Assignment V — Graph

TA: Jay(jayworking1117@gmail.com)

Deadline: ~~December 19~~, 11:59 pm, 2024
December 26

Problem 1. Rare Species Habitat Protection Plan (45%)

Description

As a wildlife researcher, you are tasked with discovering numerous rare species in a complex ecosystem. This ecosystem is represented as a graph with multiple interconnected habitats, each associated with an observation value. The time required to travel between habitats varies due to different distances. Your challenge is to develop a solution that determines the most effective exploration route. The calculation method is as follows. Each time you start exploring and reach a new habitat, its observation value is added to the total. The route should start and end at a designated point, must not exceed the traversal time limit, and should maximize the observation value of the species across all visited habitats. While you can visit the same habitat multiple times, each habitat's value is counted only once in the total observation value. However, when traveling between habitats and returning, you must account for the traversal time in both directions. Upon completing the route and returning to the starting point, you will successfully identify the most effective exploration route.

Input:

- The first line contains an integer N ($1 \le N \le 100$), representing the number of interconnected habitats in the ecosystem. Each habitat is numbered from 0 to N-1. If N = -1, the input ends, and no further processing is required.
- The second line contains N integers, representing the observation values array, where $0 \le values[i] \le 1000$ for each habitat i, indicating the observation value at each habitat.
- The third line contains an integer E ($1 \le E \le 200$), indicating the number of edges connecting the habitats.

- The next E lines each contain three integers: from, to, and time (0 ≤ from, to < N; 1 ≤ time ≤ 100), describing a bidirectional path between habitats ' from' and 'to' that takes 'time' hours to traverse.
- The next line contains an integer startpoint (0 ≤ start_point < N), specifying the designated starting and ending point for the exploration route.
- The last line contains an integer max_time (1 ≤ max_time ≤ 1000), representing the strict time limit for the exploration in hours.

Output:

Return the maximum total observation value achievable within the given maximum time.

Test case examples

Sample Input	Sample Output
6 10 25 15 20 30 35 9 0 1 5 0 2 10 1 2 5 1 3 15 2 3 10 2 4 20 3 4 5 3 5 15 4 5 10 0 40 -1	70

Example:

Given the first test case above, Figures 1 shows the illustration of a weighted graph with the input content from the first test case. Figures 2-3 show the result of finding the maximum total observation value from different starting points, respectively.

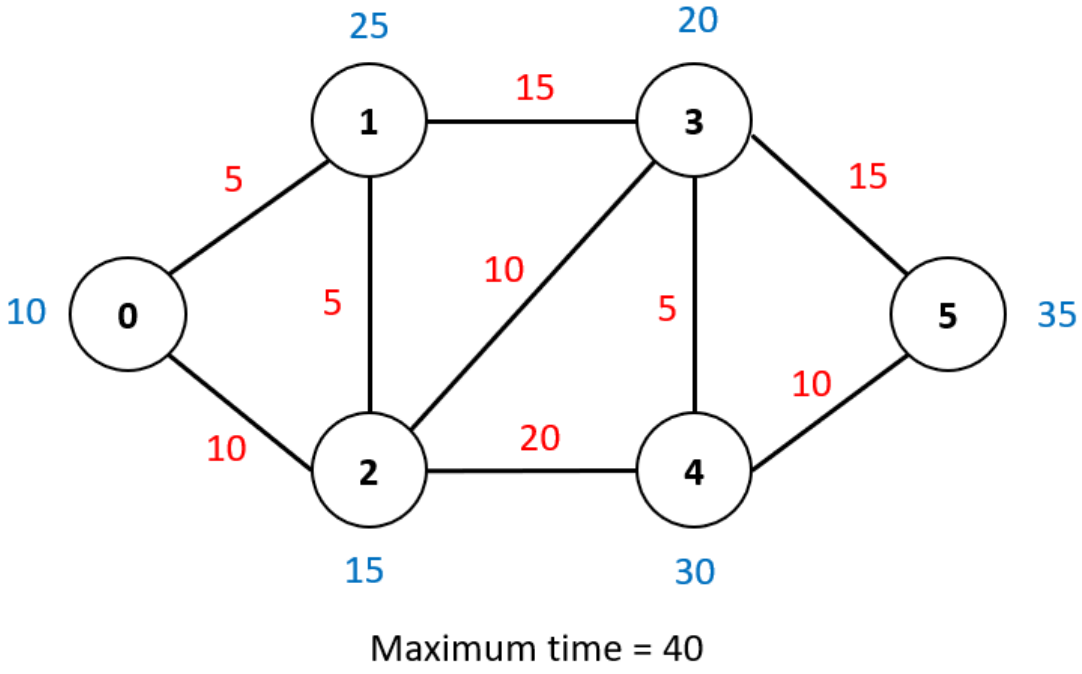


Figure 1. An example of a weighted graph with the input of the first test case.

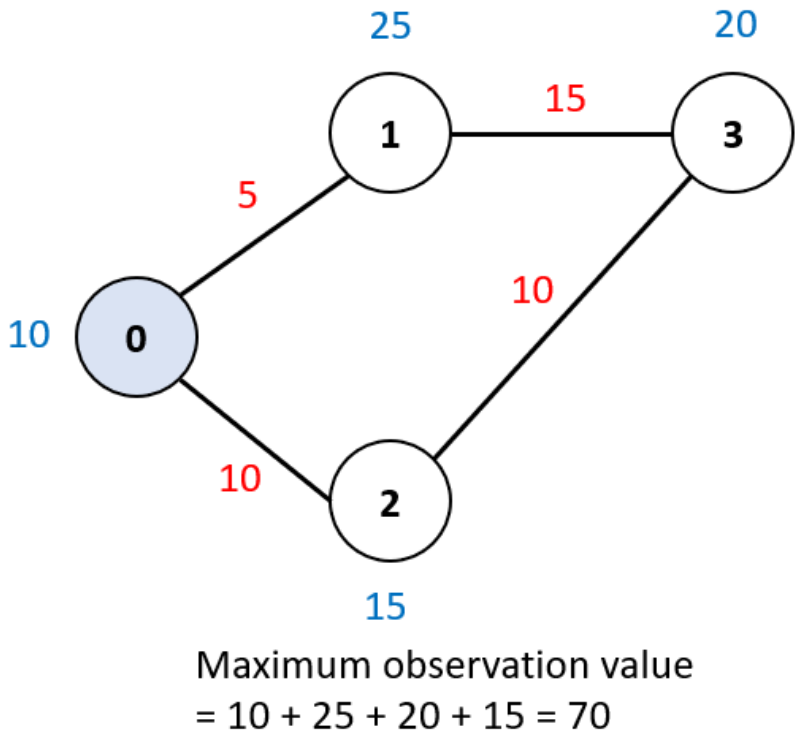


Figure 2. An example of finding the maximum observation value 70 from the start point 0 in the original graph.

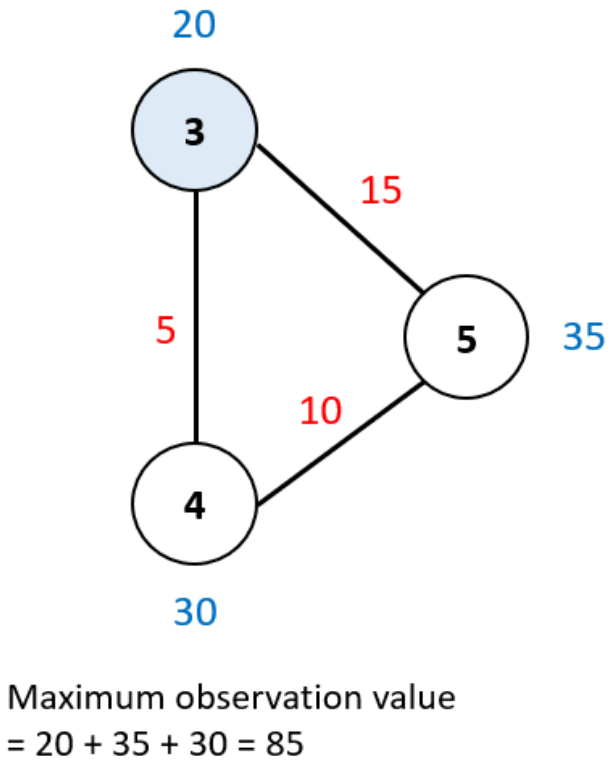


Figure 3. An example of finding the maximum observation value 85 from the start point 3 in the original graph.

Problem 2. Network Intrusion Path Analysis (50%)

Description

You are tasked with analyzing a computer network for potential intrusion paths. The network is represented as a directed graph, where each node is a computer or server with a specific security level (represented by lowercase letters a-z) and a certain number of vulnerabilities. Your goal is to find the highest risk intrusion path in the network. The procedure for path selection and analysis is as follows:

1. First, find all possible paths in the network and calculate the total number of vulnerabilities for each path. Select the path that has the maximum sum of vulnerabilities.
2. Then, for the chosen path, calculate:
   - The security level that appears most frequently in this path.
   - The maximum count of nodes with the same security level in the path.
   - The sequence of nodes from the start to the end in this path.
3. If a cycle is detected in the network, indicating a potential infinite attack loop, your program should report this instead.

During the path selection process, you may encounter special situations:

1. If there are several paths with the same total number of vulnerabilities, select the path with the highest number of nodes that have the same security level.
2. If there are still multiple paths, where the number of nodes with the same security level is identical, choose the path with the lowest security level (considering *z* as the lowest and most dangerous level, and *a* as the highest and safest level).
3. If multiple paths have the exact same number of nodes and the same security levels, select the path with the smallest node index.

By following these criteria, you complete the task of analyzing the computer network for potential intrusion paths.

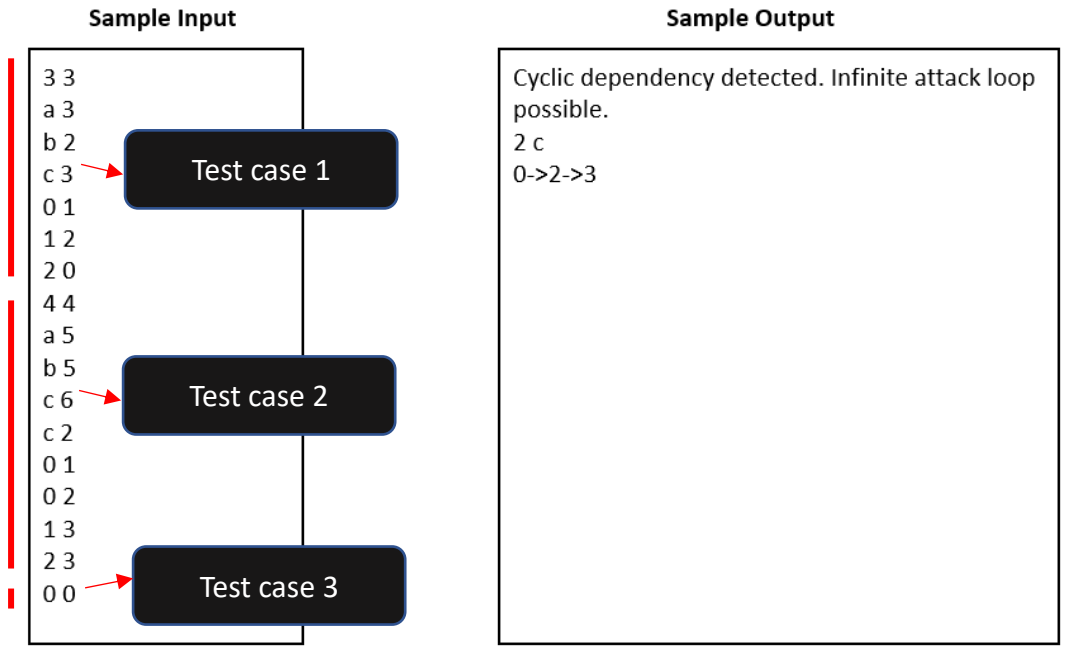
Input:

- The first line contains two integers n and m ($1 \le n \le 10,000$, $0 \le m \le 100,000$), where n is the number of nodes and m is the number of edges in the network.
- The next n lines each contain a character and an integer: the security level (a lowercase letter from 'a' to 'z') and the number of vulnerabilities ($0 \le$ vulnerabilities $\le 1,000$) for each node.
- The following m lines each contain two integers u and v ($0 \le u, v < n$), representing a directed edge from node u to node v.
- The input ends when both n and m are 0.

Output:

- If a cycle is detected in the network, output "Cyclic dependency detected. Infinite attack loop possible."
- Otherwise, output two lines:
  1. Two space-separated values: X Y, where X is the count of the most frequent security level and Y is its security level (a lowercase letter).
  2. The nodes of the highest risk path in order from the start to the end, separated by "->".

Test case examples



Examples:

Given the second test case, Figures 1 and 2 illustrate the process for determining the output.

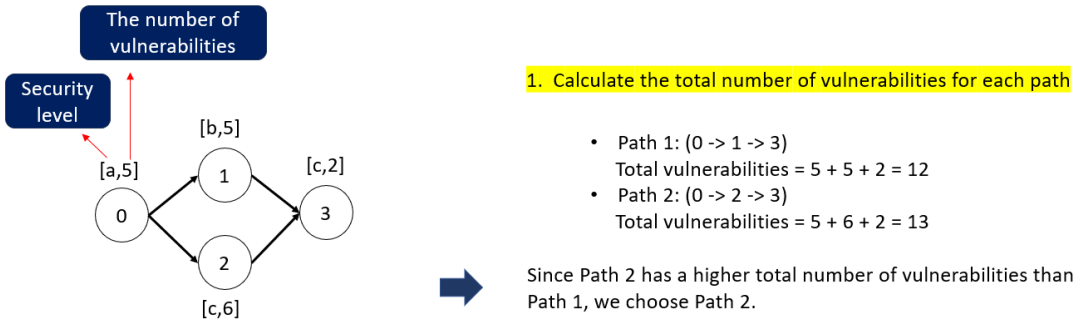


Figure 1. The example of calculating the total number of vulnerabilities for each path.

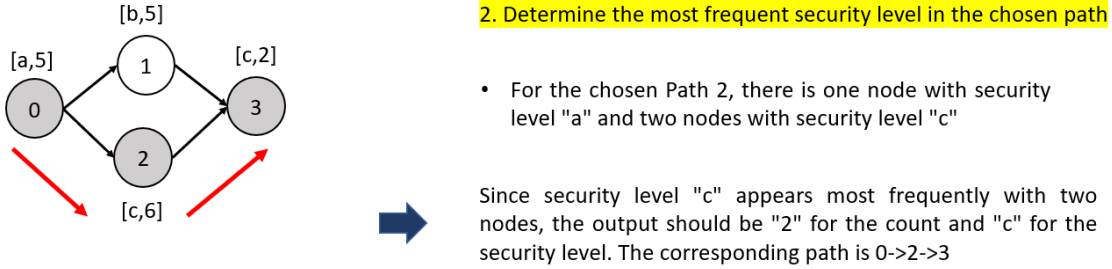


Figure 2. Determine and output the most frequent security level in the chosen path.

Examples of special cases:

Figures 3 shows if there are several paths with the same total number of vulnerabilities, select the path with the highest number of nodes that have the same security level.

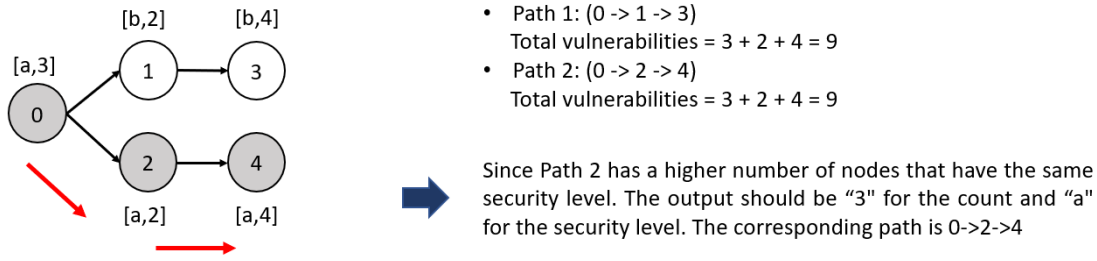


Figure 3. The example of several paths with the same total number of vulnerabilities.

Figures 4 shows if there are still multiple paths where the number of nodes with the same security level is identical, choose the path with the lowest security level.

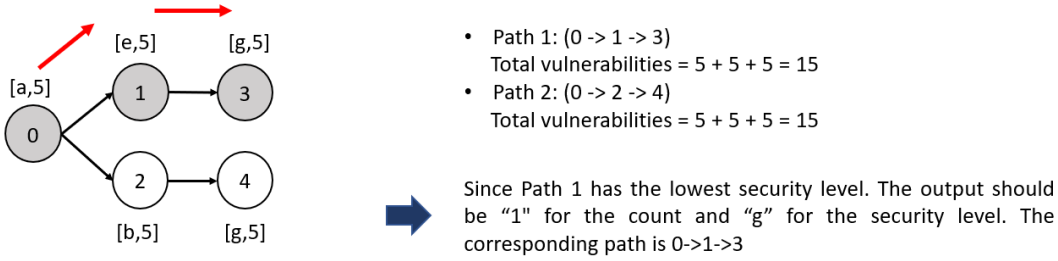


Figure 4. The example of several paths where the number of nodes with each security level is identical.

Figures 5 shows if multiple paths have the exact same number of nodes and the same security levels, select the path with the smallest node index.

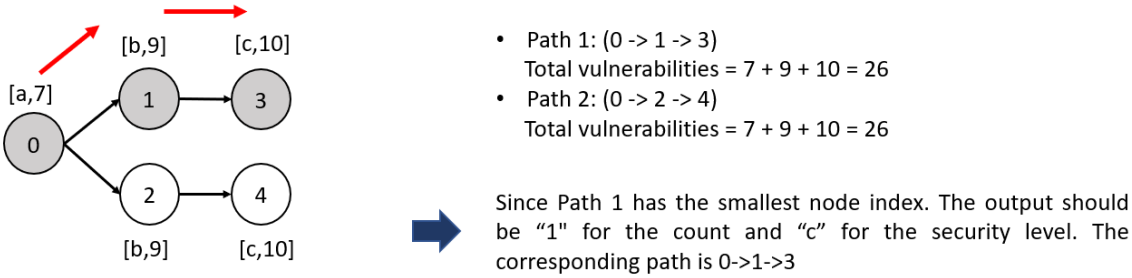


Figure 5. The example of several paths with the exact same number of nodes and security levels.

Readme, comments, and coding style

An indicator for good source code is readability. To keep source code maintainable and readable, you should add comments to your source code where reasonable. A consistent coding style also helps a lot when tracing the source code. For this assignment, please also compose a readme file in *.txt format and name it as "README.txt". This file should contain a brief explanation of how to use your program. Please remember to have your source code comments and readme file in English.

Submission

To submit your files electronically, log in to the DomJudge website through the following URL: http://domjudge.csie.io:54321
Press the submit button and choose the homework questions you want to submit. After submitting your code, DomJudge will give you a result to tell you whether your code is correct or not. Please note that your code will be evaluated by different sets of test cases. Please make sure your code can work correctly based on the description above. Additionally, you must compress your code and the README file into a **zip** file and upload it to Ecourse2. Otherwise, you will get zero points.

ATTENTION: Do NOT copy others' work or you will get a zero.

Grading policies

The TA(s) will mark and give points according to the following rules:

45% - Problem1.
50% - Problem2.
5% - Readme, comments, and coding style.