

#02-2. 재귀 함수의 활용


나정휘

<https://justicehui.github.io/>

동영상 강의

동일한 내용을 설명하는 영상이 있음

- 재귀 함수의 활용 (<https://youtu.be/8cSjBQtqEXY>)
- 재귀 함수 말고도 다양한 영상이 있으니 많은 관심 부탁...
- 한국정보올림피아드위원회 공식 유튜브 채널임



IOI KOREA
@ioikorea5159 구독자 1.71천명 동영상 255개
한국 정보올림피아드 위원회 공식 유튜브 채널입니다. >

홈

동영상

재생목록

커뮤니티

채널


정보

🔍

구독중

생성된 재생목록


정렬 기준



동영상 4개

알고리즘 강의 - 고급


모든 재생목록 보기



동영상 9개

알고리즘 강의 - 중급


모든 재생목록 보기



동영상 28개

알고리즘 강의 - 초급


모든 재생목록 보기



동영상 23개

문제 풀이 - 기타

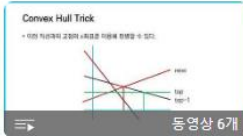
모든 재생목록 보기



동영상 6개

문제 풀이 - 기타 II


모든 재생목록 보기



동영상 6개

문제 풀이 - APIO


모든 재생목록 보기



동영상 22개

문제 풀이 - IOI

모든 재생목록 보기



동영상 57개

문제 풀이 - KOI

모든 재생목록 보기

재귀 함수

재귀 함수: 자기 자신을 호출하는 함수

- 예시: $f(n) = f(n-1) + f(n-2)$
- 용도: 점화식 계산, 완전 탐색, 반복문으로 반복하기 힘든 작업, ...

다양한 알고리즘에 사용되는 개념

- 동적 계획법, 분할 정복은 재귀적/귀납적 사고를 기반으로 함
- 트리 순회, 세그먼트 트리 등 트리 구조와 관련된 알고리즘은 주로 재귀 함수를 이용

재귀 함수의 예시 - 1

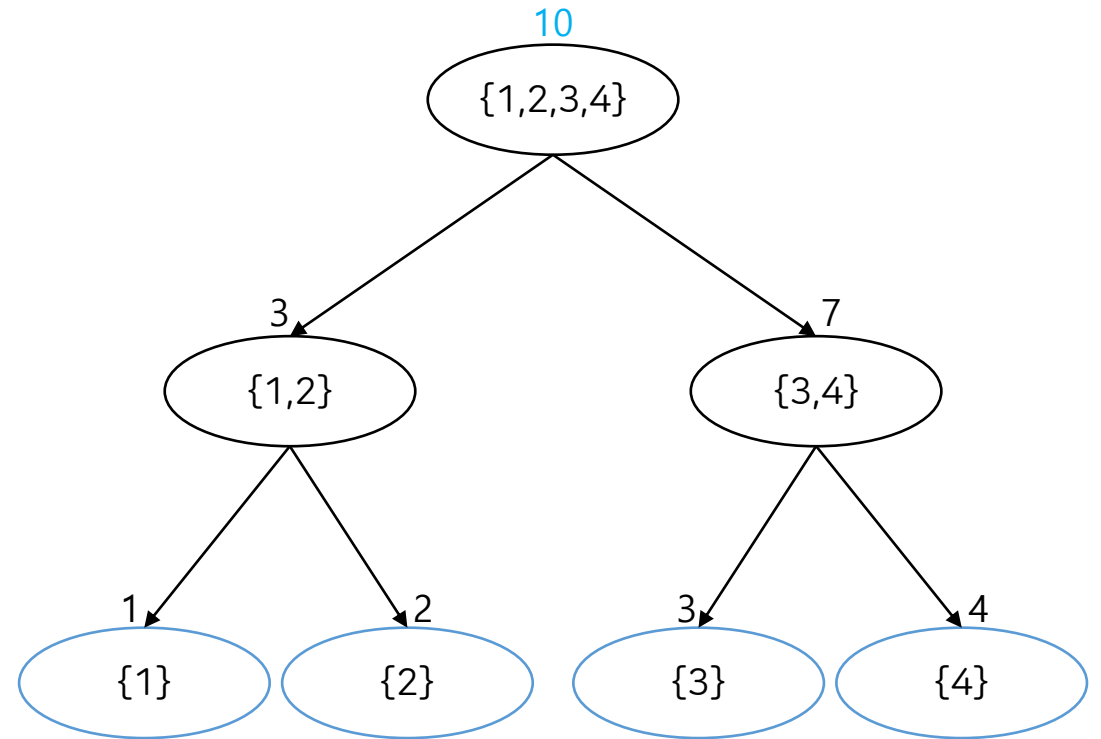
배열 A의 모든 원소의 합을 구하는 작업

- $f(l, r)$: $A[l] + A[l+1] + \dots + A[r]$ 을 구하는 함수
- $f(0, N-1)$ 을 구해야 함
- $f(l, r)$ 을 계산하는 방법
 - 일을 혼자 하는 것은 어려우니까 부하 직원 2명 고용
 - 배열을 절반으로 나눠서 직원 2명에게 분배
- 수식으로 표현
 - 배열의 중간 지점을 $m = (l + r) / 2$ 라고 하면
 - $f(l, r) = f(l, m) + f(m+1, r)$
 - $l = r$ 이면 $f(l, r) = A[l]$

재귀 함수의 예시 - 1

배열 A의 모든 원소의 합을 구하는 작업

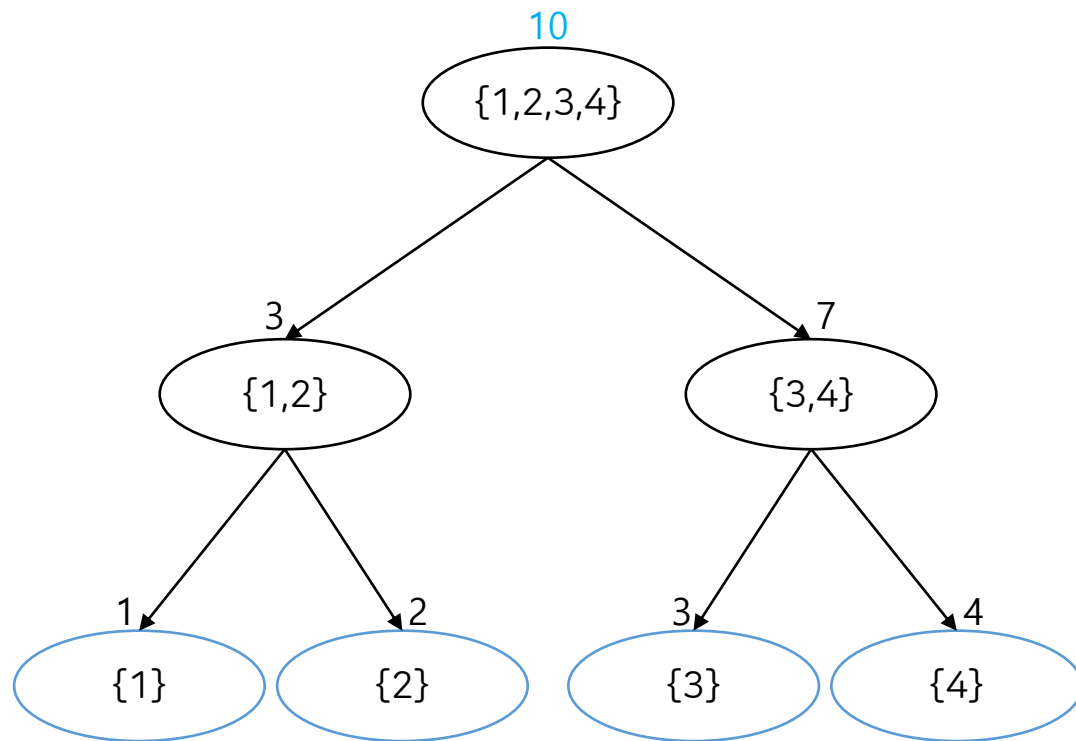
- $f(l, r)$: $A[l] + A[l+1] + \dots + A[r]$ 을 구하는 함수
- $f(l, r) = f(l, m) + f(m+1, r)$
- $l = r$ 이면 $f(l, r) = A[l]$
- 재귀 호출 과정을 살펴보면...



재귀 함수의 예시 - 1

배열 A의 모든 원소의 합을 구하는 작업

- $f(l, r)$: $A[l] + A[l+1] + \dots + A[r]$ 을 구하는 함수
- $f(l, r) = f(l, m) + f(m+1, r)$
- $l = r$ 이면 $f(l, r) = A[l]$
- 재귀 호출 과정을 살펴보면...
 - 배열의 길이가 1000000이라면?
 - 손으로 직접 호출 과정을 따라가기 어려움
 - 좋은 방법이 없을까?



재귀 함수의 예시 - 1

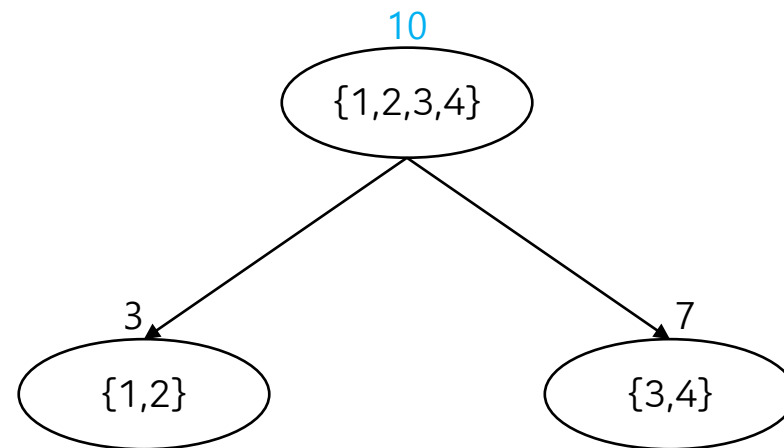
배열 A의 모든 원소의 합을 구하는 작업

- $f(l, r): A[l] + A[l+1] + \dots + A[r]$ 을 구하는 함수
- $f(l, r)$ 을 계산하는 방법
 - 일을 혼자 하는 것은 어려우니까 부하 직원 2명 고용
 - 배열을 절반으로 나눠서 직원 2명에게 분배
- 재귀 호출 과정을 파악하는 방법
 - 부하 직원이 항상 올바른 결과를 반환한다고 “믿고”
 - 내 할 일만 잘 하면 됨
 - 수학적 귀납법

재귀 함수의 예시 - 1

배열 A의 모든 원소의 합을 구하는 작업

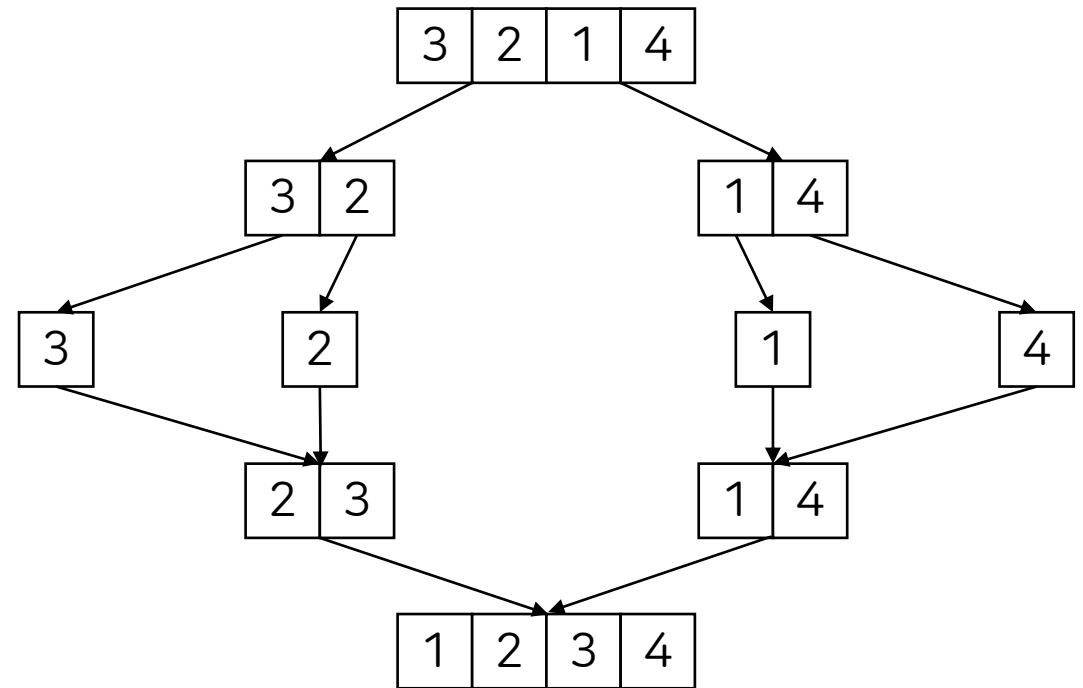
- $f(l, r)$: $A[l] + A[l+1] + \dots + A[r]$ 을 구하는 함수
- $f(l, r) = f(l, m) + f(m+1, r)$
- $l = r$ 이면 $f(l, r) = A[l]$
- 재귀 호출 과정을 살펴보면...
 - 함수 f가 올바른 결과를 반환한다고 "믿고"
 - $f(l, m) + f(m+1, r)$ 의 합만 잘 계산하면 됨



재귀 함수의 예시 - 2

배열 A를 정렬하는 작업 (합병 정렬 알고리즘)

- $f(l, r)$: $A[l..r]$ 을 정렬된 상태로 만드는 함수
- $l = r$ 이면 이미 정렬된 상태 (종료 조건)
- 재귀 함수를 이용한 정렬 방법
 - $f(l, m)$ 과 $f(m+1, r)$ 을 호출한 다음
 - $A[l..m]$ 과 $A[m+1..r]$ 이 정렬되어 있다고 "믿고"
 - 정렬된 두 배열을 잘 합치면 됨
- 자세한 방법은 분할 정복 시간에 다룸
 - 예습하고 싶으면...
 - 분할 정복의 기초 (<https://youtu.be/BqqjWGPXNaQ>)
 - 분할 정복의 응용 (<https://youtu.be/MKklbMPggGY>)



재귀 함수

재귀 호출

- 자신과 동일한 일을 하는 부하 직원에게 작업을 요청하고 그 결과물을 돌려받는 과정
- 부하 직원의 결과물이 항상 올바르다고 믿고 작업을 수행하면 됨
- = 수학적 귀납법

재귀 함수

주의사항

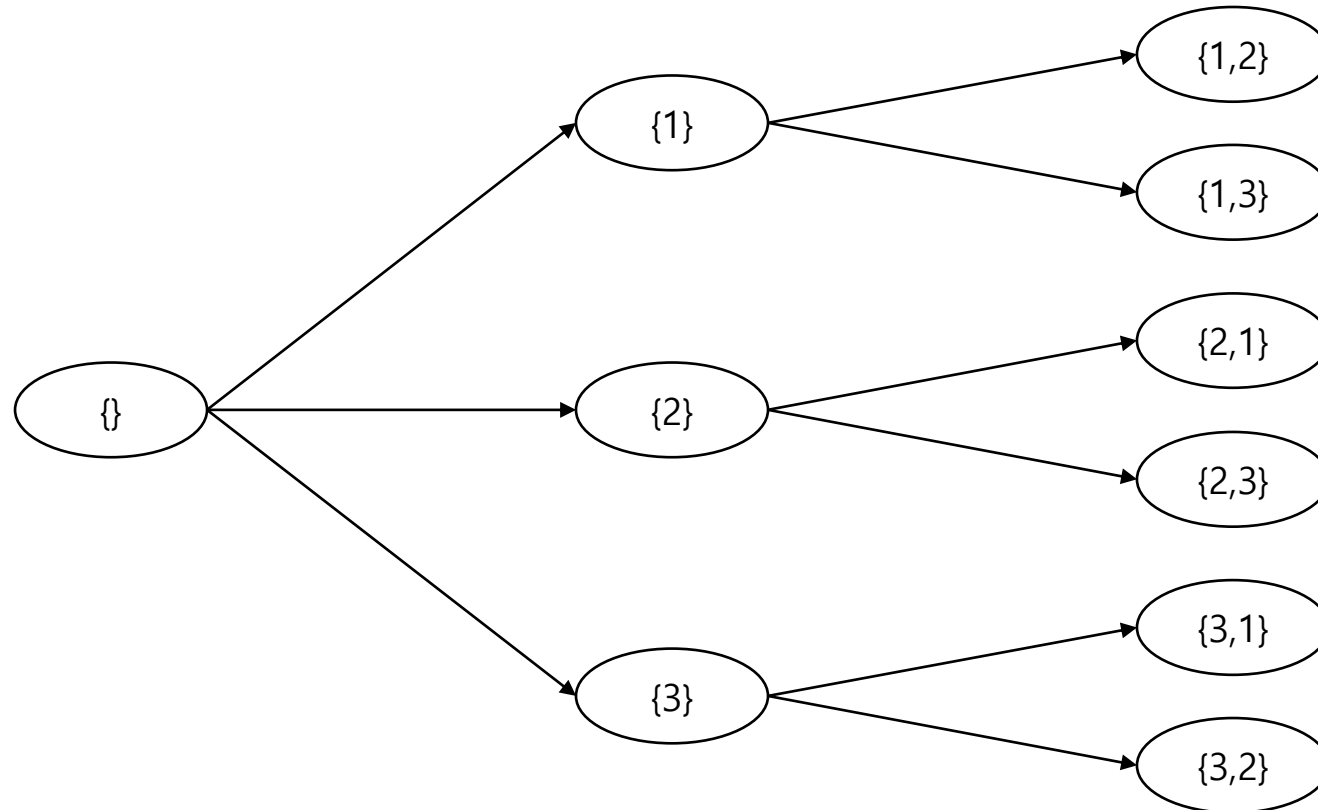
- 재귀 호출을 하지 않는 조건(종료 조건)이 있어야 함
 - 없으면 프로그램이 종료되지 않음
- 재귀 호출 과정에 사이클이 없어야 함
 - $f(a) \rightarrow f(b) \rightarrow f(c) \rightarrow \dots \rightarrow f(a)$
 - 프로그램이 종료되지 않음
- 종료 조건에 가까워지는 방향으로 설계
 - 앞에서 본 예시는 항상 $l \leq r$ 을 만족하고 $l = r$ 이 종료 조건
 - $r - l$ 이 작아지는 방향으로 재귀 호출

질문?

재귀 함수의 예시 - 3

BOJ 15649. N과 M (1)

- 1부터 N까지의 자연수 중 중복 없이 M개를 고른 수열을 모두 출력하는 문제
- 손으로 모든 순열을 구할 때는 수형도를 이용할 수 있음
- ex. $N = 3, M = 2$



재귀 함수의 예시 - 3

BOJ 15649. N과 M (1)

- f(choose, arr)
 - f(지금까지 선택한 수의 개수, 지금까지 만든 배열)
 - choose = M이면 arr 출력하고 종료
 - arr에 없는 수를 하나 골라서 맨 뒤에 추가하고 재귀 호출
 - arr[choose] = num;
 - f(choose+1, arr)
 - arr[choose] = 0;

```
void f(int n, int m, int choose, int arr[]){
    if(choose == m){ // 종료 조건
        for(int i=0; i<m; i++){
            printf("%d ", arr[i]);
        }
        printf("\n");
        return;
    }

    for(int i=1; i<=n; i++){ // n개의 수를 한 번씩 시도
        int exist = 0; // 이미 만든 배열에 i가 있으면 exist = 1
        for(int j=0; j<choose; j++){
            if(arr[j] == i) exist = 1;
        }
        if(exist == 0){
            arr[choose] = i; // i 선택
            f(n, m, choose+1, arr); // 선택한 수의 개수 1 증가
            arr[choose] = 0; // i 넣은 거 원상 복구
        }
    }
}
```

재귀 함수의 예시 - 3

BOJ 15649. N과 M (1)

- exist 변수의 값을 매번 계산하는 것을 비효율적
- use[num] = num을 사용했으면 1, 안 했으면 0
 - arr[choose] = num;
 - use[num] = 1;
 - f(choose+1, arr, use);
 - arr[choose] = 0;
 - use[num] = 0;

```
void f(int n, int m, int choose, int arr[], int use[]){
    if(choose == m){ // 종료 조건
        for(int i=0; i<m; i++){
            printf("%d ", arr[i]);
        }
        printf("\n");
        return;
    }

    for(int i=1; i<=n; i++){ // n개의 수를 한 번씩 시도
        if(use[i] == 0){ // 아직 i를 사용하지 않았다면
            arr[choose] = i; // i 선택
            use[i] = 1;
            f(n, m, choose+1, arr, use); // 선택한 수의 개수 1 증가
            arr[choose] = 0; // i 넣은 거 원상 복구
            use[i] = 0;
        }
    }
}
```

재귀 함수의 예시 - 3

BOJ 15649. N과 M (1)

- 항상 5개의 인자를 모두 넘겨줘야 할까?
 - n, m은 변하지 않는 값
 - arr, use도 포인터를 넘겨주는 것이므로 변하지 않음
 - 항상 동일한 값은 전역 변수로 빼도 됨

```
int N, M, A[8], U[9]; // 전역 변수는 0으로 초기화

void f(int choose){
    if(choose == M){
        for(int i=0; i<M; i++) printf("%d ", A[i]);
        printf("\n");
        return;
    }

    for(int i=1; i<=N; i++){
        if(!U[i]){
            A[choose] = i; U[i] = 1;
            f(choose+1);
            A[choose] = 0; U[i] = 0;
        }
    }
}
```


질문?

재귀 함수의 예시 - 4

BOJ 15650. N과 M (2)

- 1부터 N까지의 자연수 중 중복 없이 M개를 오름차순으로 고른 수열을 모두 출력하는 문제
- 현재 배열의 맨 뒤에 있는 수보다 큰 수를 선택해야 함

```
int N, M, A[8];

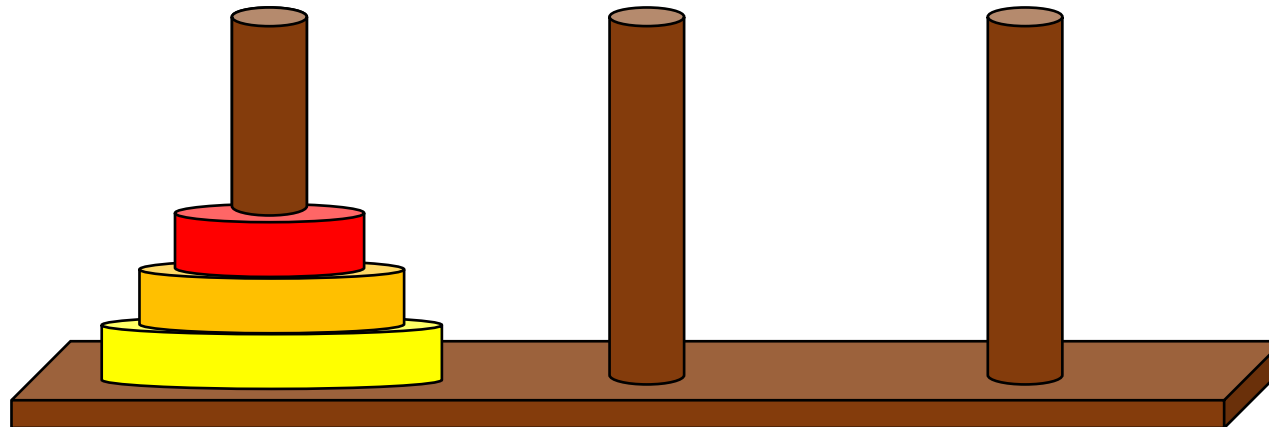
void f(int choose){
    if(choose == M){
        for(int i=0; i<M; i++) printf("%d ", A[i]);
        printf("\n");
        return;
    }

    for(int i=1; i<=N; i++){
        if(choose == 0 || A[choose-1] < i){
            A[choose] = i;
            f(choose+1);
        }
    }
}
```

재귀 함수의 예시 - 5

BOJ 11729. 하노이 탑 이동 순서

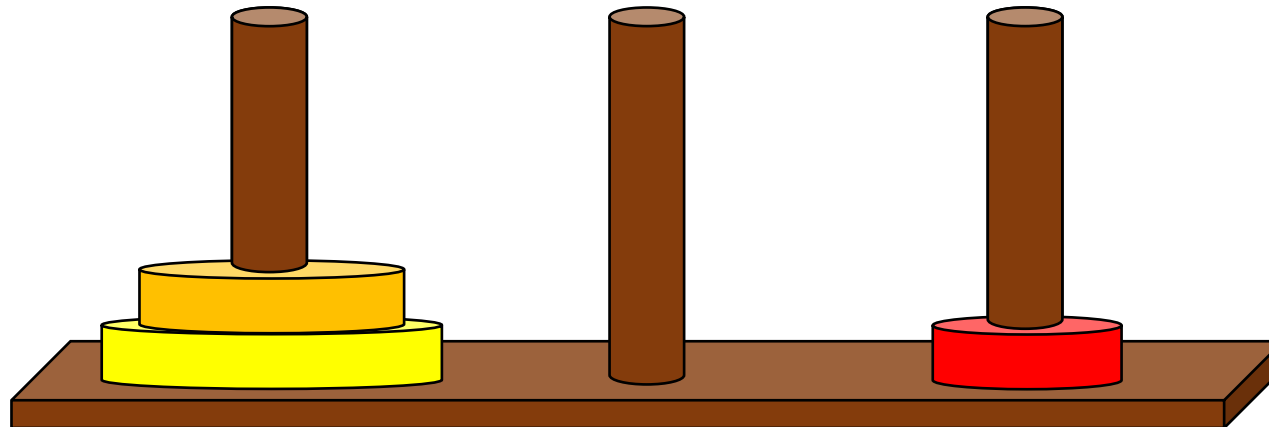
- 3개의 기둥과 크기가 서로 다른 N개의 원판이 있음
- 원판 위에는 자기 자신보다 작은 원판만 올라갈 수 있음
- 원판을 최소한으로 이동해서 첫 번째 기둥에 있는 모든 원판을 세 번째 기둥으로 옮기는 문제



재귀 함수의 예시 - 5

BOJ 11729. 하노이 탑 이동 순서

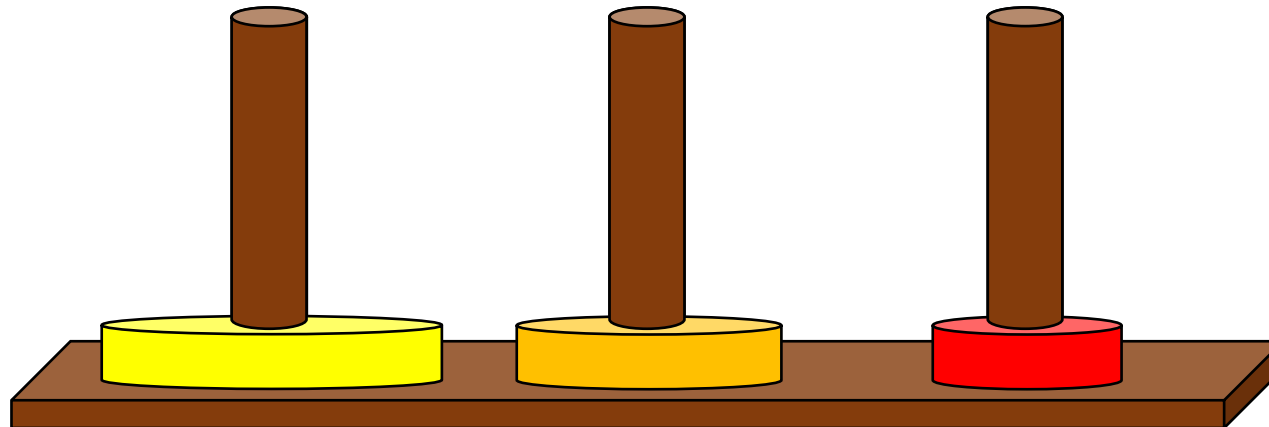
- 3개의 기둥과 크기가 서로 다른 N개의 원판이 있음
- 원판 위에는 자기 자신보다 작은 원판만 올라갈 수 있음
- 원판을 최소한으로 이동해서 첫 번째 기둥에 있는 모든 원판을 세 번째 기둥으로 옮기는 문제



재귀 함수의 예시 - 5

BOJ 11729. 하노이 탑 이동 순서

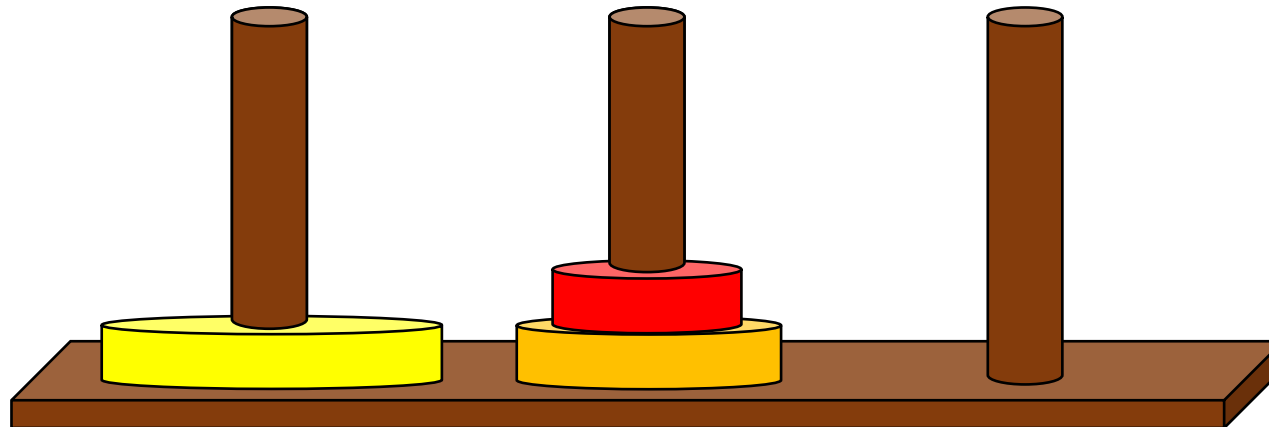
- 3개의 기둥과 크기가 서로 다른 N개의 원판이 있음
- 원판 위에는 자기 자신보다 작은 원판만 올라갈 수 있음
- 원판을 최소한으로 이동해서 첫 번째 기둥에 있는 모든 원판을 세 번째 기둥으로 옮기는 문제



재귀 함수의 예시 - 5

BOJ 11729. 하노이 탑 이동 순서

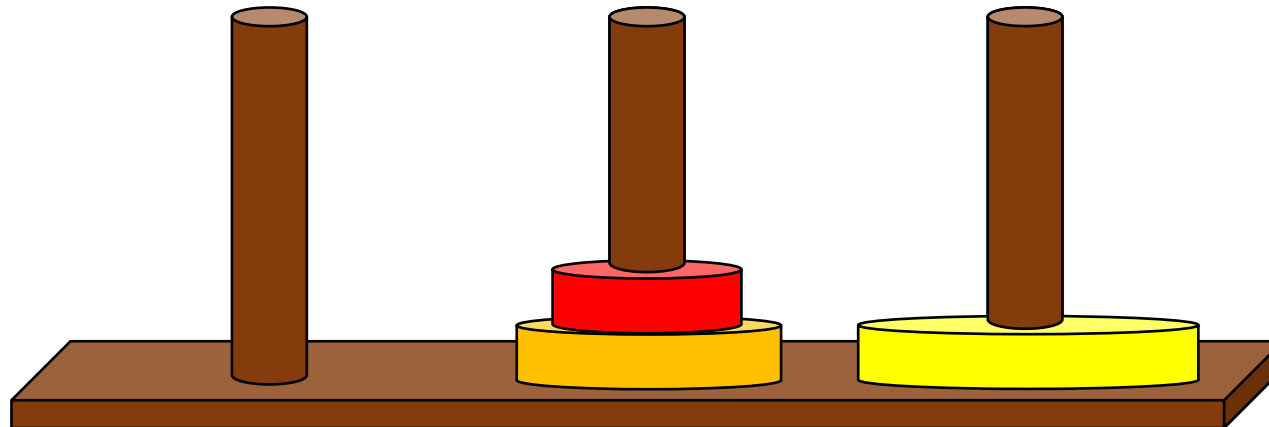
- 3개의 기둥과 크기가 서로 다른 N개의 원판이 있음
- 원판 위에는 자기 자신보다 작은 원판만 올라갈 수 있음
- 원판을 최소한으로 이동해서 첫 번째 기둥에 있는 모든 원판을 세 번째 기둥으로 옮기는 문제



재귀 함수의 예시 - 5

BOJ 11729. 하노이 탑 이동 순서

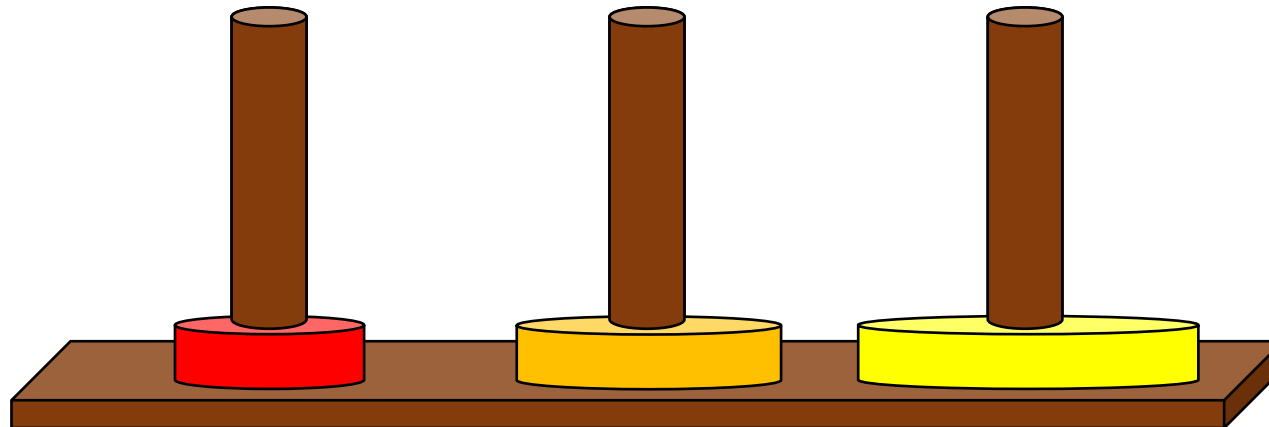
- 3개의 기둥과 크기가 서로 다른 N개의 원판이 있음
- 원판 위에는 자기 자신보다 작은 원판만 올라갈 수 있음
- 원판을 최소한으로 이동해서 첫 번째 기둥에 있는 모든 원판을 세 번째 기둥으로 옮기는 문제



재귀 함수의 예시 - 5

BOJ 11729. 하노이 탑 이동 순서

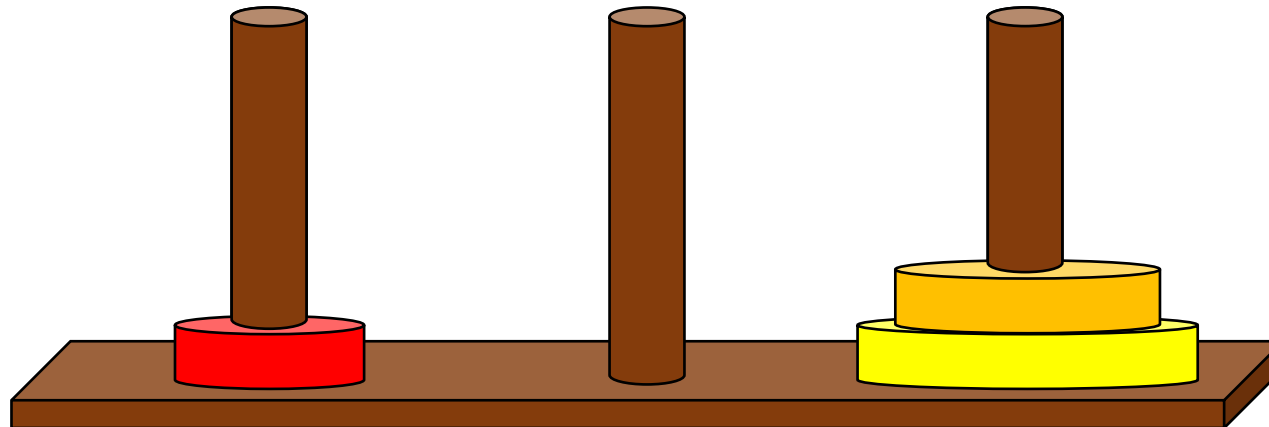
- 3개의 기둥과 크기가 서로 다른 N개의 원판이 있음
- 원판 위에는 자기 자신보다 작은 원판만 올라갈 수 있음
- 원판을 최소한으로 이동해서 첫 번째 기둥에 있는 모든 원판을 세 번째 기둥으로 옮기는 문제



재귀 함수의 예시 - 5

BOJ 11729. 하노이 탑 이동 순서

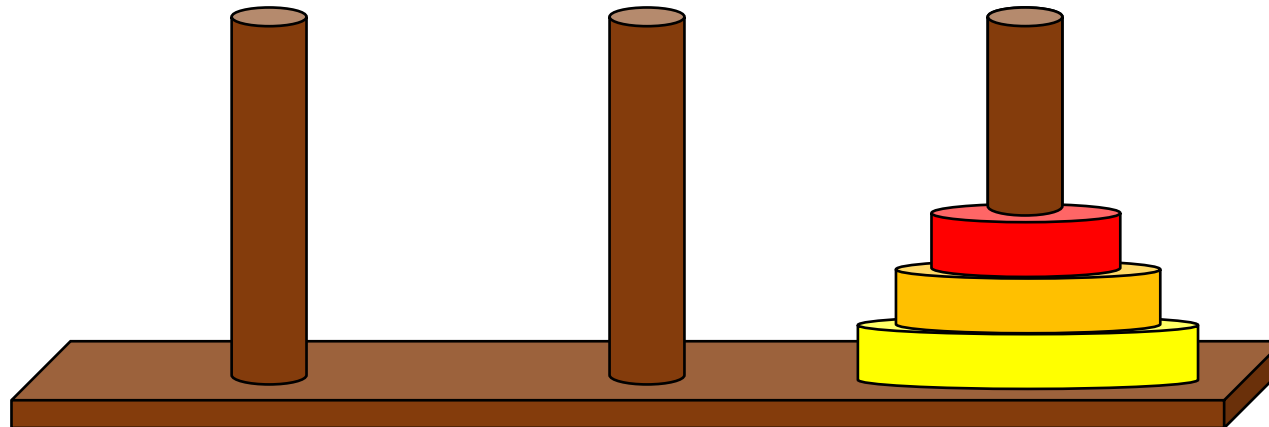
- 3개의 기둥과 크기가 서로 다른 N개의 원판이 있음
- 원판 위에는 자기 자신보다 작은 원판만 올라갈 수 있음
- 원판을 최소한으로 이동해서 첫 번째 기둥에 있는 모든 원판을 세 번째 기둥으로 옮기는 문제



재귀 함수의 예시 - 5

BOJ 11729. 하노이 탑 이동 순서

- 3개의 기둥과 크기가 서로 다른 N개의 원판이 있음
- 원판 위에는 자기 자신보다 작은 원판만 올라갈 수 있음
- 원판을 최소한으로 이동해서 첫 번째 기둥에 있는 모든 원판을 세 번째 기둥으로 옮기는 문제



재귀 함수의 예시 - 5

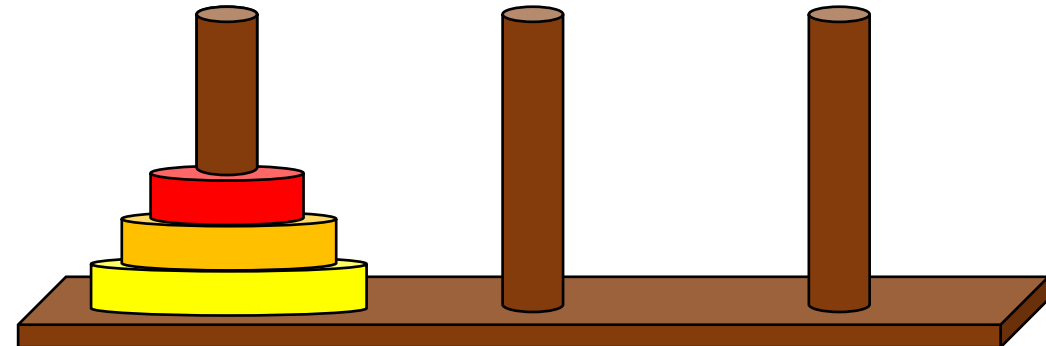
BOJ 11729. 하노이 탑 이동 순서

- 최적 전략은 어떤 형태일까?
- 첫 번째 기둥에 있는 N 개의 원판을 세 번째 기둥으로 옮겨야 함
- 가장 큰 원판부터 차례대로 세 번째 기둥으로 옮김
- 가장 큰 원판을 어떻게 꺼내지?
- 위에 있는 $N-1$ 개를 잠시 두 번째 기둥으로 옮기면 됨
- $N-1$ 개를 첫 번째 기둥에서 두 번째 기둥으로 옮기고
- N 번째 원판을 첫 번째 기둥에서 세 번째 기둥으로 옮기고
- $N-1$ 개를 두 번째 기둥에서 세 번째 기둥으로 옮기면 됨

재귀 함수의 예시 - 5

BOJ 11729. 하노이 탑 이동 순서

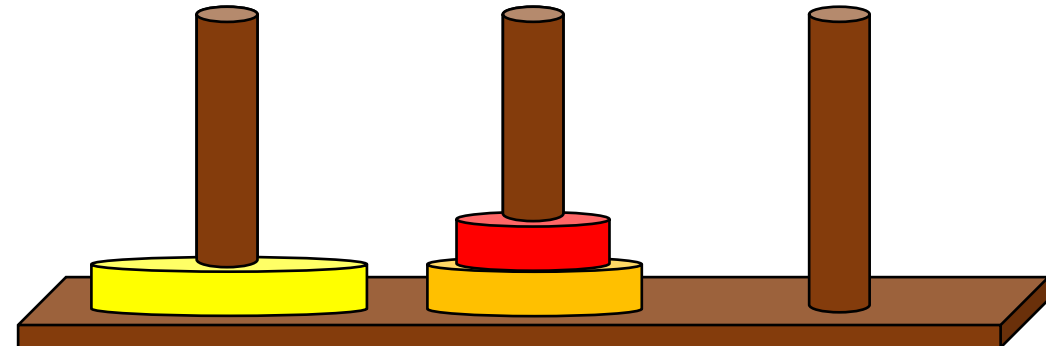
- 최적 전략은 어떤 형태일까?
- 첫 번째 기둥에 있는 N개의 원판을 세 번째 기둥으로 옮겨야 함
- 가장 큰 원판부터 차례대로 세 번째 기둥으로 옮김
- 가장 큰 원판을 어떻게 꺼내지?
- 위에 있는 N-1개를 잠시 두 번째 기둥으로 옮기면 됨
- N-1개를 첫 번째 기둥에서 두 번째 기둥으로 옮기고
- N번째 원판을 첫 번째 기둥에서 세 번째 기둥으로 옮기고
- N-1개를 두 번째 기둥에서 세 번째 기둥으로 옮기면 됨



재귀 함수의 예시 - 5

BOJ 11729. 하노이 탑 이동 순서

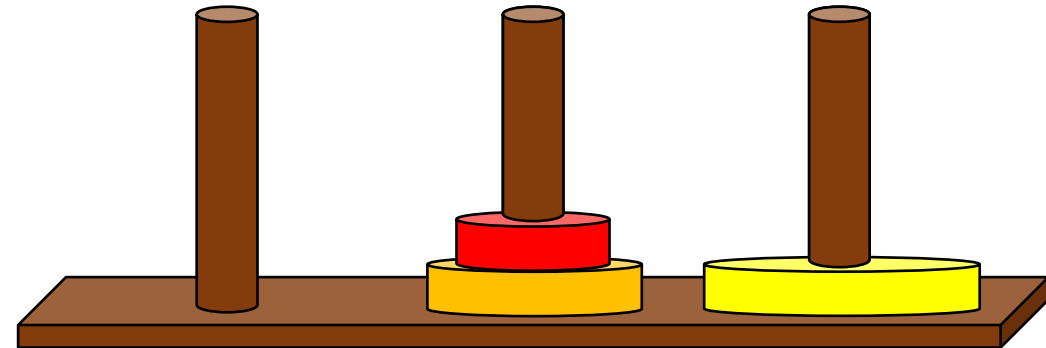
- 최적 전략은 어떤 형태일까?
- 첫 번째 기둥에 있는 N개의 원판을 세 번째 기둥으로 옮겨야 함
- 가장 큰 원판부터 차례대로 세 번째 기둥으로 옮김
- 가장 큰 원판을 어떻게 꺼내지?
- 위에 있는 N-1개를 잠시 두 번째 기둥으로 옮기면 됨
- N-1개를 첫 번째 기둥에서 두 번째 기둥으로 옮기고
- N번째 원판을 첫 번째 기둥에서 세 번째 기둥으로 옮기고
- N-1개를 두 번째 기둥에서 세 번째 기둥으로 옮기면 됨



재귀 함수의 예시 - 5

BOJ 11729. 하노이 탑 이동 순서

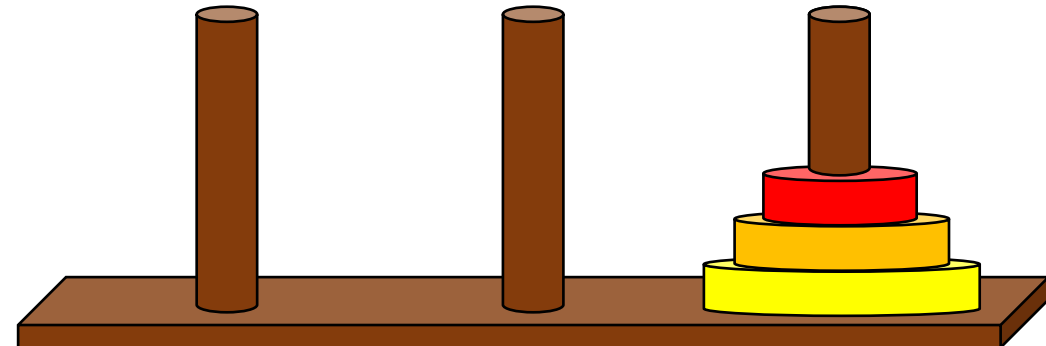
- 최적 전략은 어떤 형태일까?
- 첫 번째 기둥에 있는 N개의 원판을 세 번째 기둥으로 옮겨야 함
- 가장 큰 원판부터 차례대로 세 번째 기둥으로 옮김
- 가장 큰 원판을 어떻게 꺼내지?
- 위에 있는 N-1개를 잠시 두 번째 기둥으로 옮기면 됨
- N-1개를 첫 번째 기둥에서 두 번째 기둥으로 옮기고
- N번째 원판을 첫 번째 기둥에서 세 번째 기둥으로 옮기고
- N-1개를 두 번째 기둥에서 세 번째 기둥으로 옮기면 됨



재귀 함수의 예시 - 5

BOJ 11729. 하노이 탑 이동 순서

- 최적 전략은 어떤 형태일까?
- 첫 번째 기둥에 있는 N개의 원판을 세 번째 기둥으로 옮겨야 함
- 가장 큰 원판부터 차례대로 세 번째 기둥으로 옮김
- 가장 큰 원판을 어떻게 꺼내지?
- 위에 있는 N-1개를 잠시 두 번째 기둥으로 옮기면 됨
- N-1개를 첫 번째 기둥에서 두 번째 기둥으로 옮기고
- N번째 원판을 첫 번째 기둥에서 세 번째 기둥으로 옮기고
- N-1개를 두 번째 기둥에서 세 번째 기둥으로 옮기면 됨



재귀 함수의 예시 - 5

BOJ 11729. 하노이 탑 이동 순서

- $f(n, a, b, c)$: n 개의 원판을 a 에서 c 로 옮기는 과정을 구하는 함수
 - $n-1$ 개의 원판을 a 에서 b 로 이동 $f(n-1, a, c, b)$
 - n 번째 원판을 a 에서 c 로 이동: $f(1, a, b, c)$
 - $n-1$ 개의 원판을 b 에서 c 로 이동 $f(n-1, b, a, c)$
 - 종료 조건: $n = 1$ 일 때 원판 1개를 a 에서 c 로 이동
- 이동 횟수
 - $T(1) = 1$
 - $T(n) = 2T(n-1) + 1$
 - $T(n) = 2^n - 1$

```
void f(int n, int a, int b, int c){
    if(n == 1){
        printf("%d %d\n", a, c);
        return;
    }
    f(n-1, a, c, b);
    f(1, a, b, c);
    f(n-1, b, a, c);
}
```


질문?