







10주차 방향그래프

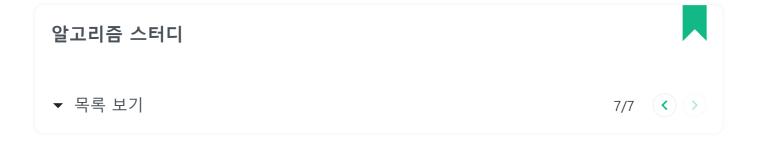
통계 수정 삭제

iming03 · 방금 전 ·

비공기



방향그래프



백준 9466번



문제

이번 가을학기에 '문제 해결' 강의를 신청한 학생들은 텀 프로젝트를 수행해야 한다. 프로젝트 팀원 수에는 제한이 없다. 심지어 모든 학생들이 동일한 팀의 팀원인 경우와 같이 한 팀만 있을 수도 있다. 프로젝트 팀을 구성하기 위해, 모든 학생들은 프로젝트를 함께하고 싶은 학생을 선택해야 한다. (단, 단 한 명만 선택할 수 있다.) 혼자 하고 싶어하는 학생은 자기 자신을 선택하는 것도 가능하다.

학생들이 $(s_1, s_2, ..., s_r)$ 이라 할 때, r=1이고 s_1 이 s_1 을 선택하는 경우나, s_1 이 s_2 를 선택하고, s_2 가 s_3 를 선택하고,..., s_{r-1} 이 s_r 을 선택하고, s_r 이 s_1 을 선택하는 경우에만 한 팀이 될 수 있다.

예를 들어, 한 반에 7명의 학생이 있다고 하자. 학생들을 1번부터 7번으로 표현할 때, 선택의 결과는 다음과 같다.

1	2	3	4	5	6	7
3	1	3	7	3	4	6

위의 결과를 통해 (3)과 (4, 7, 6)이 팀을 이룰 수 있다. 1, 2, 5는 어느 팀에도 속하지 않는다.

주어진 선택의 결과를 보고 어느 프로젝트 팀에도 속하지 않는 학생들의 수를 계산하는 프로그램을 작성하라.

입력

첫째 줄에 테스트 케이스의 개수 T가 주어진다. 각 테스트 케이스의 첫 줄에는 학생의 수가 정수 n ($2 \le n \le 100,000$)으로 주어진다. 각 테스트 케이스의 둘째 줄에는 선택된 학생들의 번호가 주어진다. (모든 학생들은 1부터 n까지 번호가 부여된다.)

출력

각 테스트 케이스마다 한 줄에 출력하고, 각 줄에는 프로젝트 팀에 속하지 못한 학생들의 수를 나타내면 된다.

예제 입력 1 복사

```
2
7
3 1 3 7 3 4 6
8
1 2 3 4 5 6 7 8
```

예제 출력 1 복사

```
3
0
∢ ►
```

풀이: 방향그래프로 표현해서 사이클이 아닌 경우 count 해서 출력

1번 코드

실습코드 고쳐보기

```
#include <stdio.h>
#include <stdlib.h>
#pragma warning(disable:4996)
typedef struct Edge {
    int dest;
    struct Edge* next;
} Edge;
typedef struct Vertex {
    Edge* head;
} Vertex;
typedef struct Graph {
    int numVertices;
    Vertex* array;
} Graph;
Graph* initializeGraph(int numVertices) {
    Graph* graph = (Graph*)malloc(sizeof(Graph));
    graph->numVertices = numVertices;
    graph->array = (Vertex*)malloc((numVertices + 1) * sizeof(Vertex));
    for (int i = 1; i <= numVertices; i++) {</pre>
        graph->array[i].head = NULL;
    }
    return graph;
}
void addEdge(Graph* graph, int src, int dest) {
```

```
Edge* newEdge = (Edge*)malloc(sizeof(Edge));
   newEdge->dest = dest;
   newEdge->next = graph->array[src].head;
   graph->array[src].head = newEdge;
}
int DFS(Graph* graph, int node, int* visited, int* notInTeam, int* inStack) {
   visited[node] = 1;
   inStack[node] = 1;
    Edge* current = graph->array[node].head;
   while (current != NULL) {
        int neighbor = current->dest;
        if (!visited[neighbor]) {
            if (DFS(graph, neighbor, visited, notInTeam, inStack)) {
                return 1; // 싸이클이 발견되면 바로 리턴
            }
        } else if (inStack[neighbor]) {
            notInTeam[neighbor] = 1;
            return 1; // 싸이클이 발견되면 바로 리턴
        current = current->next;
   }
   inStack[node] = 0;
   return 0;
}
int count(Graph* graph, int numVertices) {
    int* visited = (int*)calloc(numVertices + 1, sizeof(int));
    int* notInTeam = (int*)calloc(numVertices + 1, sizeof(int));
   int* inStack = (int*)calloc(numVertices + 1, sizeof(int));
    for (int i = 1; i \le numVertices; i++) {
        if (!visited[i] && !notInTeam[i]) {
            if (DFS(graph, i, visited, notInTeam, inStack)) {
                notInTeam[i] = 1; // 시작 노드도 싸이클에 속한 경우
            }
        }
   }
   int count = 0;
    for (int i = 1; i <= numVertices; i++) {</pre>
        if (notInTeam[i] == 0) {
            count++;
        }
   }
    free(visited);
    free(notInTeam);
    free(inStack);
   return count;
}
int main() {
    int T;
```

```
scanf("%d", &T);
     for (int t = 0; t < T; t++) {
         int n;
         scanf("%d", &n);
         Graph* graph = initializeGraph(n);
         for (int i = 1; i <= n; i++) {
            int dest;
            scanf("%d", &dest);
            addEdge(graph, i, dest);
         int result = count(graph, n);
         printf("%d\n", result);
         for (int i = 1; i <= n; i++) {
            Edge* current = graph->array[i].head;
            while (current != NULL) {
                Edge* next = current->next;
                free(current);
                current = next;
            }
         }
         free(graph->array);
         free(graph);
     }
     return 0;
 }
그런데 자꾸만 출력이
7
3 1 3 7 3 4 6
이 경우 3이 아니라 2가 나온다...
리스트로 하니까 복잡해서 저번시간에 했던 행렬로 구현해보려 n차 시도..
 #include <stdio.h>
 #include <stdlib.h>
 #pragma warning(disable:4996)
 int matrix[100001][2]; // 팀 구성 여부를 나타내는 배열
 int main() {
     int T;
     scanf("%d", &T);
```

while (T--) {
 int n;

scanf("%d", &n);

for (int i = 1; i <= n; i++) {

```
matrix[i][0] = 0; // 팀에 속하지 않음
            matrix[i][1] = 0; // 초기화
        for (int i = 1; i <= n; i++) {
            int dest;
            scanf("%d", &dest);
            matrix[i][0] = dest;
        }
        int count = 0;
        for (int i = 1; i <= n; i++) {
            if (!matrix[i][1]) { // 방문하지 않은 학생일 경우
                int current = i;
               while (!matrix[current][1]) { // 방문한 적이 없는 동안
                   matrix[current][1] = 1; // 방문 표시
                   current = matrix[current][0]; // 다음 학생으로 이동
               }
               int cycleStart = current;
                int cycleCount = 1;
               while (matrix[current][1] == 1) { // 이미 방문한 학생일 경우
                   matrix[current][1] = 2; // 팀에 속하지 않음을 표시
                   count++; // 팀에 속하지 않은 학생 수 증가
                   current = matrix[current][0]; // 다음 학생으로 이동
                   if (current == cycleStart) {
                       break; // 사이클이 완성되면 종료
                   cycleCount++;
               }
            }
        }
        printf("%d\n", count);
     }
     return 0;
 }
뭐에 씌였나...
이것도 2가 나온다
내가 문제를 잘 못 이해했나?
너무 답답하다..
 #include <stdio.h>
 #include <stdlib.h>
 #pragma warning(disable:4996)
 int matrix[100001][2]; // 팀 구성 여부를 나타내는 배열
 int main() {
     int T;
```

```
scanf("%d", &T);
while (T--) {
   int n;
   scanf("%d", &n);
   for (int i = 1; i <= n; i++) {
       matrix[i][0] = 0; // 팀에 속하지 않음
       matrix[i][1] = 0; // 방문 여부 초기화
   }
   // 선택된 학생들의 정보 입력 및 팀 구성 여부 표시
   for (int i = 1; i <= n; i++) {
       int dest;
       scanf("%d", &dest);
       matrix[i][0] = dest;
   }
   int count = 0;
   for (int i = 1; i <= n; i++) {
       if (!matrix[i][1]) { // 방문하지 않은 학생일 경우
           int current = i;
           while (!matrix[current][1]) { // 방문한 적이 없는 동안
              matrix[current][1] = 1; // 방문 표시
              current = matrix[current][0]; // 다음 학생으로 이동
           }
           int cycleStart = current;
           int cycleCount = 1;
           while (matrix[current][1] == 1) { // 이미 방문한 학생일 경우
              matrix[current][1] = 2; // 팀에 속하지 않음을 표시
              int nextStudent = matrix[current][0];
               if (current != nextStudent) {
                  count++; // 팀에 속하지 않은 학생 수 증가
              }
              current = nextStudent; // 다음 학생으로 이동
              if (current == cycleStart) {
                  break; // 사이클이 완성되면 종료
              cycleCount++;
       }
   }
   printf("%d\n", count);
}
return 0;
```

이젠 4가 나온다...

}

알고리즘 스터디 시간에 고쳐봐야지~



강민돌 민돌이의 공부



이전 포스트 今 9주차 그래프 구현

0개의 댓글

댓글을 작성하세요

댓글 작성

