**BÁO CÁO ĐÁNH GIÁ BÀI TẬP VỀ NHÀ CỦA NHÓM 6**

**Đề bài:**

Input:

* n, m là số đỉnh và số cạnh (1<n<1000; 0<m<5000).
* Trong m dòng,  nhập u, v, w với u, v là 2 đỉnh của cạnh và w là trọng số của cạnh (-1<u,v<n;  0<w<1000).
* A, B, K là đỉnh xuất phát, đỉnh đến và chi phí yêu cầu của đường từ A đến B (-1<A, B<n; 0<K<5000)

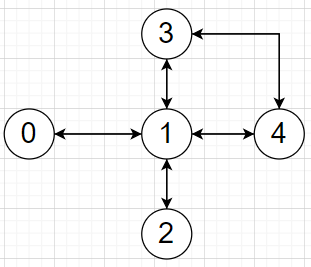
Output: Số con đường từ A đến B có chi phí bằng K, đường đi không được qua 1 điểm 2 lần

**Đánh giá:**

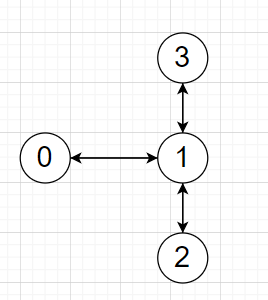
1. **Phương pháp DFS (Nhóm 1, 2, 7):**  
   Nếu như ta dùng DFS, ta có thể xác định ra được những con đường không tạo thành chu trình với chính nó.  
   Cả 2 nhóm đều rất thành thạo trong việc sử dụng Matrix biểu diễn đồ thị.  
   Kết quả bài toán sau khi các bạn giải thỏa yêu cầu của đề bài.

1. **Phương pháp BFS (Nhóm 3, 4, 5, 9, 11):**

Phương pháp BFS sẽ rất khó để xác định đường đi chứa chu trình.  
Nhưng các bạn đã sử dụng 3 biến Nodetrước - Nodehiệntại - Nodesau để biểu diễn cho 1 phần tử visited. Phương pháp này có hiệu quả trong việc xác định chu trình có nhiều hơn 2 node.

Với K = 5 thì phương pháp này đã loại được đường đi 0->1->3->4->1->2.

Nhưng nếu chu trình của mình chỉ có 2 node thì trong một vài trường hợp sẽ không đưa ra kết quả chính xác.

Vd:

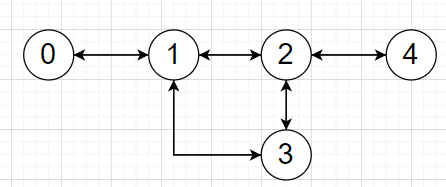
Với đồ thị như hình vẽ, trọng số mỗi cạnh đều bằng 1, hỏi đường đi từ 0 đến 2 có K = 4, kết quả sẽ là không có đường đi nào. Nhưng kết quả của phương pháp này lại đưa ra kết quả 1 đường đi từ 0 đến 2: 0 -> 1 -> 3 -> 1 ->2. Tại vì con đường này có 1 chu trình 1<->3 chỉ có 2 node.

Tóm lại, các nhóm đã chọn con đường BFS khác với ý đồ của người ra đề, nhưng đã rất cố gắng phát triển BFS theo yêu cầu đề bài. Nhưng do thiếu dữ liệu test nên đã không nhận ra sai sót.

1. **Phương pháp BFS dùng PriorityQueue (Nhóm 8, 12):**

Phương pháp dùng PriorityQueue khá tương tự Dijkstra, ở việc tìm ra con đường có trọng số nhỏ nhất hiện tại để tiếp tục thực hiện quan sát tại node cuối đường đó. Phương pháp này giúp cho việc loại bỏ tất cả những con đường khác trong queue có trọng số lớn hơn K.

Tương tự với vấn đề khi dùng BFS, rất khó để xác định chu trình trong đường đi. Các nhóm này đã sử dụng mỗi phần tử trong visited là một con đường trực tiếp từ 1 node đến 1 node để tránh cho việc đi lại con đường đó, theo hướng ngược lại. Nhưng nếu chúng ta thử đi trên con đường này, và đánh dấu đoạn đường đã đi rồi, không được đi ngược lại trên những đoạn đường đó, thì đã chặn đi việc đi trên con đường đó trên những hướng đi khác. Việc đó sẽ khiến cho số con đường tìm thấy giảm đi rất nhiều.

Vd: 

Tìm đường đi từ 0 đến 4 có K = 4. Có kết quả là 0->1->->2->4  
Tại node 1, ta không cần phải hướng nhìn tới node 0, vì đoạn đường 0->1 đã được đánh dấu và không cho đi ngược lại theo hướng 1->0.  
Nhưng cũng tại node 1, ta đưa node 2, 3 vào queue. Ta đi tới node 2 trước, và đánh dấu đoạn đường 2->3. Rồi ta đi tới node 3, lúc này, đoạn đường 3->2 đã bị chặn.

**Tổng kết lại:**

* Phương pháp BFS do các bạn đề xuất vẫn chưa thực sự giải quyết chính xác việc phát hiện chu trình trên đường đi.
* Nhưng với việc thử sức trên một giải thuật mới sẽ giúp các bạn phát triển khả năng thiết kế thuật toán.