KT AIVLE 코딩 스터디 – 완전탐색/그리디 알고리즘

성용제

완전탐색 알고리즘

가능한 모든 경우의 수를 다 체크해서 정답을 찾는 방법

1. 사용된 알고리즘이 적절한가? (문제를 해결할 수 있는가)
2. 효율적으로 동작하는가

1. 입력으로 주어지는 데이터(N)의 크기가 작다
2. 답의 범위가 작고, 임의의 답을 하나 선택했을 때 문제 조건을 만족하는지 역추적 가능
3. 문제 조건 중 하나를 고정하여 풀이를 고민한다.

완전탐색 기법

1. BRUTE FORCE 기법 – 반복/조건문을 활용해 모두 테스트 하는 방법
2. 순열 – N개의 원소 중 R개의 원소를 중복 허용 없이 나열
3. 재귀 – 자기 자신을 호출
4. 비트마스크 – 2진수 표현 기법을 활용
5. BACKTRACKING – 현재 상태에서 가능한 후보군으로 가지치기를 하며 탐색
6. DFS/BFS

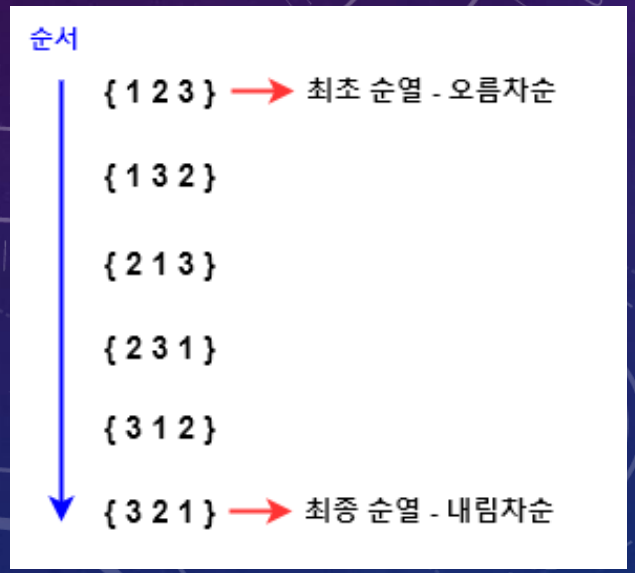
순열

임의의 수열이 있을 때 다른 순서로 연산하는 방법

시간 복잡도는 O(N!)

NEXT_PERMUTATION

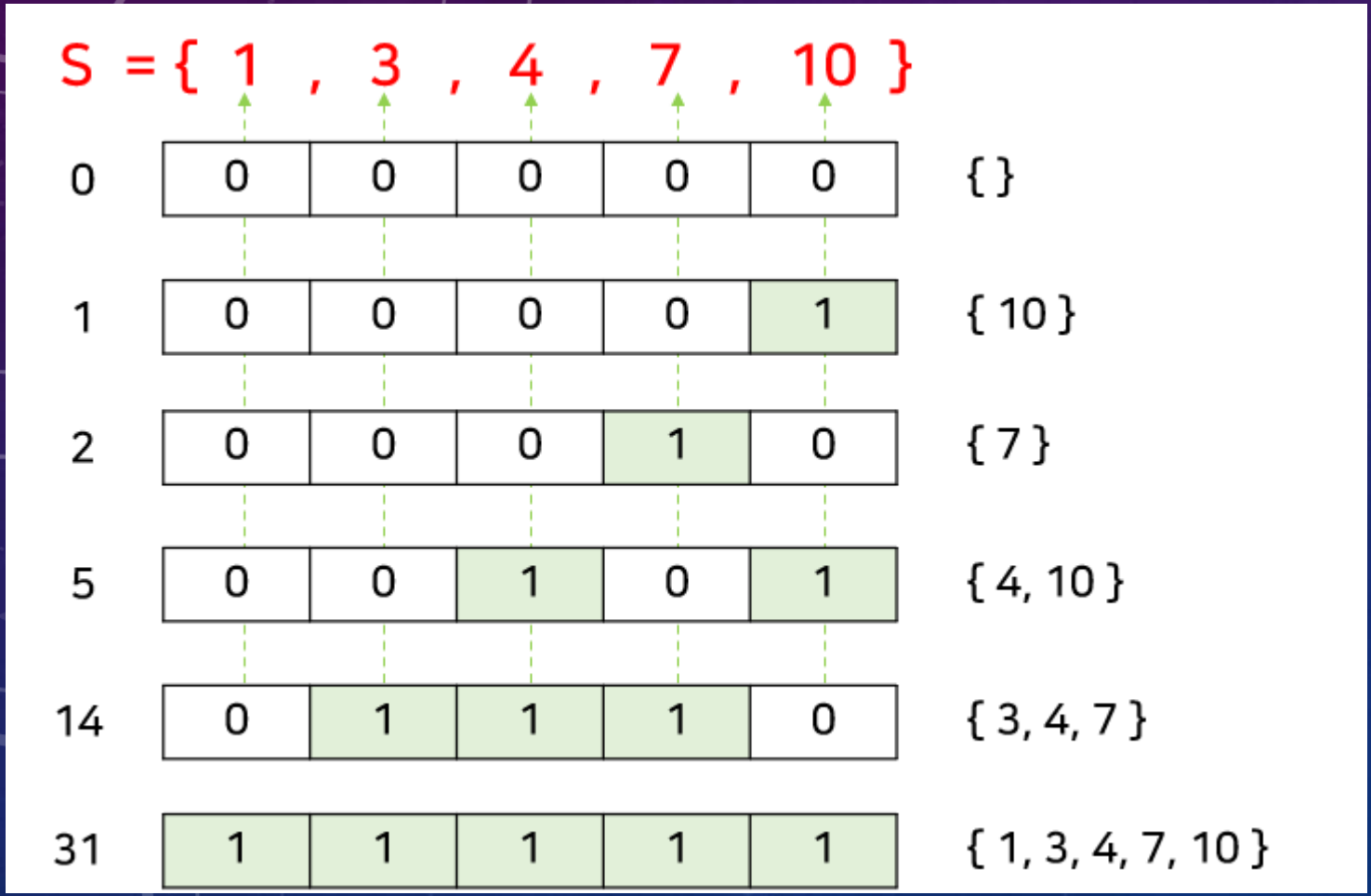
PREV_PERMUTATION




```cpp
#include <iostream>
#include <vector>
#include <algorithm>
#include <string>
using namespace std;
int main()
{
    vector<int> v{1,2,3,4};
    //string v = "1234";

    do{
        for(auto i : v){
            cout<<i<<" ";
        }
        cout<<'\n';
    }while(next_permutation(v.begin(),v.end()));

    vector<int> bit{1,1,0,0};

    do{
        for(int i=0;i<v.size();i++){
            if(bit[i]==1) cout<<v[i]<<" ";
        }
        cout<<'\n';
    }while(prev_permutation(bit.begin(),bit.end()));
}
```

비트마스크

2진수를 이용하는 컴퓨터 연산을 이용



그리디 알고리즘

선택의 순간마다 당장 눈앞에 보이는 최적의 상황만을 쫓아서 결과를 도출하는 방법

1. 선택 절차 : 현제 상태에서 최적의 해답을 선택한다.
2. 적절성 검사 : 선택됨 해가 문제의 조건을 만족하는 지 검사한다.
3. 해답 검사 : 원래의 문제가 해결되었는지 검사한다. 해결되지 않는다면 선택 절차로 돌아간다.

조건

1. 탐욕적 선택 속성 : 앞의 선택이 이후의 선택에 영향을 주지 않는다.
2. 최적 부분 구조 : 문제에 대한 최종 해결 방법은 부분 문제에 대한 최적 문제 해결 방법으로 구성된다.