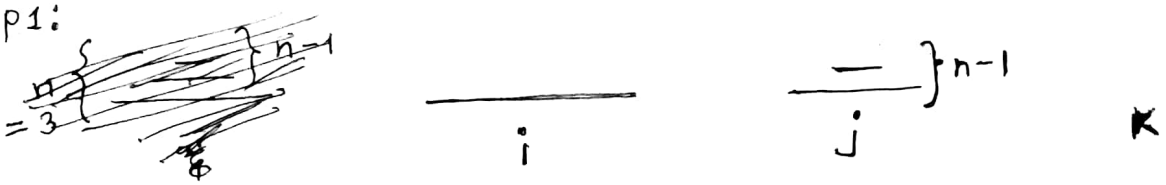
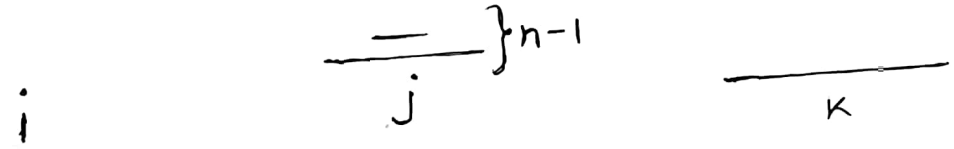
Tower of Hanoi:

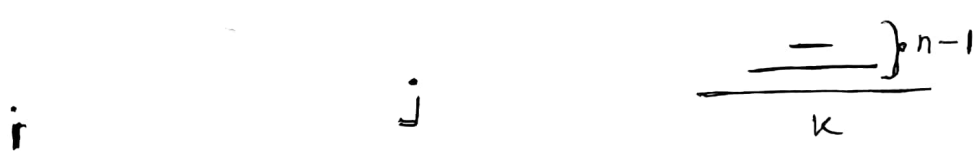
step1:



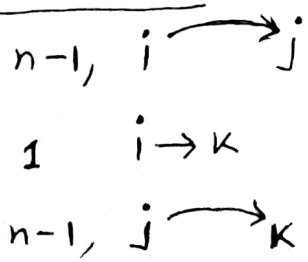
step2:



step3:



When n>1



When n=1


```
void Hanoi(int n, int i, int j
                         int k){

  if(n==1)
    printf("%d→%d", i, k);
  else {
    Hanoi(n-1, i, k, j);
    printf("%d→%d", i, k);
    Hanoi(n-1, j, i, k);
  }

}
```

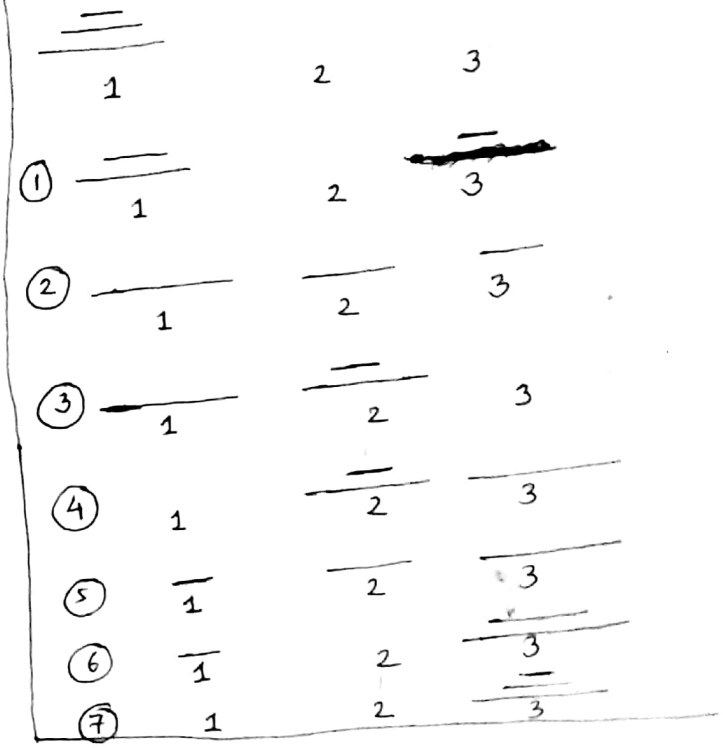
Find output sequence for Tower of Hanoi algorithm when n = 3.

- Tower of Hanoi algorithm

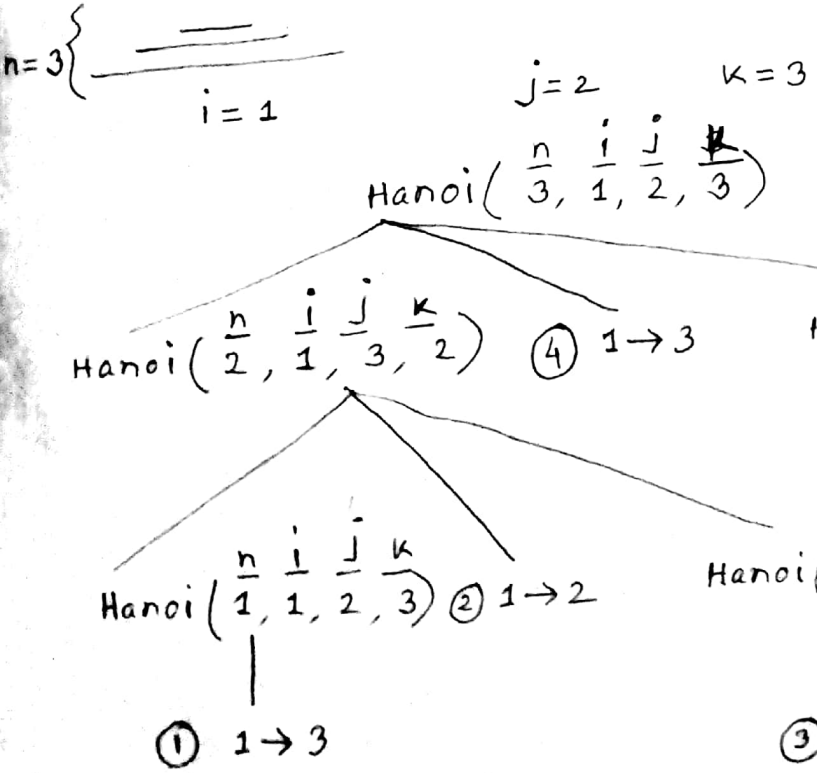
```
void Hanoi(int n, int i, int j, int k)
{
  if (n == 1)
    printf("%d → %d", i, k);
  else {
    Hanoi(n-1, i, k, j);
    printf("%d → %d", i, k);
    Hanoi(n-1, j, i, k);
  }
}
```

output sequence

① 1 → 3	⑤ 2 → 1
② 1 → 2	⑥ 2 → 3
③ 3 → 2	⑦ 1 → 3
④ 1 → 3	



Recursion Tree for n = 3



① Convert the following infix expression into postfix using stack

$a+b-(c*d/e-f)\uparrow c$

Ans:

Infix expression	stack	Postfix expression
a+b-(c*d/e-f)↑c		
+b-(c*d/e-f)↑c		a
b-(c*d/e-f)↑c	+	a
-(c*d/e-f)↑c	+	ab
(c*d/e-f)↑c	-	ab+
c*d/e-f)↑c	-(	ab+
*d/e-f)↑c	-(	ab+c
d/e-f)↑c	-(*	ab+c
/e-f)↑c	-(*	ab+cd
e-f)↑c	-(/	ab+cd*
-f)↑c	-(/	ab+cd*e
f)↑c	-(-	ab+cd*e/
)↑c	-(-	ab+cd*e/f
↑c	-	ab+cd*e/f-
c	-↑	ab+cd*e/f-
	-↑	ab+cd*e/f-c
		ab+cd*e/f-c↑-

top(stack) >= insert
⇒ + >= - (True)

top(stack) >= insert
⇒ * >= / (True)

top(stack) >= insert
⇒ / >= - True

top(stack) >= insert
⇒ - >= ↑ False

Postfix

Operator	Priority
+	1
-	1
*	2
/	2
↑	3

Evaluate the Postfix expression when a = 1, b = 2, c = 1
d = 2, e = 2, f = 1,

Postfix expression	stack	operation
ab+cd*e/f-c↑-		
b+cd*e/f-c↑-	a	
+cd*e/f-c↑-	~~a~~ ~~b~~	
cd*e/f-c↑-	3	a + b = 1+2 = 3
d*e/f-c↑-	3 c	
*e/f-c↑-	3 c d	
e/f-c↑-	3 2	c * d = 1*2 = 2
/f-c↑-	3 ~~2~~ ~~e~~	
f-c↑-	3 1	2/e = 2/2 = 1
-c↑-	3 ~~1~~ ~~f~~	1 - f = 1 - 1 = 0
c↑-	3 0	
↑-	3 0 c	
-	~~3~~ ~~0~~	0 ↑ c = 0^1 = 0
	3	3 - 0 = 3

Result = 3

Verification

a + b - (c*d/e - f) ↑ c
= 1 + 2 - (1*2/2 - 1) ↑ 1
= 3 - (1 - 1) ↑ 1
= 3 - 0 ↑ 1
= $3 - 0^1$
= 3 - 0
= 3 ⇒ Result