1. Bài toán

a)Yêu cầu: Hãy tô màu đồ thị sao cho số lượng màu càng ít càng tôt.

b)Input:

Cho đồ thị G=(V,E), với |V| < 10,000 đỉnh.
Từ file dothi.txt trong đó:

- Dòng đầu tiên ghi hai số nguyên n và m là số đỉnh và số cạnh của đồ thị;

- Ở m dòng tiếp theo, mỗi dòng ghi 2 số nguyên tương ứng với một cạnh của đồ thị.

c)Output:

File dothitomau.txt với

- dòng đầu tiên là số màu tối đa bạn dùng để tô (các màu đánh số từ 1,2,…);

- Ở n dòng tiếp theo, dòng thứ i là màu để tô đỉnh i.

2.Thuật toán sử dụng

Chương trình này sử dụng thuật toán tô màu đồ thị DSatur (Degree of Saturation) để giải quyết bài toán tô màu đồ thị.

Đầu tiên, khai báo biến:
-n: số đỉnh của đồ thị
-m: số cạnh của đồ thị
-adj: danh sách kề của đồ thị.
-degree: mảng lưu trữ bậc của các đỉnh.
-saturation: mảng lưu trữ độ bão hòa của các đỉnh.
-colors: mảng lưu trữ màu được sử dụng cho từng đỉnh.

Tiếp theo tạo hàm DSatur():

Hàm DSatur() sẽ được gọi từ hàm main() để thực hiện thuật toán tô màu đồ thị DSatur. Trong hàm Dsatur() này, chương trình sẽ thực hiện các bước sau:

B1: Khởi tạo mảng "degree" chứa số bậc của từng đỉnh, mảng "colors" chứa màu của từng đỉnh (ban đầu toàn bộ các đỉnh chưa được tô màu, khởi tạo giá trị ban đầu cho mảng "colors" bằng -1) và "set used_colors" để đánh dấu các màu đã được sử dụng.

B2: Đọc vào danh sách kề của đồ thị từ đầu vào và tính độ bão hoà của từng đỉnh. (Độ bão hòa của một đỉnh trong đồ thị tô màu là số lượng màu khác nhau được sử dụng để tô màu các đỉnh kề với nó.)

B3: Lặp lại cho đến khi tất cả các đỉnh đều được tô màu:

-Tìm đỉnh có độ bậc lớn nhất và chưa được tô màu hoặc có độ bậc bằng nhau thì chọn đỉnh có độ bậc sắp xếp giảm dần cao nhất.
-Tìm tập các màu đã được sử dụng bởi các đỉnh kề với đỉnh được chọn.
-Chọn màu chưa được sử dụng trong tập trên và gán cho đỉnh được chọn.
-Cập nhật độ bão hoà của các đỉnh kề với đỉnh được chọn.
B4: Trả về số màu ít nhất cần tô và màu của từng đỉnh.

B5: Sau khi hàm DSatur() được thực thi, hàm main() sẽ in ra màn hình thông tin về số màu ít nhất cần thiết để tô màu đồ thị và màu của từng đỉnh.

!!! Lưu ý rằng: Chương trình giả định rằng đồ thị là không có hướng và số đỉnh của đồ thị không vượt quá 100000. Nếu đầu vào không phù hợp với các giả định này, chương trình có thể không hoạt động.

3. Đánh giá thuật toán

-Hiệu suất: Thuật toán DSatur có hiệu suất tốt và được chấp nhận rộng rãi trong việc giải quyết bài toán tô màu đồ thị. Độ phức tạp của thuật toán là O(n^2), với n là độ lớn của đồ thị, nghĩa là nó chạy nhanh với các đồ thị có số đỉnh ít.

-Chính xác: Thuật toán DSatur cho kết quả tối ưu khi giải quyết bài toán tô màu đồ thị. Nó đảm bảo rằng không có hai đỉnh kề nhau nào được tô cùng màu. Với các trường hợp đặc biệt, ví dụ như đồ thị hai phân và cây, DSatur cho kết quả tối ưu ngay cả với độ phức tạp thấp.

-Dễ sử dụng: Thuật toán DSatur có cấu trúc đơn giản, và rất dễ sử dụng. Nó có thể được cài đặt bằng ngôn ngữ lập trình thông dụng như C++, Java, Python,...

=>Có thể nói, DSatur là một thuật toán tốt cho bài toán tô màu đồ thị với hiệu suất đáng kể và độ chính xác cao. Nó là một trong những phương pháp đơn giản và dễ sử dụng nhất để giải quyết bài toán này. Tuy nhiên, đối với các đồ thị lớn hoặc đặc biệt có thể cần phải sử dụng các thuật toán khác để đạt được kết quả tối ưu trong thời gian hợp lý.

4. Đánh giá kết quả

-Thỏa mãn tính đúng đắn của cách tô màu. Luôn cho ra số lượng màu thỏa mãn 2 đỉnh kề nhau không trùng màu.
-Tuy nhiên, có thể với một vài đồ thị sẽ không cho ra được số màu ít nhất cần tô.

5. Tài liệu tham khảo

-https://www.geeksforgeeks.org/dsatur-algorithm-for-graph-coloring/