백준 6603 로또

202020988 조아영

001 문제 소개

002 아이디어

003 코드 설명

문제 소개

문제

독일 로또 : {1, 2, 3, ..., 49} 중에서 6개를 고름
49가지의 수 중 k(k>6)개의 수를 골라 집합 S를 만든 다음 그 수만 가지고 번호를 선택함

ex) k = 8, S = {1, 2, 3, 5, 8, 13, 21, 34}
=> 수를 고르는 모든 방법
=> [1,2,3,4,8,13], [1,2,3,5,8,21], ... [3,5,8,13,21,34]
=> 총 28가지

집합 S, k가 주어졌을 때 수를 고르는 모든 방법을 구하는 문제

문제 소개

입력

각 테스트 케이스를 한 줄에 입력(마지막 줄은 0)
첫번째 수 : k(6<k<13),
다음 k개의 수 : 집합 S에 포함 되는 수
(S의 원소는 오름차순으로 주어짐)

출력

테스트 케이스마다 수를 고르는 모든 방법 사전 순 출력
테스트 케이스 사이에는 빈줄 출력

아이디어

백트래킹 재귀함수로 구현 !

- 테스트 케이스 무한 반복, 0 입력 시 종료
: while 문 사용하여 입력 받고, 함수 호출

- 로또 6개의 수가 다 채워지면 출력
: 배열에 하나씩 append, pop

- 사전순으로 출력
: 사용된 수 인지 체크하는 배열(check)을 만들어,
T / F를 조절해가며 사전 순으로 출력 하도록

코드 설명

```
while True:
    S = list(map(int, sys.stdin.readline().split()))
    if(S[0] == 0):
        break

    del S[0]

    check = [False] * (len(S))
    num = []

    dfs(0)
    print()
```

테스트 케이스 계속 반복

테스트 케이스 한 줄 입력

첫번째 수 k가 0이면 종료

두번째 수부터가 집합 S의 수 이므로

배열의 길이만큼 check 배열 생성
(T/F) 저장할 배열 -> 초기화는 F로

로또를 저장할 배열

테스트 케이스 마다 빈줄로 구분

```
import sys

def dfs(cnt):
    if(cnt == 6):
        for i in num:
            print(i, end = " ")
        print()
        return

    for i in range(0, len(S)):
        if(check[i]):
            continue

        check[i] = True
        num.append(S[i])
        dfs(cnt + 1)
        num.pop()
        for j in range(i+1, len(S)):
            check[j] = False
```

num 배열(로또 수 저장)의 길이가 6이 되면

배열의 원소 출력

줄바꿈

배열S의 길이 만큼 반복

이미 사용된 수

사용한 수라고 체크 (T)

배열S의 값을 num에 넣음

재귀 호출

로또를 사전 순으로 출력

-> 인덱스 i+1부터 끝까지 F로 체크 함

감사합니다