알고리즘 4주차 / 김기현

Q1260> DFS/BFS

|  |
| --- |
| for i in range(M): |
|  | a, b = map(int, input().split()) |
|  | graph[a][b] = 1 |
|  | graph[b][a] = 1 |

* 연결된 노드끼리의 정보를 2차원 배열로 나타냄
* Ex> graph[1][2] = 1 # 1번 노드에 2가 연결됨

Graph[2][1] = 1 # 2번 노드에 1이 연결됨

DFS는 스택 자료구조를 이용하여 탐색하는 깊이 우선 탐색법이다

이 때, 재귀함수의 호출이 스택과 구조적으로 똑같아 재귀함수로 코드를 작성하였다.

Def dfs(v)

* 우선 v라는 노드를 받으면 해당 노드를 방문했다는 표시로 visited[v] = 1을 한다. 그 후, 방문하지 않았으며 v 노드에 연결되어있는 노드를 for문으로 찾은 후 해당 노드를 재귀적으로 호출한다.

BFS는 큐 자료구조를 이용하여 탐색하는 넓이 우선 탐색법이다

우선 큐에 시작 노드를 넣은 후 visited[V] = 1 로 방문 처리를 해준다. 그 후 큐가 완전히 비어질때까지 while문을 돌린다. While문 안에서는 큐의 최상단 노드를 pop한 후, 해당 노드에 연결되어있으며 방문한 적이 없는 노드들을 차례대로 큐에 집어넣어준다. 그리고 다시 최상단 노드를 pop하며 반복한다.

Q1339> 단어 수학

접근법>

입력으로 들어오는 알파벳들의 자릿수들의 합을 딕셔너리에 저장하여 최댓값을 구한다

Ex> GCF ACDEB

딕셔너리(key : value) 안 -> ‘G’ : 100, ‘C’ : 1010, ‘F’: 1, ‘A’:10000, ‘D’:100, ‘E’:10, ‘B’:1

이후 value를 기준으로 내림차순으로 정렬한 후 처음부터 9,8,7…을 곱한 후 합치면 결과가 나온다

코드분석>

|  |
| --- |
| # 단어 수학 |
|  | N = int(input()) |
|  | # 입력 값 받기 |
|  | alpha = [] |
|  | alpha\_collection = {} |
|  | for i in range(N): |
|  | alpha.append(input()) |
|  |  |
|  | for i in range(N): |
|  | digit = 1  # 입력 받은 문자열을 차례대로 검사(1의 자리부터 검사 시작) |
|  | for j in range(len(alpha[i])-1,-1,-1):  # 만약 alpha\_collection이라는 딕셔러니안에 알파벳이 없을 경우 알파벳 key를 추가하며 그 알파벳이 해당하는 자릿수인 digit을 value로 넣는다 |
|  | if alpha[i][j] not in alpha\_collection: |
|  | alpha\_collection[alpha[i][j]]=digit |
|  | # alpha\_collection안에 있는 경우 원래 있던 value에 지금의 digit의 더한다  else: |
|  | alpha\_collection[alpha[i][j]]+=digit  # 그 다음 자릿수 검사 |
|  | digit\*=10 |
|  | # alpha\_collection에 있는 value들을 내림차순으로 정렬한다 |
|  | sort\_alpha = sorted(alpha\_collection.values(),reverse=True) |
|  |  |
|  | sum = 0 |
|  | num = 9  # 내림차순으로 정렬된 value들을 차례대로 9,8,7,… 곱하면서 sum에 더한다 |
|  | for i in range(len(sort\_alpha)): |
|  | sum += sort\_alpha[i]\*num |
|  | num-=1 |
|  | print(sum) |