**TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI**

**Viện Công nghệ Thông tin và Truyền thông**

****

**BÁO CÁO PROJECT I**

**Đề tài:**

Cấu trúc dữ liệu các tập không giao nhau

và ứng dụng vào cài đặt thuật toán Kruskal.

Sinh viên thực hiện: Nguyễn Đức Duy.

MSSV: 20140724.

Mã lớp: 655794.

Giảng viên hướng dẫn: PGS.TS Nguyễn Đức Nghĩa.

**HÀ NỘI – 2016**

Mục lục

[**1.** **Các vấn đề lý thuyết**. 3](#_Toc469300737)

[**1.1.** **Mô tả bài toán.** 3](#_Toc469300738)

[**1.2.** **Thuật toán Kruskal.** 3](#_Toc469300739)

[**1.3.** **Cấu trúc dữ liệu các tập không giao nhau.** 4](#_Toc469300740)

[**1.4.** **Triển khai rừng các tập không giao nhau.** 5](#_Toc469300741)

[**1.5.** **Thuật toán Kruskal sử dụng cấu trúc dữ liệu các tập không giao nhau.** 7](#_Toc469300742)

[**2.** **Hệ thống chương trình giải bài toán.** 8](#_Toc469300743)

[**2.1.** **Mô tả thiết kế chức năng.** 8](#_Toc469300744)

[**2.2.** **Thiết kế dữ liệu.** 9](#_Toc469300745)

[**2.3.** **Các modun chính của hệ thống.** 10](#_Toc469300746)

[**3.** **Kết quả tính toán.** 12](#_Toc469300747)

[**3.1.** **Xây dựng bộ dữ liệu test thử chương trình** 12](#_Toc469300748)

[**3.2.** **Thống kê thời gian tính của chương trình theo kích thước dữ liệu** 13](#_Toc469300749)

[**4.** **Hướng dẫn sử dụng chương trình.** 14](#_Toc469300750)

[**5.** **Chương trình nguồn** 15](#_Toc469300751)

[**6.** **Tài liệu tham khảo.** 34](#_Toc469300752)

1. **Các vấn đề lý thuyết**.
   1. **Mô tả bài toán.**

Cho đồ thị *G* = (*V*, *E*) là đồ thị vô hướng liên thông với tập đỉnh.

*V* = {1, 2, 3,…, *n*} và tập cạnh E gồm m cạnh. Mỗi cạnh e của đồ thị *G* được gán với 1 số thực *c(e)*, gọi là độ dài của nó.

Giả sử *H* = (*V*, *T*) là cây khung của đồ thị *G*.

Ta gọi độ dài *c*(*H*) của cây khung H là tổng độ dài các cạnh của nó:

*c*(*H*) =

Vấn đề đặt ra là trong số tất cả các cây khung của đồ thị *G* hãy tìm cây khung với độ dài nhỏ nhất.

Cây khung như vậy được gọi là cây khung nhỏ nhất của đồ thị và bài toán đặt ra được gọi là bài toán cây khung nhỏ nhất.

* 1. **Thuật toán Kruskal.**

Thuật toán sẽ xây dựng tập cạnh T của cây khung nhỏ nhất *H* = (*V*, *T*) theo từng bước. Trước hết sắp xếp các cạnh của đồ thị *G* theo thứ tự không giảm của độ dài. Bắt đầu từ tập *T* = , ở mỗi bước ta sẽ lần lượt duyệt trong danh sách cạnh đã sắp xếp, từ cạnh có độ dài nhỏ đến cạnh có độ dài lớn hơn, để tìm ra cạnh mà việc bổ sung nó vào tập *T* không tạo thành chu trình trong tập này. Thuật toán sẽ kết thúc khi ta thu được tập *T* gồm *n*-1 cạnh. Cụ thể thuật toán có thể mô tả như sau:

Mã giả của thuật toán Kruskal:

**begin**

***T* := ;**

**while ( |*T*| < *n* -1) and (*E* ) do**

**begin**

**Chọn *e* là cạnh có độ dài nhỏ nhất trong *E*;**

***E* := *E* \ {*e*};**

**if ( *T* {*e*} không chứa chu trình) then *T* := *T*  {*e*};**

**end;**

**end;**

* 1. **Cấu trúc dữ liệu các tập không giao nhau.**

Một cấu trúc dữ liệu các tập không giao nhau chứa một tập hợp

*S* = { } của các tập phân biệt. Mỗi tập được xác định bởi một đại diện, là một phần tử nào đó trong tập.

Gọi *x* là một phần tử bất kì.

Cấu trúc dữ liệu các tập không giao nhau hỗ trợ các thao tác sau:

* MAKE-SET(*x*): tạo ra một tập mới chỉ có một phần tử duy nhất là *x*.
* UNION(*x*, *y*): hợp nhất hai tập chứa *x* và *y* là  và vào một tập mới. Cần giả sử rằng 2 tập này đã phân biệt trước đó. Đại diện của tập mới này có thể là bất kì phần tử nào trong và Trong các bài toán thực tế ta thường chỉ định đại diện của hoặc làm đại diện của tập mới.
* FIND\_SET(*x*): trả về một con trỏ đại diện (duy nhất) của tập chứa phần tử *x*.

Gọi n là số lượng các phép MAKE-SET.

Và m là tổng số các phép MAKE-SET, FIND-SET và UNION.

Giả sử rằng n phép MAKE-SET được thực hiện trước tất cả các phép khác.

Vì các tập là phân biệt nên mỗi phép UNION sẽ làm giảm số lượng tập đi một. Vì thế sau nhiều nhất *n*- 1 phép UNION, chỉ còn lại duy nhất một tập.

* 1. **Triển khai rừng các tập không giao nhau.**

Đây là một cách triển khai của cấu trúc các tập không giao nhau. Ta đại diện mỗi tập bởi một cây, mỗi node trên cây chỉ chứa đúng một phần tử của tập. Mỗi node có một con trỏ trỏ tới cha của nó. Gốc của cây chứa một đại diện duy nhất của cây, và nó là cha của chính nó.

Các thao tác trong cấu trúc dữ liệu các tập không giao nhau được viết lại như sau:

* MAKE-SET: tạo ra một cây với chỉ một node.
* FIND-SET: trả về phần tử đại diện của tập, bằng cách đi theo con trỏ cha cho đến khi tìm được node gốc của cây. Các node được thăm trên đường hướng về gốc tạo thành “con đường tìm kiếm” (find-path).
* UNION: hợp nhất hai cây thành một, bằng cách cho node gốc của cây này trỏ tới node gốc của cây kia.

Ta sẽ áp dụng hai kĩ thuật Heuristics để cải thiện thời gian chạy:

Đầu tiên là UNION-BY-RANK: ý tưởng là làm gốc của cây có số node ít hơn trỏ tới gốc của cây có số node nhiều hơn. Ở mỗi node ta chứa một hạng (rank) là giới hạn trên cho chiều cao của node đó.

Thứ hai là PATH-COMPRESSION: kỹ thuật này được sử dụng xuyên suốt trong quá trình FIND-SET để làm cho mỗi node trên find-path trỏ trực tiếp tới node gốc. PATH-COMPRESSION không làm thay đổi rank của bất kì node nào.

Để triển khai kĩ thuật UNION-BY-RANK ta phải theo dõi rank của các node. Với mỗi node *x* ta lưu một giá trị nguyên rank[*x*] là giới hạn trên chiều cao của *x*.

Khi một tập đơn được tạo bởi MAKE-SET, node duy nhất trong tập này được khởi tạo với rank[*x*] = 0. Mỗi thao tác FIND-SET không làm thay đổi rank của bất kì node nào.

Khi áp dụng UNION-BY-RANK với hai cây sẽ có hai trường hợp xảy ra, tùy thuộc vào rank của hai gốc ở hai cây.

Nếu các gốc có rank khác nhau ta sẽ để gốc có rank cao hơn làm cha của gốc có rank thấp hơn, nhưng rank của bản thân mỗi gốc không thay đổi.

Nếu hai gốc có rank bằng nhau, ta sẽ chọn ngẫu nhiên một gốc làm cha đồng thời tăng rank của gốc cha này lên một.

Gọi p[*x*] là cha của *x*.

Thủ tục LINK là thủ tục con được gọi bởi UNION-BY-RANK nhận con trỏ tới hai gốc làm tham số đầu vào.

Mã giả cho triển khai rừng các tập không giao nhau:

**MAKE-SET(*x*)  
 1. *p[x] ← x*  
 2. *rank[x]* ← 0**

**UNION-BY-RANK(*x*, *y*)  
 LINK ( FIND-SET(*x*), FIND-SET(*y*) )**

**LINK( *x*, *y* )  
 1. if *rank[x]* > *rank[y]*  
 2. then *p[y] ← x*  
 3. else *p[x] ← y*  
 4. if *rank[x] = rank[y]***

**5. then *rank[y] ← rank[y]* + 1**

**FIND-SET( *x*)**

1. **if *x p[x]***
2. **then *p[x]* ← FIND-SET( *p[x]* )**
3. **return *p[x]***

Mỗi lần gọi FIND-SET(*x*) sẽ trả về *p[x]* ở dòng 3.

Nếu *x* là gốc, dòng 2 sẽ không được thực hiện và trả về *x = p[x].*

Nếu *x* không phải là gốc, dòng 2 sẽ được thực hiện và gọi đệ quy với tham số đầu vào là *p[x]* sẽ trả về một con trỏ tới gốc. Dòng 2 cập nhật node *x* để trỏ trực tiếp tới gốc và con trỏ này được trả về ở dòng 3.

* 1. **Thuật toán Kruskal sử dụng cấu trúc dữ liệu các tập không giao nhau.**

Sử dụng cấu trúc dữ liệu các tập không giao nhau để lưu trữ các tập phân biệt của các thành phần trong bài toán.

Mỗi tập chứa các đỉnh trong một cây của rừng hiện tại.

Thao tác FIND-SET(*u*) sẽ trả về phần tử đại diện của tập chứa *u*.

Vì thế ta có thể kiểm tra được hai đỉnh *u* và *v* có thuộc về cùng một cây hay không bằng cách so sánh FIND-SET(*u*) và FIND-SET(*v*)

Các cây được hợp nhất lại với nhau bằng việc sử dụng UNION.

**MST-KRUSKAL (*G*, *w*)  
1. *A* ← Ø  
2. for each vertex *v* ∈ *V[G]*  
3. do MAKE-SET(*v*)  
4. sort the edges of *E* into nondecreasing order by weight *w*  
5. for each edge (*u*, *v*) ∈ *E*, taken in nondecreasing order by weight  
6. do if FIND-SET(*u*) ≠ FIND-SET(*v*)**

**7. then *A* ← *A* ∪ {(*u*, *v*)}**

**8. UNION(*u*, *v*)**

**9. return *A***

Từ dòng 1 🡪 3 khởi tạo một tập rỗng *A* và tạo ra |*V*| cây, mỗi cây chứa một đỉnh.

Dòng 4, các cạnh thuộc *E* được sắp xếp theo thứ tự không giảm của trọng số.

Vòng lặp từ dòng 5 🡪 8 kiểm tra với mỗi cạnh (*u*, *v*) liệu hai đỉnh *u*, *v* có thuộc về cùng một cây hay không.

Nếu đúng, cạnh (*u*, *v*) sẽ không được thêm vào cây khung nhỏ nhất vì tạo ra chu trình và nó bị loại.

Ngược lại nếu hai đỉnh *u*, *v* thuộc hai cây khác nhau, cạnh (*u*, *v*) sẽ được thêm vào cây khung *A* ở dòng 7 và hai cây chứa hai đỉnh *u*, *v* sẽ được hợp nhất làm một.

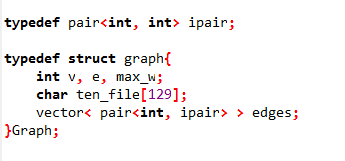
1. **Hệ thống chương trình giải bài toán.**
   1. **Mô tả thiết kế chức năng.**

Chương trình có 4 chức năng chính:

* Nhập dữ liệu
* Từ bàn phím
* Từ file có sẵn
* Tạo ngẫu nhiên
* Hiển thị dữ liệu đã nhập
* Hiển thị ra màn hình
* Ghi ra file
* Hiển thị kết quả
* Hiển thị ra màn hình
* Ghi ra file
* Hiển thị các bước trung gian
* Xem các bước trung gian
* Không xem các bước trung gian
  1. **Thiết kế dữ liệu.**

Cần biểu diễn đồ thị (graph), các tập không giao nhau (disjointsets) bởi các kiểu dữ liệu phù hợp.

* Graph được biểu diễn bởi 1cấu trúc struct

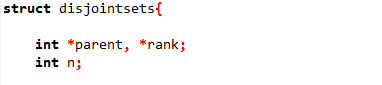


Gồm các thuộc tính:

* + - * Số lượng đỉnh, cạnh (v, e).
      * Trọng số tối đa trên tập cạnh, được dùng khi tạo ngẫu nhiên (max\_w).
      * Tên file khi lưu đồ thị, tối đa 128 kí tự (ten\_file).
      * Tập cạnh của đồ thị biểu diễn bởi 1 cấu trúc vector, trong đó mỗi cạnh được biểu diễn bởi 1 pair.

Trong một cạnh bao gồm:

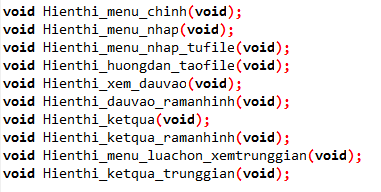
* + - * + 1 cặp ipair <int, int> chứa 2 đỉnh của cạnh.
        + 1 biến int chứa trọng số cạnh đó.
* Disjointsets được biểu diễn bởi 1cấu trúc struct



Gồm các thuộc tính:

* + Mảng cha (parent) chứa các phần tử cha tương ứng của các phần tử trong tập.
  + Mảng rank (rank) chứa hạng tương ứng của các phần tử trong tập.
  + Số lượng phần tử của tập(n).
  1. **Các modun chính của hệ thống.**Hệ thống gồm các modun phục vụ chức năng hiển thị, xử lý yêu cầu người dùng, modun giải quyết bài toán, và các modun liên quan khác.

Các modun hiển thị:



Các modun này đa phần hiển thị các menu chức năng cho người dùng lựa chọn (giống như mô tả bởi tên của nó).

Ngoại lệ:

1. Hienthi\_dauvao\_ramanhinh(void).

Đọc các thông tin đã lưu trong đồ thị g, hiển thị ra màn hình.

2. Hienthi\_ketqua\_ramanhinh(void).

Gọi tới modun giải quyết bài toán KruskalMST(void).

3. Hienthi\_ketqua\_trunggian(void).

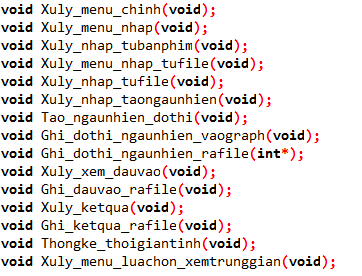
Dựa theo modun giải quyết bài toán KruskalMST(void),

hiển thị chi tiết từng bước tính toán ra màn hình.

4. Hienthi\_huongdan\_taofile(void).

Hiển thị hướng dẫn tạo file đầu vào chuẩn cho chương trình.

Các modun xử lý:



1. Xuly\_menu\_chinh(void).

Xử lý lựa chọn của người dùng chọn “Thoát” hay “Nhập dữ liệu”. Nếu là “Nhập dữ liệu”, gọi tới modun 2.

1. Xuly\_menu\_nhap(void).

Xử lý lựa chọn của người dùng chọn “Nhập từ bàn phím” , “Nhập từ file” hay “Tạo ngẫu nhiên” tương ứng với các modun 3, 4, 6.

1. Xuly\_nhap\_tubanphim(void).

Cho người dùng nhập số đỉnh, số cạnh của đồ thị.

Cho nhập từng cạnh gồm đỉnh đầu, cuối, trọng số trên cạnh.

Tất cả thông tin trên sẽ được lưu vào đồ thị g.

1. Xuly\_menu\_nhap\_tufile(void).

Xử lý lựa chọn của người dùng chọn “Xem hướng dẫn tạo file” trước hay chọn “Nhập từ file”.

1. Xuly\_nhap\_tufile(void).

Các thông tin tương tự như đọc dữ liệu từ bàn phím nhưng được lấy từ file do người dùng cung cấp.

Lưu tất cả thông tin vào đồ thị g.

1. Xuly\_nhap\_taongaunhien(void).

Cho phép người dùng nhập thông tin đồ thị muốn tạo.

Lưu lại thông tin đó, sau đó lần lượt gọi tới 2 modun:

Tao\_ngaunhien\_dothi(void) .

Ghi\_dothi\_ngaunhien\_vaograph(void).

1. Tao\_ngaunhien\_dothi(void).

Dựa vào thông tin người dùng cung cấp và không gian bộ nhớ của máy tính, tiến hành xây dựng đồ thị.

1. Ghi\_dothi\_ngaunhien\_vaograph(void).

Nếu tạo thành công đồ thị từ modun 7, ghi vào đồ thị g.

1. Ghi\_dothi\_ngaunhien\_rafile(void).

Ghi đồ thị ngẫu nhiên vừa tạo ra file.

1. Xuly\_xem\_dauvao(void).

Xử lý lựa chọn của người dùng “Hiển thị dữ liệu đầu vào ra màn hình”, “Ghi dữ liệu đầu vào ra file" hay bỏ qua.

1. Ghi\_dauvao\_rafile(void).

Đọc từ đồ thị g và ghi thông tin đồ thị ra file.

1. Xuly\_ketqua(void).

Xử lý lựa chọn của người dùng chọn “Hiển thị kết quả ra màn hình” hay “Ghi kết quả ra file".

1. Ghi\_ketqua\_rafile(void).

Gọi tới modun giải quyết bài toán KruskalMST\_file(char\*).

1. Thongke\_thoigiantinh(void)

Thống kê thời gian tính toán của giải thuật

1. Xuly\_menu\_luachon\_xemtrunggian(void).

Xử lý lựa chọn của người dùng chọn “Hiển thị kết quả trung gian” hay không.

Các modun giải quyết bài toán:



1. addedge(Graph\*, int, int, int).

Thêm cạnh vào đồ thị gồm thông tin đỉnh đầu, cuối, trọng số

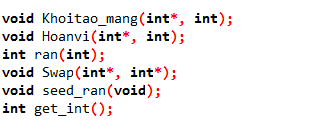
1. KruskalMST(void).

Thực hiện các bước tính toán, xử lý bài toán, trả về độ dài của cây khung nhỏ nhất của đồ thị.

1. KruskalMST\_file(void).

Tương tự như modun 2 nhưng ghi kết quả ra file.

Các modun khác liên quan:



1. Khoitao\_mang( int\* a, int n).

Tạo mảng gồm n phần tử, giá trị mỗi phần tử bằng chỉ số của nó.

1. ran(int k).

Trả về 1 số ngẫu nhiên trong khoảng 0 🡪 k-1.

1. seed\_ran(void).

Làm cho ran(int) trả về giá trị khác nhau sau mỗi lần gọi.

1. Swap( int\* a, int\* b).

Đổi chỗ 2 phần tử a, b (trong mảng).

1. Hoanvi( int\* a, int n).

Đổi chỗ lần lượt từng phần tử của mảng với 1 phần tử nào đó đứng sau nó trong mảng, thực hiện các phép đổi chỗ này n lần.

1. get\_int(void).

Trả về kí tự số mà người dùng nhập vào từ bàn phím.

1. **Kết quả tính toán.**
   1. **Xây dựng bộ dữ liệu test thử chương trình.**

Bộ dữ liệu test được sinh bởi chương trình “**A Graph Generation Package**” với chức năng sinh“Random Connected Graph”.

Mỗi file test chứa thông tin của n đỉnh và m cạnh, m = (n-1) \* n / 2

Bộ test gồm các loại đồ thị với tỷ số giữa số cạnh(m) và số đỉnh(n) khác nhau:

* Đồ thị thưa: m = 2n, 4n, 8n, 10n
* Đồ thị dày: m =

Số đỉnh n thử với các giá trị khác nhau: n = 1000, 5000, 10000, 20000, …

Với n = 10000, kích thước file dữ liệu test của đồ thị dày lên tới hơn 1GB.

Với n = 20000, vẫn có thể sinh được file dữ liệu test của đồ thị dày tuy nhiên thời gian sinh quá lâu (gần 200 triệu cạnh).

Với n = 50000 trở đi, không thể sinh được bất kì đồ thị thưa nào.

Tên của file dữ liệu test có dạng “sodinh\_socanh”.

VD: n = 5000, m = 25000000, tên file là “5k25m”.

* 1. **Thống kê thời gian tính của chương trình theo kích thước dữ liệu.**

Cấu hình máy sử dụng để thực nghiệm:

CPU: Intel Core i5-5200U 2.2 GHz (3 MB L3 Cache, up to 2.7 GHz).

RAM: 6 GB (DDR3 Bus 1600 MHz).

Hệ điều hành: Window 10 Education

Compiler: Dev C++ 5.11

Mỗi trường hợp được test 10 lần, kết quả trung bình được ghi trong bảng sau:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| m n | 1000 | 5000 | 10000 | 20000 |
| m = 2n | 0.0015 (s) | 0.0016 (s) | 0.0078 (s) | 0.0204 (s) |
| m = 4n | 0.0015 (s) | 0.0093 (s) | 0.0189 (s) | 0.0422 (s) |
| m = 8n | 0.0015 (s) | 0.0203 (s) | 0.0469 (s) | 0.0889 (s) |
| m = 10n | 0.0031 (s) | 0.0233(s) | 0.0580 (s) | 0.1139 (s) |
| m = | 0.3140 (s) | 9.4470(s) | 41.0019 (s) | Rất lâu |

1. **Hướng dẫn sử dụng chương trình.**
2. Sau khi chạy chương trình, màn hình console xuất hiện menu nhập.
3. Người dùng chọn phương thức nhập phù hợp trong 3 chức năng:

* Nhập từ bàn phím:
* Yêu cầu nhập vào số đỉnh, số cạnh của đồ thị.
* Sau đó nhập chi tiết đỉnh đầu, đỉnh cuối và trọng số trên từng cạnh.
* Nhập từ file có sẵn:
* Người dùng xem hướng dẫn tạo file dữ liệu chuẩn.
* Yêu cầu nhập đúng tên file dữ liệu.
* Tạo ngẫu nhiên đồ thị:

Yêu cầu nhập vào số đỉnh, số cạnh của đồ thị và trọng số tối đa của cạnh.

1. Sau đó, người dùng có thể chọn chức năng ghi dữ liệu vừa nhập ra file, hiển thị ra màn hình hoặc bỏ qua.
2. Tiếp theo, người dùng có thể chọn chức năng ghi kết quả ra file, hiển thị kết quả ra màn hình.
3. Cuối cùng, người dùng có thể chọn chức năng xem hoặc không xem các bước tính toán trung gian.
4. **Chương trình nguồn.**

**#include<iostream>**

**#include<stdio.h>**

**#include<conio.h>**

**#include<iostream>**

**#include<vector>**

**#include<algorithm>**

**#include<windows.h>**

**#include <time.h>**

**#include<stdlib.h>**

**using namespace std;**

**typedef pair<int, int> ipair;**

**typedef struct graph{**

**int v, e, max\_w;**

**char ten\_file[129];**

**vector< pair<int, ipair> > edges;**

**}Graph;**

**struct disjointsets{**

**int \*parent, \*rank;**

**int n;**

**disjointsets(int n){**

**this->n = n;**

**parent = new int[n + 1];**

**rank = new int[n + 1];**

**for (int i = 0; i <= n; i++){**

**rank[i] = 0;**

**parent[i] = i;**

**}**

**}**

**int find\_set(int u){**

**if (u != parent[u])**

**parent[u] = find\_set(parent[u]);**

**return parent[u];**

**}**

**void unions(int x, int y){**

**if (rank[x] > rank[y])**

**parent[y] = x;**

**else{**

**parent[x] = y;**

**}**

**if (rank[x] == rank[y]){**

**rank[y]++;**

**}**

**}**

**};**

**void Hienthi\_menu\_chinh(void);**

**void Hienthi\_menu\_nhap(void);**

**void Hienthi\_menu\_nhap\_tufile(void);**

**void Hienthi\_huongdan\_taofile(void);**

**void Hienthi\_xem\_dauvao(void);**

**void Hienthi\_dauvao\_ramanhinh(void);**

**void Hienthi\_ketqua(void);**

**void Hienthi\_ketqua\_ramanhinh(void);**

**void Hienthi\_menu\_luachon\_xemtrunggian(void);**

**void Hienthi\_ketqua\_trunggian(void);**

**void Xuly\_menu\_chinh(void);**

**void Xuly\_menu\_nhap(void);**

**void Xuly\_nhap\_tubanphim(void);**

**void Xuly\_menu\_nhap\_tufile(void);**

**void Xuly\_nhap\_tufile(void);**

**void Xuly\_nhap\_taongaunhien(void);**

**void Tao\_ngaunhien\_dothi(void);**

**void Ghi\_dothi\_ngaunhien\_vaograph(void);**

**void Ghi\_dothi\_ngaunhien\_rafile(int\*);**

**void Xuly\_xem\_dauvao(void);**

**void Ghi\_dauvao\_rafile(void);**

**void Xuly\_ketqua(void);**

**void Ghi\_ketqua\_rafile(void);**

**void Thongke\_thoigiantinh(void);**

**void Xuly\_menu\_luachon\_xemtrunggian(void);**

**void Khoitao\_mang(int\*, int);**

**void Hoanvi(int\*, int);**

**int ran(int);**

**void Swap(int\*, int\*);**

**void seed\_ran(void);**

**int get\_int();**

**void addedge(Graph\*, int, int, int);**

**int KruskalMST(void);**

**int KruskalMST\_file(char\*);**

**static Graph graph, graphBanDau;**

**static Graph \*g = &graph;**

**static Graph \*gBanDau = &graphBanDau;**

**clock\_t begin, end;**

**char tenfile\_thongke[129];**

**int main(){**

**seed\_ran();**

**Hienthi\_menu\_chinh();**

**Xuly\_menu\_chinh();**

**Hienthi\_xem\_dauvao();**

**Xuly\_xem\_dauvao();**

**Hienthi\_ketqua();**

**Xuly\_ketqua();**

**Hienthi\_menu\_luachon\_xemtrunggian();**

**Xuly\_menu\_luachon\_xemtrunggian();**

**return 0;**

**}**

**void Hienthi\_menu\_chinh(void){**

**printf("\n 0. Thoat chuong trinh");**

**printf("\n 1. Nhap du lieu");**

**printf("\n Ban chon? ");**

**}**

**void Xuly\_menu\_chinh(void){**

**while(1){**

**int choice = get\_int();**

**if(choice == 0){**

**exit(0);**

**}**

**else if(choice == 1){**

**Hienthi\_menu\_nhap();**

**Xuly\_menu\_nhap();**

**break;**

**}**

**else{**

**printf( " --- Nhap vao loi. Vui long nhap lai! --- \n " );**

**}**

**}**

**}**

**void Hienthi\_menu\_nhap(void){**

**system("cls");**

**printf("\n 1. Tu ban phim");**

**printf("\n 2. Tu file");**

**printf("\n 3. Tao ngau nhien");**

**printf("\n Ban chon? ");**

**}**

**void Xuly\_menu\_nhap(void){**

**while(1){**

**int choice = get\_int();**

**if(choice == 1){**

**Xuly\_nhap\_tubanphim();**

**break;**

**}**

**else if(choice == 2){**

**Hienthi\_menu\_nhap\_tufile();**

**Xuly\_menu\_nhap\_tufile();**

**break;**

**}**

**else if(choice == 3){**

**Xuly\_nhap\_taongaunhien();**

**break;**

**}**

**else{**

**printf( " --- Nhap vao loi. Vui long nhap lai! --- \n " );**

**}**

**}**

**}**

**void Xuly\_nhap\_tubanphim(void){**

**system("cls");**

**int a, b, w;**

**while(1){**

**printf("\n Nhap so dinh: ");**

**g->v = get\_int();**

**gBanDau->v = g->v;**

**printf(" Nhap so canh: ");**

**g->e = get\_int();**

**gBanDau->e = g->e;**

**if(g->v <= 0 || g->e <= 0){**

**printf(" Luu y nhap so luong dinh hoac canh duong!\n\n");**

**system("cls");**

**continue;**

**}**

**break;**

**}**

**printf("\n Nhap vao canh va trong so tren canh: \n");**

**if(g->e < (g->v - 1)){**

**printf(" Luu y nhap toi thieu %d canh!\n\n", g->v - 1);**

**g->e = g->v - 1;**

**}else if(g->e > (g->v)\*(g->v - 1)/2){**

**printf(" Luu y nhap toi da %d canh!\n\n", (g->v)\*(g->v - 1)/2);**

**g->e = (g->v)\*(g->v - 1)/2;**

**}**

**int mangCanh[g->e + 1][g->e + 1];**

**for(int i = 0; i < g->e + 1; i++)**

**for(int j = 0; j < g->e + 1; j++){**

**mangCanh[i][j] = 0;**

**}**

**for (int i = 0; i < g->e; i++) {**

**a = get\_int();**

**b = get\_int();**

**if(a == b){**

**printf(" Luu y khong nhap canh vong!\n\n");**

**i = i - 1;**

**continue;**

**}**

**if(a > g->v || a <= 0 || b > g->v || b <= 0){**

**printf(" Luu y nhap vao dinh trong pham vi tu 1 den %d !\n\n", g->v);**

**i = i - 1;**

**continue;**

**}**

**if(mangCanh[a][b] != 0){**

**printf(" Canh nay da duoc nhap truoc do. Vui long nhap lai!\n\n");**

**i = i - 1;**

**continue;**

**}**

**w = get\_int();**

**if(w <= 0){**

**printf(" Luu y nhap trong so duong!\n\n");**

**i = i - 1;**

**continue;**

**}**

**mangCanh[a][b] = w;**

**mangCanh[b][a] = w;**

**printf("\n");**

**addedge(g, a, b, w);**

**addedge(gBanDau, a, b, w);**

**}**

**}**

**void Hienthi\_menu\_nhap\_tufile(void){**

**system("cls");**

**printf("\n 1. Huong dan tao file du lieu chuan");**

**printf("\n 2. Nhap tu file");**

**printf("\n Ban chon? ");**

**}**

**void Xuly\_menu\_nhap\_tufile(void){**

**while(1){**

**int choice = get\_int();**

**if(choice == 1){**

**Hienthi\_huongdan\_taofile();**

**Xuly\_nhap\_tufile();**

**break;**

**}**

**else if(choice == 2){**

**Xuly\_nhap\_tufile();**

**break;**

**}**

**else{**

**printf( " --- Nhap vao loi. Vui long nhap lai! --- \n " );**

**}**

**}**

**}**

**void Hienthi\_huongdan\_taofile(void){**

**system("cls");**

**printf("\n\n--------------------------------------------HUONG DAN TAO FILE DU LIEU CHUAN--------------------------------------------\n\n");**

**printf(" A. Quy uoc.\n");**

**printf(" 1. Hai so dau tien trong file la so dinh va so canh cua do thi.\n");**

**printf(" 2. Tu do tro di moi bo ba so lien nhau chinh la thong tin ve mot canh.\n");**

**printf(" Hai so dau tien la hai dinh, so thu 3 la trong so cua canh\n\n\n");**

**printf(" B. Luu y.\n");**

**printf(" 1. File chi duoc chua du lieu so va khoang trang.\n");**

**printf(" 2. Tat ca du lieu so trong file phai la so duong.\n");**

**printf(" 3. Cac dinh nhap vao phai nam trong gioi han dinh cho phep.\n");**

**printf(" 4. Do thi khong duoc chua canh vong (hai dinh trung nhau).\n");**

**printf(" 5. Do thi khong duoc chua cac canh lap (VD: 2 - 3 va 3 - 2).\n");**

**printf(" ...\n\n\n");**

**printf(" Neu ban khong dam bao duoc file du lieu chuan, ban co the su dung chuc nang \"Tao ngau nhien\".");**

**printf("\n\n\n\n Nhan phim bat ki de tiep tuc ...");**

**getch();**

**}**

**void Xuly\_nhap\_tufile(void){**

**system("cls");**

**char ten\_file[129];**

**FILE \*fp;**

**int a, b, w;**

**while(1){**

**printf("\n\n Nhap vao ten file de doc du lieu: ");**

**scanf("%s", ten\_file);**

**if((fp = fopen(ten\_file, "rt")) == NULL){**

**printf("\n File khong ton tai!\n Vui long nhap lai!");**

**}**

**else{**

**printf("\n Dang doc du lieu tu file %s ...\n\n", ten\_file);**

**break;**

**}**

**}**

**fscanf(fp, "%d", &(g->v));**

**fscanf(fp, "%d", &(g->e));**

**gBanDau->v = g->v;**

**gBanDau->e = g->e;**

**for (int i = 0; i < g->e; i++) {**

**fscanf(fp, "%d", &a);**

**fscanf(fp, "%d", &b);**

**fscanf(fp, "%d", &w);**

**if(g->v <= 0 || g->e <= 0 || w <= 0 || a > g->v || a <= 0 || b > g->v || b <= 0 || a == b){**

**printf(" File du lieu xuat hien loi bat thuong!\n");**

**printf(" Vui long xem muc \"Huong dan tao file du lieu dau vao chuan\".");**

**printf(" \n\n\n\n\nNhan phim bat ki de thoat chuong trinh...");**

**getch();**

**exit(1);**

**}**

**addedge(g, a, b, w);**

**addedge(gBanDau, a, b, w);**

**}**

**strcpy(tenfile\_thongke, ten\_file);**

**fclose(fp);**

**}**

**void Xuly\_nhap\_taongaunhien(void){**

**system("cls");**

**printf("\n Nhap so dinh: ");**

**g->v = get\_int();**

**printf("\n Nhap so canh: ");**

**g->e = get\_int();**

**printf("\n Nhap trong so toi da cua canh: ");**

**g->max\_w = get\_int();**

**printf("\n\n Nhap ten file luu do thi: ");**

**scanf("%s", g->ten\_file);**

**strcpy(tenfile\_thongke, g->ten\_file);**

**Tao\_ngaunhien\_dothi();**

**Ghi\_dothi\_ngaunhien\_vaograph();**

**}**

**void Tao\_ngaunhien\_dothi(){**

**int i, j, count, index, \*adj\_matrix, \*tree;**

**if ( ( adj\_matrix = ( int \* ) calloc( g->v \* g->v, sizeof( int ) ) ) == NULL ) {**

**printf( " Khong du khong gian cho do thi!\n" );**

**printf("\n\n\n\n Nhan phim bat ki de thoat chuong trinh ...");**

**getch();**

**exit(1);**

**return;**

**}**

**if ( ( tree = ( int \* ) calloc( g->v, sizeof( int ) ) ) == NULL ) {**

**printf( " Khong du khong gian cho do thi!\n" );**

**printf("\n\n\n\n Nhan phim bat ki de thoat chuong trinh ...");**

**getch();**

**exit(1);**

**return;**

**}**

**printf( "\n\t Dang bat dau xay dung do thi ..." );**

**Khoitao\_mang( tree, g->v );**

**Hoanvi( tree, g->v );**

**for ( i = 1; i < g->v; i++ ) {**

**j = ran( i );**

**//Khoi tao dung v-1 canh, va chung tao nen do thi lien thong, cac canh khac trong mang chua khoi tao**

**adj\_matrix[ tree[ i ] \* g->v + tree[ j ] ] = adj\_matrix[ tree[ j ] \* g->v + tree[ i ] ] = 1 + ran( g->max\_w ) ;**

**}**

**int soCanh = min(max(g->e, g->v-1), (g->v)\*(g->v-1)/2);**

**g->e = soCanh;**

**for ( count = g->v - 1; count < soCanh; ) {**

**i = ran( g->v );**

**j = ran( g->v );**

**if ( i == j )**

**continue;**

**if ( i > j )**

**Swap( &i, &j );**

**index = i \* g->v + j;**

**if ( !adj\_matrix[ index ] ) {**

**adj\_matrix[ index ] = 1 + ran( g->max\_w );**

**count++;**

**}**

**}**

**Ghi\_dothi\_ngaunhien\_rafile(adj\_matrix);**

**free( tree );**

**free( adj\_matrix );**

**}**

**void Ghi\_dothi\_ngaunhien\_vaograph(){**

**FILE \*fp;**

**int a, b, w;**

**fp = fopen(g->ten\_file, "rt");**

**fscanf(fp, "%d", &(g->v));**

**fscanf(fp, "%d", &(g->e));**

**gBanDau->v = g->v;**

**gBanDau->e = g->e;**

**for (int i = 0; i < g->e; i++) {**

**fscanf(fp, "%d", &a);**

**fscanf(fp, "%d", &b);**

**fscanf(fp, "%d", &w);**

**addedge(g, a, b, w);**

**addedge(gBanDau, a, b, w);**

**}**

**fclose(fp);**

**}**

**void Khoitao\_mang( int\* a, int end ){**

**int i;**

**for ( i = 0; i < end; i++ )**

**\*a++ = i;**

**}**

**void Hoanvi( int\* a, int n ){**

**int i;**

**for ( i = 0; i < n - 1; i++ )**

**Swap( a + i + ran( n - i ), a + i );**

**}**

**void Swap( int\* a, int \*b ){**

**int temp;**

**temp = \*a;**

**\*a = \*b;**

**\*b = temp;**

**}**

**int ran( int k ){**

**return rand() % k;**

**}**

**void seed\_ran( void ){**

**srand( ( unsigned short ) time( NULL ) );**

**}**

**void Ghi\_dothi\_ngaunhien\_rafile(int\* adj\_matrix){**

**int i, j, index;**

**FILE \*fp;**

**if ( ( fp = fopen( g->ten\_file, "w" ) ) == NULL ) {**

**printf( " Khong the mo file \"%s\" de ghi!\n", g->ten\_file );**

**return;**

**}**

**fprintf( fp, "%-8d %-8d\n", g->v, g->e );**

**for ( i = 1; i < g->v; i++ )**

**for ( j = i + 1; j <= g->v; j++ ) {**

**index = ( i - 1 ) \* g->v + j - 1;**

**if ( adj\_matrix[ index ] )**

**fprintf( fp, "%-8d %-8d %-8d\n", i, j, adj\_matrix[ index ] );**

**}**

**fclose( fp );**

**printf( "\n\t Do thi da duoc ghi vao file \"%s\".\n", g->ten\_file );**

**printf("\n\n\n\n Nhan phim bat ki de tiep tuc ...");**

**getch();**

**}**

**void Hienthi\_xem\_dauvao(void){**

**system("cls");**

**printf("\n 1. Hien thi du lieu dau vao ra man hinh");**

**printf("\n 2. Ghi du lieu dau vao ra file");**

**printf("\n 3. Tiep tuc");**

**printf("\n Ban chon? ");**

**}**

**void Xuly\_xem\_dauvao(void){**

**while(1){**

**int choice = get\_int();**

**if(choice == 1){**

**Hienthi\_dauvao\_ramanhinh();**

**break;**

**}**

**else if(choice ==2){**

**Ghi\_dauvao\_rafile();**

**break;**

**}**

**else if(choice == 3){**

**break;**

**}**

**else{**

**printf( " --- Nhap vao loi. Vui long nhap lai! --- \n " );**

**}**

**}**

**}**

**void Hienthi\_dauvao\_ramanhinh(void){**

**system("cls");**

**printf("\n %-8d%-8d", g->v, g->e);**

**vector< pair<int, ipair> >::iterator it;**

**for (it = g->edges.begin(); it != g->edges.end(); it++){**

**int u = it->second.first;**

**int v = it->second.second;**

**int w = it->first;**

**printf("\n %-8d%-8d%-8d", u, v, w);**

**}**

**printf("\n\n\n\n Nhan phim bat ki de tiep tuc ...");**

**getch();**

**}**

**void Ghi\_dauvao\_rafile(){**

**char ten\_file[129];**

**FILE \*fp;**

**printf("\n Nhap ten file de ghi du lieu dau vao: ");**

**scanf("%s", ten\_file);**

**if ( ( fp = fopen(ten\_file, "wt" ) ) == NULL ) {**

**printf( "Khong the mo file \"%s\" de ghi!\n", ten\_file);**

**return;**

**}**

**fprintf( fp, "%-8d %-8d\n", g->v, g->e );**

**vector< pair<int, ipair> >::iterator it;**

**for (it = g->edges.begin(); it != g->edges.end(); it++){**

**int u = it->second.first;**

**int v = it->second.second;**

**int w = it->first;**

**fprintf(fp, "%-8d %-8d %-8d\n", u, v, w);**

**}**

**fclose(fp);**

**printf("\n Dau vao da duoc ghi ra file \"%s\"", ten\_file);**

**printf("\n\n\n\n Nhan phim bat ki de tiep tuc ...");**

**getch();**

**}**

**void Hienthi\_ketqua(){**

**system("cls");**

**printf("\n 1. Hien thi ket qua ra man hinh");**

**printf("\n 2. Ghi ket qua ra file");**

**printf("\n 3. Thong ke thoi gian tinh");**

**printf("\n Ban chon? ");**

**}**

**void Xuly\_ketqua(void){**

**while(1){**

**int choice = get\_int();**

**if(choice == 1){**

**Hienthi\_ketqua\_ramanhinh();**

**break;**

**}**

**else if(choice == 2){**

**Ghi\_ketqua\_rafile();**

**break;**

**}**

**else if(choice == 3){**

**Thongke\_thoigiantinh();**

**break;**

**}**

**else{**

**printf( " --- Nhap vao loi. Vui long nhap lai! --- \n " );**

**}**

**}**

**}**

**void Hienthi\_ketqua\_ramanhinh(void){**

**system("cls");**

**printf("\n Cay khung cua do thi la: \n\n");**

**int mst\_wt = KruskalMST();**

**printf("\n Do dai cua cay khung nho nhat la: %d\n\n\n\nNhan phim bat ki de tiep tuc ...", mst\_wt);**

**getch();**

**}**

**void Ghi\_ketqua\_rafile(void){**

**char ten\_file\_ketqua[129];**

**printf("\n Nhap ten file ket qua: ");**

**scanf("%s", ten\_file\_ketqua);**

**KruskalMST\_file(ten\_file\_ketqua);**

**printf("\n Ket qua da duoc ghi ra file \"%s\"", ten\_file\_ketqua);**

**printf("\n\n\n\n Nhan phim bat ki de tiep tuc ...");**

**getch();**

**}**

**void Thongke\_thoigiantinh(void){**

**begin = clock();**

**int mst\_wt = 0;**

**sort(g->edges.begin(), g->edges.end());**

**disjointsets ds(g->v);**

**vector< pair<int, ipair> >::iterator it;**

**for (it = g->edges.begin(); it != g->edges.end(); it++){**

**int u = it->second.first;**

**int v = it->second.second;**

**int set\_u = ds.find\_set(u);**

**int set\_v = ds.find\_set(v);**

**if (set\_u != set\_v){**

**mst\_wt += it->first;**

**ds.unions(set\_u, set\_v);**

**}**

**}**

**end = clock();**

**double ketQua = (double) (end- begin) / CLOCKS\_PER\_SEC;**

**strcat(tenfile\_thongke, "\_ThongKe");**

**FILE \*fp;**

**fp = fopen(tenfile\_thongke, "a");**

**fprintf(fp, "%.6lf\n", ketQua);**

**fclose(fp);**

**printf("\n\n Thoi gian tinh toan: %.6lf (s)", ketQua);**

**printf("\n\n\n\n Nhan phim bat ki de tiep tuc ...");**

**getch();**

**}**

**void Hienthi\_menu\_luachon\_xemtrunggian(void){**

**system("cls");**

**printf("\n Ban co muon xem ket qua trung gian? ");**

**printf("\n 1. Co");**

**printf("\n 2. Khong");**

**printf("\n Ban chon? ");**

**}**

**void Xuly\_menu\_luachon\_xemtrunggian(void){**

**while(1){**

**int choice = get\_int();**

**if(choice == 1){**

**Hienthi\_ketqua\_trunggian();**

**break;**

**}**

**else if(choice == 2){**

**break;**

**}**

**else{**

**printf( " --- Nhap vao loi. Vui long nhap lai! --- \n " );**

**}**

**}**

**}**

**void Hienthi\_ketqua\_trunggian(void){**

**system("cls");**

**int mst\_wt = 0;**

**printf("\n Khoi tao do dai cay khung ban dau = 0\n");**

**Sleep(1000);**

**printf("\n Tap canh chua sap xep:\n");**

**vector< pair<int, ipair> >::iterator it;**

**printf("\n-------Canh-------|---Trong so---|\n");**

**for (it = gBanDau->edges.begin(); it != gBanDau->edges.end(); it++){**

**int u = it->second.first;**

**int v = it->second.second;**

**int w = it->first;**

**printf("%d\t-\t%d |\t %d\t |\n", u, v, w);**

**Sleep(1000);**

**}**

**printf("---------------------------------|\n\n\n\n");**

**Sleep(1000);**

**printf("\n Tap canh da sap xep:\n");**

**vector< pair<int, ipair> >::iterator it2;**

**printf("\n-------Canh-------|---Trong so---|\n");**

**for (it2 = g->edges.begin(); it2 != g->edges.end(); it2++){**

**int u = it2->second.first;**

**int v = it2->second.second;**

**int w = it2->first;**

**printf("%d\t-\t%d |\t %d\t |\n", u, v, w);**

**Sleep(1000);**

**}**

**printf("---------------------------------|\n");**

**Sleep(1000);**

**struct disjointsets ds(g->v);**

**int mst\_wt2 = 0;**

**int dem = 0;**

**vector< pair<int, ipair> >::iterator it3;**

**for (it3 = g->edges.begin(); it3 != g->edges.end(); it3++){**

**if(dem == g->v - 1){**

**printf("\n\nCay khung nho nhat tim duoc co do dai la: %d", mst\_wt2);**

**break;**

**}**

**int u = it3->second.first;**

**int v = it3->second.second;**

**int w = it3->first;**

**int set\_u = ds.find\_set(u);**

**int set\_v = ds.find\_set(v);**

**printf("\nXet canh %d - %d\n", u, v);**

**Sleep(1000);**

**printf("\nSet(%d) = %d", u, set\_u);**

**printf("\nSet(%d) = %d", v, set\_v);**

**Sleep(1000);**

**if(set\_u == set\_v){**

**printf("\n\n-->Loai canh %d - %d do 2 dinh cua canh nay cung thuoc 1 tap\n", u, v);**

**}else{**

**printf("\n\n-->Hop nhat 2 tap chua dinh %d va dinh %d\n", u, v);**

**if (set\_u != set\_v){**

**ds.unions(set\_u, set\_v);**

**dem++;**

**}**

**Sleep(1000);**

**printf("\n\nDo dai cay khung hien tai = %d + %d = %d", mst\_wt2, w, mst\_wt2 + w);**

**mst\_wt2 = mst\_wt2 + w;**

**}**

**printf("\n----------------------------------------------------------\n\n\n");**

**}**

**printf("\n\n\n\n Nhan phim bat ki de thoat chuong trinh ...");**

**getch();**

**}**

**int KruskalMST(){**

**int mst\_wt = 0;**

**sort(g->edges.begin(), g->edges.end());**

**disjointsets ds(g->v);**

**vector< pair<int, ipair> >::iterator it;**

**for (it = g->edges.begin(); it != g->edges.end(); it++){**

**int u = it->second.first;**

**int v = it->second.second;**

**int set\_u = ds.find\_set(u);**

**int set\_v = ds.find\_set(v);**

**if (set\_u != set\_v){**

**printf(" %-8d - %8d \n", u, v);**

**mst\_wt += it->first;**

**ds.unions(set\_u, set\_v);**

**}**

**}**

**return mst\_wt;**

**}**

**int KruskalMST\_file(char\* ten\_file\_ketqua){**

**FILE \*fp;**

**fp = fopen(ten\_file\_ketqua, "wt");**

**int mst\_wt = 0;**

**sort(g->edges.begin(), g->edges.end());**

**disjointsets ds(g->v);**

**vector< pair<int, ipair> >::iterator it;**

**for (it = g->edges.begin(); it != g->edges.end(); it++){**

**int u = it->second.first;**

**int v = it->second.second;**

**int set\_u = ds.find\_set(u);**

**int set\_v = ds.find\_set(v);**

**if (set\_u != set\_v){**

**fprintf(fp, " %-8d - %8d \n", u, v);**

**mst\_wt += it->first;**

**ds.unions(set\_u, set\_v);**

**}**

**}**

**fclose(fp);**

**fp = fopen(ten\_file\_ketqua, "a");**

**fprintf(fp, " %d", mst\_wt);**

**fclose(fp);**

**return mst\_wt;**

**}**

**int get\_int(){**

**char buff[ 30 ];**

**int val;**

**for ( ; ; ) {**

**scanf( "%s", buff );**

**if ( sscanf( buff, "%d", &val ) != 1 ){**

**printf( " --- Nhap vao loi. Vui long nhap lai! --- \n " );**

**}**

**else**

**return val;**

**}**

**}**

**void addedge(Graph \*g, int u, int v, int w){**

**g->edges.push\_back({ w,{ u, v } });**

**}**

1. **Tài liệu tham khảo.**

* Kruskal’s Minimum Spanning Tree using STL in C++

[**http://www.geeksforgeeks.org/kruskals-minimum-spanning-tree-using-stl-in-c/**](http://www.geeksforgeeks.org/kruskals-minimum-spanning-tree-using-stl-in-c/)

* **Introduction to Algorithms *Third Edition*** –

Thomas H. Cormen- Charles E. Leiserson- Ronald L. Rivest- Clifford Stein.

* **A Graph Generation Package-** Richard Johnsonbaugh and Martin Kalin- Department of Computer Science and Information Systems.