```
#====================================
#   Column dominance
#====================================
    # 각 minterm 에 대해 매칭해본다
    col_checklist = [[] for i in range(len(original_minterm))]  #minterm하고 pi동시저장한 곳
    for i in range(len(original_minterm)):
        col_checklist[i].append(original_minterm[i])
        for j in range(len(a2)):
            if epi_test[i][j] == 'right':
                y = a2[j]
                col_checklist[i].append(y)

    col_checklist2 = [] #PI만 저장해둔 곳
    for x in range(len(col_checklist)):
        col_checklist2.append(col_checklist[x][1:])

    dominant = []
    for x in range(len(col_checklist2)):
        for y in range(x+1,len(col_checklist2)):
            list1 = col_checklist2[x]
            list2 = col_checklist2[y]
            intersection = list(set(list1) & set(list2))
            if intersection == list1 and list1 != list2:
                if list2 not in dominant:
                    dominant.append(list2)
            elif intersection == list2 and list1 != list2:
                if list1 not in dominant:
                    dominant.append(list1)

    for x in range(len(col_checklist)):
        if col_checklist2[x] in dominant:
            if col_checklist[x] not in answer:
                done = col_checklist[x]
                answer.append(done[0])


    return answer
```

설명에 앞서, 이 코드 전에는 과제로 진행된 pi,epi찾는 코드가 있었고, a2라는 리스트에 pi가 저장되어 있습니다.

col_checklist: 2차원 리스트로, 이 리스트 안에는 minterm 개수만큼의 리스트가 저장되어 있습니다.

첫 번째 문단은 col_checklist에 minterm과 그에 대응하는 pi가 저장되는 과정입니다. col_checklist안의 각 리스트의 맨 처음자리에는 minterm이 저장되고, 그 뒤에 pi들이 저장됩니다.

두 번째 문단은 col_checklist2라는 리스트에, col_checklist안에 있던 리스트를 그대로 저장하는데 대신 minterm을 제외한 pi들만 저장하는 것입니다.

세 번째 문단은 col_checklist2 안의 리스트들을 비교하는 것입니다. 즉, 누군가 지배하는 관계를 갖는지 확인하는 단계입니다. 각 리스트들의 요소의 교집합을 구하고, 만약 교집합이 어느 하나의 리스트와 완전히 일치한다면 나머지 다른 리스트가 지배하는 것임을 알 수 있습니다. 지배하고 있는 minterm의 pi들을 dominant라는 리스트에 저장해둡니다.

마지막 문단은 답을 구하는 단계입니다. dominant에 저장된 pi들에 해당하는 minterm을 정답에 append합니다. 결과값을 리턴해주면 다음과 같습니다.

```
print(solution([3,4,0,2,3,4]))
print(solution([4,8,0,3,4,5,6,9,11,13]))
print(solution([5,17,0,2,3,5,7,9,11,13,14,15,18,20,22,23,28,29,31]))
print(solution([6,34,0,4,5,6,8,11,14,15,18,20,21,22,23,25,28,29,30,31,33,35,36,38,39,40,42,43,44,46,47,48,53,55,59,61]))
```

```
['000', '010']
['0100', '1011']
['00010', '00111', '01011', '01101', '01111']
['000100', '001111', '010100', '010101', '010110', '010111', '011101', '011110', '011111', '100011', '100110', '101011', '101110', '101111', '110101']
```