Bài Tập Lớn 2
Năm học 2022-2023-HK 3

HIỆN THỰC MỘT SỐ GIẢI THUẬT TRÊN ĐỒ THỊ

1. TÀI LIỆU CUNG CẤP CHO SINH VIÊN

Sinh viên download file assigment2.zip từ trang BKeL của môn học. Khi giải nén file này, sẽ có được các file sau:

input.txt	Một file input ví dụ
main.cpp	Chương trình chính
common.h	Định nghĩa lớp digitList và Integer
assignment2.cpp	Mã nguồn hiện thực bởi sinh viên
Assignment2.pdf	File pdf mô tả nội dung bài tập lớn

*Lưu ý rằng **sinh viên không được phép thay đổi file** main.cpp, common.h **khi hiện thực chương trình**. Ngoài ra, các hàm do sinh viên viết không được xuất bất kỳ dữ liệu nào ra màn hình khi thực thi.*

Để dịch và thực thi chương trình, sinh viên chứa cả 4 files main.cpp, common.h và assignment2.cpp trong cùng một thư mục; sau đó chỉ cần dịch và thực thi **duy nhất** file main.cpp. Mọi công việc cần phải làm sẽ được hiện thực trong file assignment2.cpp, tuy nhiên không cần dịch và thực thi file này.

2. NỘP BÀI

Sinh viên chỉ nộp đúng một file*: **(1) assignment2-MSSV.cpp (MSSV là mã số sinh viên, tên file phải được viết thường***. **Tất cả các file khác sẽ bị tự động xoá khi chấm bài.** File được nộp phải là file chương trình gốc, sinh viên không được nén file khi nộp bài.

Khi chấm bài sẽ sử dụng môi trường lập trình CodeBlocks 13.12, nên yêu cầu sinh viên sử dụng môi trường này để lập trình. Sinh viên phải kiểm tra chương trình của mình trên môi trường này trước khi nộp bài.

Thời hạn chót nộp bài là **23h00 ngày 23/08/2023**. Bài nộp trễ sẽ KHÔNG được chấp nhận.

Địa chỉ nộp bài: **ctdlgt.bku@gmail.com**.

3. XỬ LÝ GIAN LẬN

Bài tập lớn phải được sinh viên **TỰ LÀM**. Sinh viên sẽ bị coi là gian lận nếu:

- Có sự giống nhau bất thường giữa mã nguồn của các bài nộp. Trong trường hợp này, **TẤT CẢ** các bài nộp đều bị coi là gian lận. Do vậy sinh viên phải bảo vệ mã nguồn bài tập lớn của mình.
- **Sinh viên không được copy mã nguồn từ bất cứ nguồn nào.**
- **Trong trường hợp bị phát hiện gian lận, sinh viên sẽ nhận điểm 0 bài tập lớn.**

KHÔNG CHẤP NHẬN BẤT KỲ GIẢI THÍCH NÀO VÀ KHÔNG CÓ BẤT KỲ NGOẠI LỆ NÀO!

4. HƯỚNG DẪN BAN ĐẦU

Bài tập lớn 2 được phát triển từ bài Lab 5, gồm tổng cộng 5 chức năng: (1) duyệt đồ thị theo chiều sâu; (2) duyệt đồ thị theo chiều rộng; (3) tìm thứ tự topo theo chiều sâu; (4) tìm thứ tự topo theo chiều rộng; (5) tìm cây phủ tối thiểu.

Sinh viên hoàn thiện các hàm trong tập tin assignment2.cpp, có thể viết thêm một số hàm phụ trợ trong tập tin này.

Trong tập tin assignment2.cpp KHÔNG được định nghĩa thêm bất kỳ cấu trúc dữ liệu nào, các cấu trúc dữ liệu cần thiết đã được định nghĩa trong tập tin common.h

(1) Hàm `string depthFirstTraversal(Graph &myGraph, VertexType startVertex)`

Hàm này trả về kết quả duyệt đồ thị theo chiều sâu dưới dạng chuỗi. Lưu ý, trong chuỗi kết quả, mỗi đỉnh cách nhau ĐÚNG MỘT KHOẢNG TRẮNG.

(2) Hàm `string breadthFirstTraversal(Graph &myGraph, VertexType startVertex)`

Hàm này trả về kết quả duyệt đồ thị theo chiều rộng dưới dạng chuỗi. Lưu ý, trong chuỗi kết quả, mỗi đỉnh cách nhau ĐÚNG MỘT KHOẢNG TRẮNG.

(3) Hàm `string depthTopoSort(Graph &myGraph)`

Hàm này trả về kết quả liệt kê thứ tự topo theo chiều sâu dưới dạng chuỗi. Lưu ý, trong chuỗi kết quả, mỗi đỉnh cách nhau ĐÚNG MỘT KHOẢNG TRẮNG.

(4) Hàm `string breadthTopoSort(Graph &myGraph)`

Hàm này trả về kết quả liệt kê thứ tự topo theo chiều rộng dưới dạng chuỗi. Lưu ý, trong chuỗi kết quả, mỗi đỉnh cách nhau ĐÚNG MỘT KHOẢNG TRẮNG.

(5) Hàm `string minSpanTree(Graph &myGraph, VertexType startVertex)`

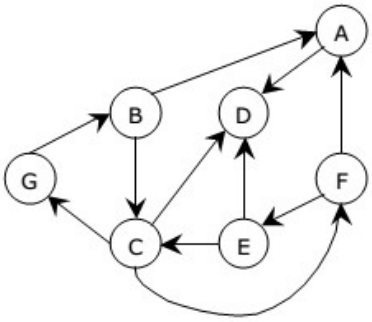
Hàm này trả về tổng trọng số của cây phủ tối thiểu dưới dạng chuỗi. Do các testcase chỉ gồm các số nguyên, nên tổng trọng số này sẽ là số nguyên. Sau đó, phải chuyển số nguyên này thành chuỗi và trả về chuỗi.

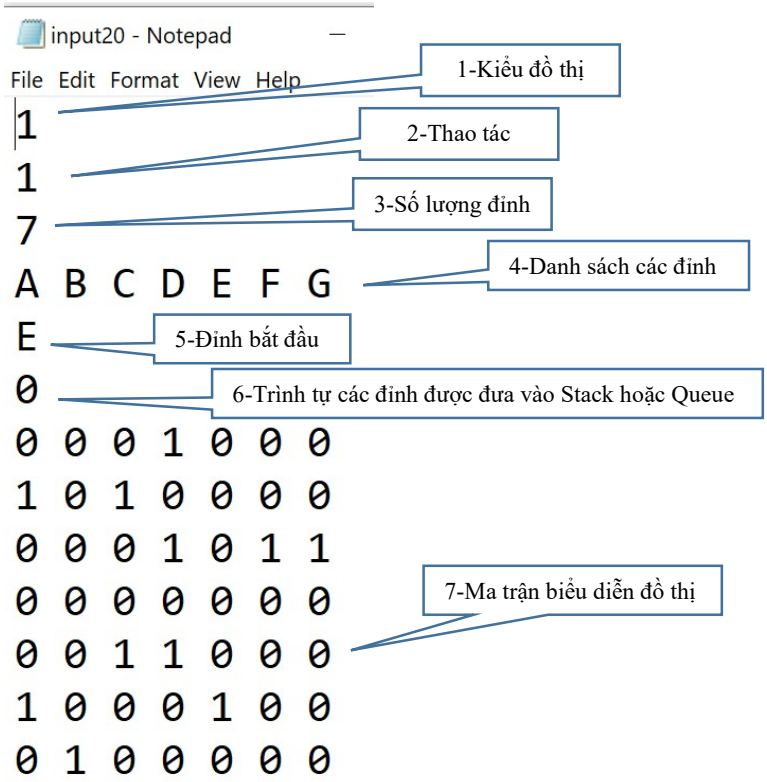
(6) Hàm `int readFile(Graph &myGraph, char* filename )`

Hàm này có chức năng đọc file dữ liệu vào trong cấu trúc myGraph. Sinh viên tham khảo bài tập lớn 1 để biết cách đọc file văn bản.

5. ĐỊNH DẠNG CỦA FILE SỐ LIỆU NHẬP

Hình 1 là một ví dụ về đồ thị và file dữ liệu tương ứng của nó





Hình 1

1) Kiểu đồ thị

   Giá trị này bằng 0 → đồ thị vô hướng
   Giá trị này bằng 1 → đồ thị có hướng

2) Thao tác
   Giá trị này bằng 1 → duyệt đồ thị theo chiều sâu
   Giá trị này bằng 2 → duyệt đồ thị theo chiều rộng
   Giá trị này bằng 3 → liệt kê thứ tự topo theo chiều sâu
   Giá trị này bằng 4 → liệt kê thứ tự topo theo chiều rộng
   Giá trị này bằng 5 → tìm cây phủ tối thiểu

3) Số lượng đỉnh

   Giá trị này cho biết đồ thị có tổng cộng bao nhiêu đỉnh

4) Danh sách các đỉnh

   Liệt kê tên tất cả các đỉnh của đồ thị. Lưu ý rằng các đỉnh KHÔNG cần phải liệt kê theo thứ tự bảng chữ cái, mà có thể liệt kê theo thứ tự bất kỳ. Thứ tự liệt kê các đỉnh sẽ KHÔNG ảnh hưởng đến kết quả của các thao tác.
   Ví dụ, cùng đồ thị như *hình 1*, ta cũng có thể có file dữ liệu như sau:

```
1
1
7
E C D B A F G
E
0
0 1 1 0 0 0 0
0 0 1 0 0 1 1
0 0 0 0 0 0 0
0 1 0 0 1 0 0
0 0 1 0 0 0 0
1 0 0 0 1 0 0
0 0 0 1 0 0 0
```

5) Đỉnh bắt đầu

   Với một số thao tác, cần phải chỉ định đỉnh bắt đầu, chẳng hạn như thao tác duyệt đồ thị

6) Trình tự đưa các đỉnh vào Stack hoặc Queue

   Một số thao tác như duyệt đồ thị hoặc tìm thứ tự topo, cần phải chỉ định trình tự đưa các đỉnh vào Stack hoặc Queue.
   Giá trị này bằng 0 → Các đỉnh được đưa vào theo thứ tự từ bé đến lớn (xếp theo bảng chữ cái)
   Giá trị này bằng 1 → Các đỉnh được đưa vào theo thứ tự từ lớn đến bé (xếp theo bảng chữ cái)

7) Ma trận biểu diễn đồ thị

   Đối với ma trận không có trọng số thì dùng giá trị 0 và 1, nếu có trọng số thì sử dụng trọng số.