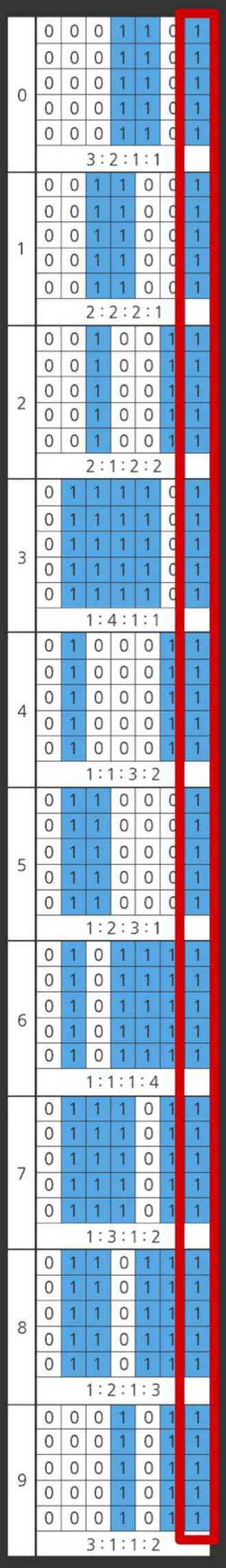
[SWEA]1240 단순 2진 암호코드

0323



암호코드를 발견한뒤 해독 ⟶ 올바른 암호코드인지 확인한다

i) 마지막 암호코드를 찾는다
- 암호코드의 마지막 숫자는 무조건 1이기 때문에 배열 뒤에서부터 확인
- 좌표저장 : arr[row][column]

여기서부터 7자리숫자 8개를 찾는다

ex) 0000<u>0111011</u>0<u>1100010</u><u>1111011</u><u>0110001</u><u>0110001</u><u>0001101</u><u>0010011</u><u>0111011</u>0000000000
0000011101101100010111011011000101100010001101001001101110110000000000
0000011101101100010111011011000101100010001101001001101110110000000000
0000011101101100010111011011000101100010001101001001101110110000000000
0000011101101100010111011011000101100010001101001001101110110000000000
0000011101101100010111011011000101100010001101001001101110110000000000

ii) 찾은 암호코드를 10진수로 변환

ex) <u>0111011</u>
뒤에서부터 순회

n = 0
만약 '1'이면, n = n & 1 << i (i: 0~6)

i = 0 ⟶ '1'
n = n & 1<<0
= 1 ($1_{(2)}$)
i = 1 ⟶ '1'
n = n & 1<<1
= 3 ($11_{(2)}$)
i = 2 ⟶ '0'
n = 3 ($011_{(2)}$)
⋮

iii) 딕셔너리 생성

key : 코드의 배열을 10진수로 변환한 값
value : 코드번호
{13:0, 25:1, ···, 11:9}

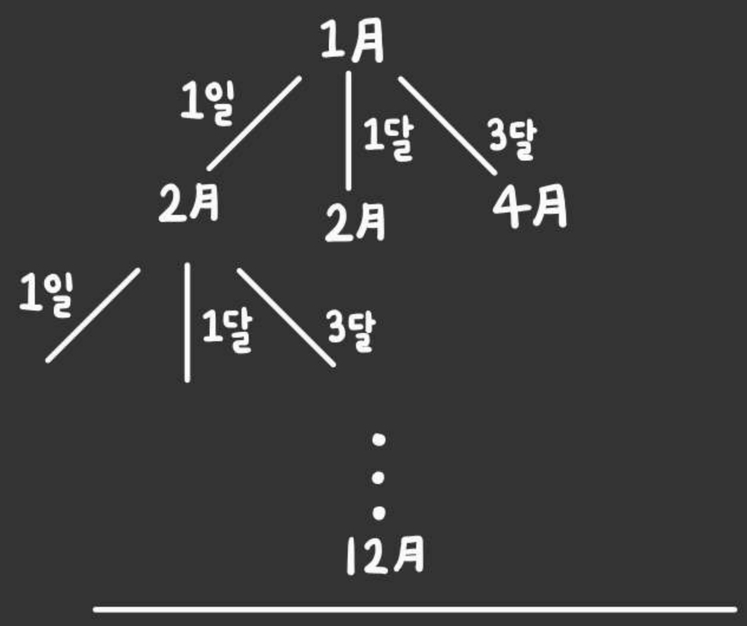
iv) 딕셔너리 활용하여 코드 번호로 변환 (반복)

ex) 59 ⟶ 59를 key로 갖는 딕셔너리 값찾기

v) 올바른 암호인지 확인 후, 출력
(조건문)

[SWEA]1952 수영장

완전탐색으로 이용권 경우의 수를 모두 탐색한다.




```
def check(i, 지불금액, 일일이용권금액, 한달이용권금액, 3달이용권금액)
    지불금액이 최솟값보다 크면리턴
    i > 12이면 리턴
      (단 지불금액<최솟값이면, 최솟값 = 지불금액)
    check(i+1, ···)
    check(i+1, ···)
    check(i+3, ···)
```

종료조건 : 12보다 크면종료
& 이용권 금액 합이 이전에 기록된 합보다 크면 종료

[SWEA] 2105 디저트카페

9	8	9	8
4	6	9	4
8	7	7	8
4	5	3	5

종료조건 : 4번 반복하면 끝
(변이 4개니까)

정답조건
i) 직사각형
- 마지막에 출발한 카페로 돌아온다
- 대변의 길이가 같다
- delta 이동 (범위체크)

ii) 디저트가 중복되면 안된다

iii) 디저트를 가장 많이 먹을수 있는 경로

를 고려하여 완전탐색