Description | Hints | Submissions | Discussions | Notes

All Pairs Shortest Path

Difficulty	Time Limit	Memory	Score
[illegible]	2 sec	256000KB	100

80/80 XP | 30/30

Description

We have given an adjacency representation of a **directed weighted graph** and an array of vertices. At each iteration, a vertex is removed from the graph. Vertices are removed in the order given in the array. When the vertex is removed, all the edges that go in and out are also removed.

Print the sum of all pairs shortest path **just before each iteration**.

Input Format

The first line contains integer n ($1 \le n \le 500$) — the number of vertices in the graph.

Next n lines contain n integers each — the graph adjacency matrix: the j-th number in the i-th line a_{ij} ($1 \le a_{ij} \le 10^5$, $a_{ii} = 0$) represents the weight of the edge that goes from vertex i to vertex j.

The next line contains n distinct integers: $x_1, x_2, ..., x_n$ ($1 \le x_i \le n$) — the order of vertices removed from the graph.

Output Format

Print N space-separated numbers, where ith number represents the sum of all pairs shortest path just before ith removal.

Constraints

$1 \le N \le 500$

Sample Input 1

Copy

```
2
0 5
4 0
1 2
```

C++14 | 00:00:00 | 12 px

```
17      }
18      int queries[n];
19      for(int i=0;i<n;i++){
20          cin>>queries[i];
21      }
22      vector<lli>ans;
23      for(int k=n-1;k>=0;k--){
24          for(int i=1;i<=n;i++){
25              for(int j=1;j<=n;j++){
26                  g[i][j]=min(g[i][j],g[i][queries[k]]+g[queries
                    [k]][j]);
27              }
28          }
29          lli temp=0;
30          for(int i=k;i<n;i++){
31              for(int j=k;j<n;j++){
32                  temp+=g[queries[i]][queries[j]];
33              }
34          }
35          ans.push_back(temp);
36
37      }
38      reverse(ans.begin(),ans.end());
39      for(auto i:ans){
40          cout<<i<<" ";
41      }
42
43      return 0;
```

Sample Tests | Manual Tests

Test Case 1 | Test Case 2

Console | Run on Sample