TRƯỜNG ĐẠI HỌC NHA TRANG

**KHOA CÔNG NGHỆ THÔNG TIN**



**CÀI ĐẶT THUẬT TOÁN XÉN TỈA ĐOẠN THẲNG**

**BẰNG THUẬT TOÁN SUTHERLAND HOGMAN**

**GVHD : Ths. ĐOÀN VŨ THỊNH**

**SVTH : Châu Cẩm Ly**

**MSSV : 59131356**

**Lớp : 59.CNTT-3**

Khánh Hòa, tháng 01 năm 2020

**MỤC LỤC**

[DANH MỤC HÌNH ii](#_Toc30078179)

[TÓM TẮT 1](#_Toc30078180)

[1.GIỚI THIỆU 2](#_Toc30078181)

[1.1 Thuật toán xén tỉa 2](#_Toc30078182)

[1.2 Thuật toán xén tỉa đoạn thẳng 3](#_Toc30078183)

[1.3 Thuật toán xén tỉa đa giác 6](#_Toc30078184)

[1.4. DevC/C++ và thư viện grpahics.h 8](#_Toc30078185)

[2. PHƯƠNG PHÁP 8](#_Toc30078186)

[2.1. Các bước cài đặt Dev\_C++ và thư viện graphic.h 8](#_Toc30078187)

[2.2.Nhập dữ liệu 9](#_Toc30078188)

[2.3. Thuật toán cắt xén đa giác 10](#_Toc30078189)

[2.3.1. Trường hợp cắt xén cạnh bên trái 10](#_Toc30078190)

[2.3.3.Trường hợp cắt xén của cửa sổ bên phải 13](#_Toc30078191)

[2.3.4. Trường hợp cắt xén của cửa sổ bên dưới 14](#_Toc30078192)

[3. KẾT QUẢ 15](#_Toc30078193)

[3.1. Dữ liệu nhập từ bàn phím 15](#_Toc30078194)

[3.2.Kết quả của xén tỉa đa giác 15](#_Toc30078195)

[3.2.1. Xét trường hợp cắt xén từ bên trái 15](#_Toc30078196)

[3.2.2. Kết quả đa giác sau khi cắt theo chiều từ trái→ trên→ phải→ dưới 16](#_Toc30078197)

[4. THẢO LUẬN 16](#_Toc30078198)

# DANH MỤC HÌNH

[Hình 1.1 Ví dụ minh họa của ứng dụng xén tỉa đa giác với Powerpoint 2013 2](#_Toc30074791)

[Hình 1.2. Mã vùng của 9 vùng mặt phẳng 4](#_Toc30074792)

[Hình 1.3. Xác định tọa độ các điểm cắt của đoạn thẳng P1 P2 và cửa sổ cắt 5](#_Toc30074793)

[Hình 2.1. Lựa chọn trình biên dịch 9](#_Toc30074794)

[Hình 2.2. Đặt tên cho project 9](#_Toc30074795)

[Hình 2.3.Nhập thông số từ bàn phím. 9](#_Toc30074796)

[Hình 2.4. Xét các điều kiện để có thể xảy ra ở cửa sổ bên trái 10](#_Toc30074797)

[Hình 2.5.Xét các điều kiện có thể xảy ra với cửa sổ cắt bên trên 11](#_Toc30074798)

[Hình 2.6. Xét các điều kiện có thể xảy ra với cửa sổ cắt bên phải 13](#_Toc30074799)

[Hình 2.7. Xét các điều kiện có thể xảy ra với cửa sổ cắt bên dưới 14](#_Toc30074800)

[Hình 3.1.Giao diện nhập từ bàn phím 15](#_Toc30074801)

[Hình 3.2. Các trường hợp cắt xén của đa giác 15](#_Toc30074802)

[Hình 3.3.Vùng màu vàng nét đứt là đa giác sau khi bị xén 16](#_Toc30074803)

# TÓM TẮT

Trong phần giới thiệu đã giới thiệu một số lý thuyết cơ bản về đồ họa máy tính từ đơn giản nhất cắt xén điểm, cắt xén đoạn thẳng tiếp đến là các kỹ thuật xén tỉa đa giác và giới thiệu các thuật toán tăng dần theo từng năm như thuật toán Cohen Sutherland, thuật toán Liang-Barsky, thuật toán Cyrus – Beck và thuật toán đã tối ưu hơn tất cả các thuật toán trước đó và trình bày thuật toán Sutherland của Hodgman trong cắt tỉa đa giác, đồng thời giới thiệu phần mềm DevC/C++ và thư viện grpahics.h .

Trong phần phương pháp đã trình bày các bước cài đặt phần mềm Dev\_C++ và thư viện graphic.h. Sau đó nêu ra cách nhập dữ liệu vào máy tình bằng hình thức như thế nào. Tiếp theo là nêu ra các trường hợp có thể xảy ra với cửa sổ cắt trong cắt xén đa giác theo thứ tự trái, phải, trên , dưới và các chiều đi mà các đoạn thẳng có thể xảy ra.

Trong phần kết quả trình bày các kết quả của đa giác sau khi nhập dữ liệu vào và kết quả sau khi bị cắt xén theo thứ tự trên, dưới, phải, trái.

# 1.GIỚI THIỆU

Đồ họa máy tính là một trong những lĩnh vực lí thú nhất và phát triển nhanh nhất của tin học. Ngay từ khi xuất hiện, đồ họa máy tính đã có sức lôi cuốn mãnh liệt, cuốn hút rất nhiều người ở nhiều lĩnh vực khác nhau như khoa học, nghệ thuật, kinh doanh, quản lí,... Tính hấp dẫn và đa dạng của đồ họa máy tính có thể được minh họa rất trực quan thông qua việc khảo sát các ứng dụng của nó.

Thuật toán xén tỉa trong đồ họa máy tính có tầm quan trọng rất lớn và được sử dụng rộng rãi trong các phần mềm phổ biến hiện này. Từ những phần mềm đơn giản như Paint, Powerpoint trong bộ Office của Window đến những ứng dụng thiết kế đồ họa chuyên nghiệp như Photoshop, AutoCad.

## 1.1 Thuật toán xén tỉa

Vùng mà đối tượng được xén tỉa được gọi là cửa sổ xén tỉa. Cửa sổ cắt có thể là hình chữ nhật, hình tròn, cửa sổ lồi, cửa sổ lõm, tùy vào các chiến lượt xén tỉa khác nhau dẫn đến các thuật toán xén tỉa giống nhau. Cửa sổ xén tỉa là hình chữ nhật, và đối tượng xén là 1 đa giác. Sử dụng thuật toán xén tỉa để loại bỏ phần hình ảnh bên ngoài cửa sổ xén ta được phần hình ảnh của đối tượng nằm bên trong cửa sổ xén tỉa. Xén tỉa là tiến trình xác định các điểm của một đối tượng nằm trong hay ngoài cửa sổ hiển thị. Nằm trong được hiển thị, nằm ngoài loại bỏ. Vùng được dùng để xén hình gọi là cửa sổ xén (clip window).

|  |
| --- |
|  |

Hình 1.1 Ví dụ minh họa của ứng dụng xén tỉa đa giác với Powerpoint 2013

Trong đó hình a thể hiện đa giác cần xén tỉa màu (màu xanh) và cửa sổ cắt màu đỏ.Hình b biểu diễn kết quả sau khi xén tỉa(vùng nét đứt màu xanh).Tùy thuộc vào các ứng dụng dùng để vẽ mà cửa sổ cắt xén có thể có dạng là hình đa giác hay đường cong khép kín. Đơn giản thuật toán xén tỉa đa giác là trường hợp cụ thể của xén tỉa đoạn thẳng.

## 1.2 Thuật toán xén tỉa đoạn thẳng

Thuật toán Cohen Sutherland là một thuật toán đồ họa máy tính . Thuật toán chia không gian hai chiều thành 9 vùng và sau đó xác định hiệu quả các đường và các phần của đường có thể nhìn thấy ở vùng trung tâm (khung nhìn).Thuật toán được phát triển vào năm 1967 trong quá trình mô phỏng chuyến bay của Daniel Cohen và Ivan Sutherland *(*[*https://en.wikipedia.org/wiki/Cohen%E2%80%93Sutherland\_algorithm*](https://en.wikipedia.org/wiki/Cohen%E2%80%93Sutherland_algorithm))*.*

Thuật toán Liang-Barsky (1983) được phát triển bởi Liang, You-Dong và Barsky, Brian A. Thuật toán này sử dụng phương trình tham số của một dòng và bất đẳng thức mô tả phạm vi của cửa sổ cắt để xác định các giao điểm giữa đoạn thẳng và cửa sổ cắt. Thuật toán này hiệu quả hơn so với Cohen-sutherland, nhưng Cohen-sutherland xác định vị trí tương đối của đoàn thẳng so với vùng nhanh hơn nhiều, còn thuật toán này xem xét các đoạn thẳng cần xén tỉa sẽ hoàn toàn vào hoặc ra khỏi của sổ cắt. Thuật toán Liang-Barsky hiệu quả hơn phiên bản Cyrus-Beck vì thử nghiệm loại bỏ tầm thường bổ sung có thể tránh tính toán cả bốn giá trị tham số cho các đường không giao nhau với hình chữ nhật clip. *(*[*https://stackoverflow.com/questions/48352036/how-can-i-measure-the-overlap-between-a-line-and-a-rectangle*](https://stackoverflow.com/questions/48352036/how-can-i-measure-the-overlap-between-a-line-and-a-rectangle)*).*

Giả sử (x,y) là toạ độ của một điểm, vậy điểm đó được hiển thị khi thoả mãn:

Xwmin <= x <= Xwmax

Ywmin <= y <= Ywmax

**Các trường hợp đoạn thẳng so với cửa sổ cắt**

|  |  |
| --- | --- |
| **TH1:** Các đoạn thẳng nằm ngoài hoàn toàn cửa sổ cắt thì không cắt đoạn thẳng. |  |
| **TH2:** Các đoạn thẳng nằm trong hoàn toàn cửa sổ cắt thì không cắt đoạn thẳng. |  |
| **TH3:** Nếu đoạn thẳng cắt cửa sổ cắt thì phân chia qua điểm cắt phần nằm trong hoặc nằm ngoài cửa sổ cắt. |  |

**Thuật toán được chia thành các bước sau:**

**Bước 1:** Gán mã vùng 4-bit cho mỗi điểm cuối của đoạn thẳng.

**Bước 2:** Mã vùng được xác định theo 9 vùng (hình 1.2) của mặt phẳng mà các điểm cuối nằm vào đó.

|  |  |
| --- | --- |
|  |  |

Hình 1.2. Mã vùng của 9 vùng mặt phẳng

**Bước 3:** Sử dụng các mã vùng để xác định các trường hợp của đoạn thẳng.

Xét mã vùng của 2 điểm đầu cuối P1, P2 của đoạn thẳng cần xén. Ta có các trường hợp sau:

Nếu mã của P1 hoặc P2 đều = 0000 thì toàn bộ đoạn thẳng thuộc phần hiển thị.

Nếu mã của P1 và P2 có cùng một vị trí mà P1 AND P2 != 0000 => cùng phía.

Nếu không nằm trong 2 trường hợp sau đường thẳng cần được xén tỉa.

Tìm giao điểm của đường thẳng với cửa sổ, (với phần mở rộng của đường biên).

Nếu: Bit 1 là 1: cắt y = ymax

Bit 2 là 1: cắt y = ymin

Bit 3 là 1: cắt x = xmax

Bit 4 là 1: cắt x = xmin

Đối với thuật toán Cohen Shutherland, khi tìm giao điểm của đoạn thẳng cần xén tỉa với các cạnh của cửa sổ cắt bằng cách dùng các tham số của phương trình đường thẳn nên chưa tối ưu về mặt thời gian.

Để khắc phục điều đó thuật toán Liang-Barsky ra đời. thuật toán Liang-Barsky (được đặt theo tên của You-Dong Liang và Brian A. Barsky) sử dụng phương trình tham số của đường thẳng và bất đẳng thức mô tả phạm vi của cửa sổ cắt để xác định các giao điểm giữa đường thẳng và ửa sổ cắt. Với các giao điểm này, nó biết phần nào của đường nên được vẽ. Thuật toán này hiệu quả hơn đáng kể so với Cohen-Sutherland.

Cả 2 thuật toán trên đều có điểm hạn chế là cửa sổ xén là hình chữ nhật và không cho phép cửa sổ hình đa giác khác, Cyrus – Beck (1978) đã giải quyết bài toán này.

**Minh họa thuật toán xén tỉa đoạn thẳng**

Phương pháp để tìm điểm cắt (x,y) của (x1,y1) (x2,y2)

Ta có:

hay

Nếu y nằm trên đường thẳng đứng:

Nếu x nằm trên đường thẳng nằm ngang:

Từ phương trình:

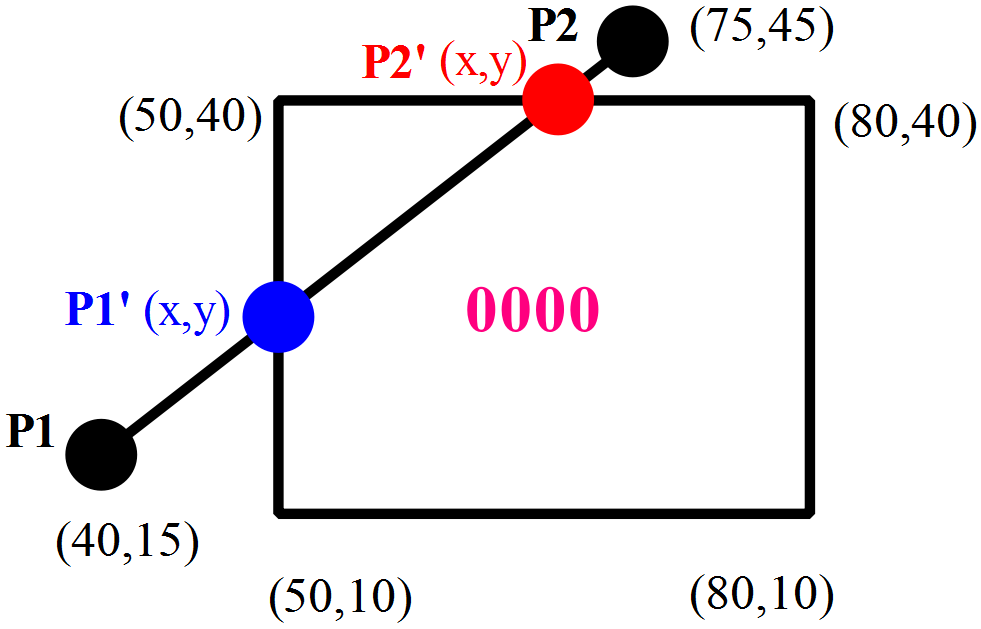
và

Cụ thể cho cửa sổ cắt tỉa hình chữ nhật có tạo độ A(50,10); B(80,10); C(80,40); D(50,40). Mã vùng dành cho các điểm cuối của đoạn thẳng có P1(40,15), P2(75,15) được xác định như sau.

* Ta có: và

xwmin=50; xwmax=80; ywmin=10; ywmax=40;

* Bit code (ABRL) P1: 0001 #0
* Bit code (ABRL) P2: 1000 #0
* AND(P1,P2)=0000 nên P1P2 thỏa điều kiện cần xén
* Xác định điểm cắt thứ nhất P1’(50,y):
* Xác định điểm cắt thứ hai P2’(x,40):



Hình 1.3. Xác định tọa độ các điểm cắt của đoạn thẳng P1 P2 và cửa sổ cắt

Trong đó điểm P1’(x,y) được biểu diễn bởi màu xanh là điểm cắt thứ nhất của đoạn thẳng P1 với cửa sổ cắt hình chữ nhật.Điểm P2’(x,y) có màu đỏ là điểm cắt thứ hai giữa canh CD của hình chữ nhật và đoạn thẳng P2.

## 1.3 Thuật toán xén tỉa đa giác

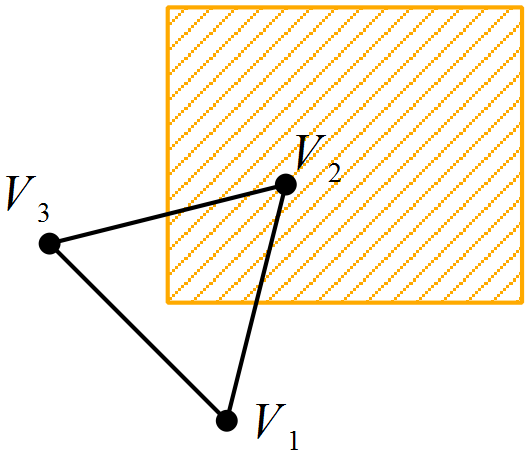
Để cắt đa giác, chúng ta cần sửa đổi các bước xén tỉa đoạn thẳng. Một cạnh đa giác được xử lý với một cạnh của của sổ xén tùy thuộc vào hướng của cạnh đa giác đến cửa sổ xén. Những gì chúng ta cần là phần giới hạn sau khi cắt. Để xén tỉa đa giác, chúng ta cần một thuật toán sẽ tạo ra một hoặc nhiều vùng xén (các cạnh của cửa sổ xén). Đầu ra của một đa giác đã được xén tỉa phải là một chuỗi các đỉnh xác định ranh giới của đa giác được xén tỉa.

Thuật toán Sutherland của Hodgman là một thuật toán được sử dụng để cắt các đa giác. Nó hoạt động bằng cách lần lượt mở rộng từng dòng của đa giác clip lồi và chỉ chọn các đỉnh từ đa giác ([*https://en.wikipedia.org/wiki/Sutherland%E2%80%93Hodgman\_algorithm*](https://en.wikipedia.org/wiki/Sutherland%E2%80%93Hodgman_algorithm)*).*

Cho P1,P2,…,PN là danh sách các đỉnh của đa giác. Cho cửa sổ cắt tỉa ABCD. Thứ tự các trường hợp sẽ được xén như sau:

|  |  |
| --- | --- |
| **TH1**. v1 bên trong mặt phẳng và v2 bên trong mặt phẳng nên đoạn thẳng sẽ không bị cắt. |  |
| **TH2**. v1 bên trong mặt phẳng và v2 bên ngoài mặt phẳng nên v’1 là điểm cắt với chiều từ trong ra ngoài. |  |
| **TH3**. v1 mặt phẳng bên ngoài và v2 bên ngoài mặt phẳng nên đoạn thẳng sẽ không bị cắt. |  |
| **TH4**. v1 là điểm bên ngoài và v2 bên trong mặt phẳng nên v’1 là điểm cắt với chiều từ ngoài vào trong. |  |

Cho đa giác dương sau và cửa sổ cắt, hãy xác định các đỉnh cắt:



|  |  |
| --- | --- |
| ***Bước 0:*** Áp dụng quy luật được nêu ở trên cho đa giác dương để xác định chiều của các cạnh. |  |
| ***Bước 1:*** **Cắt bên trái**  Xét cạnh V1-V2 (In-In): V2  Xét cạnh V2-V3 (In-Out): V’2  Xét cạnh V3-V1(Out-In): V3’-V1 |  |
| ***Bước 2:*** Cắt bên phải  Xét cạnh V1-V2 (In-In): V2  Xét cạnh V2-V’2 (In-In): V’2  Xét cạnh V’2-V’3 (In-In): V’3  Xét cạnh V’3-V1(In-In): V1 |  |
| ***Bước 3:*** Cắt bên trên  Xét cạnh V1-V2 (In-In): V2  Xét cạnh V2-V’2 (In-In): V’2  Xét cạnh V’2-V’3 (In-In): V3’  Xét cạnh V’3-V1(In-In): V1 |  |
| ***Bước 4:*** Cắt bên dưới  Xét cạnh V1-V2 (Out-In): V1’, V2  Xét cạnh V2-V’2 (In-In): V’2  Xét cạnh V’2-V’3 (In-OUT): V2’’  Xét cạnh V’3-V1(Out-Out): NULL |  |
| ***Bước 5:*** Vẽ lại các cạnh lại theo thứ tự |  |

## 1.4. DevC/C++ và thư viện grpahics.h

Dev-C++ (Dev-Cpp,  phiên bản  hiện tại  là  4.9.9.2) là một bộ công cụ phát triển tích hợp (IDE Integrated Development Environment) các ứng dụng C/C++ thuộc dạng mã nguồn mở. DevCpp dựa  trên  trình biên dịch mã nguồn mở MinGW  (Minimalist GNU\* for Windows, http://www.mingw.org/). MinGW sử dụng GCC (the GNU g++ compiler collection) sử dụng cho cả hệ thống Windows và Linux.

  Thư viện đồ họa graphics.h hay tên chính xác và đầy đủ của nó là Borland Graphics Interface – còn được biết đến với tên gọi BGI – là một thư viện đồ họa rất phổ biến trên DOS và các máy tính chạy hệ điều hành Windows thời kì đầu như Windows 95, Windows 98. Thư viện này cung cấp cho người dùng 2 file: graphics.h và graphics.lib để có thể sử dụng được với ngôn ngữ C/C++.

# 2. PHƯƠNG PHÁP

## 2.1. Các bước cài đặt Dev\_C++ và thư viện graphic.h

**Bước 1:** Dowload và cài đặt Dev\_C++ cho máy tính theo hướng dẫn của đường dẫn sau đây.( <https://quantrimang.com/huong-dan-cai-dev-c-158149>)

**Bước 2:** Tải thư viện đồ họa về máy tính theo đường dẫn sau đây. (<http://www.mediafire.com/file/q4ldrr5svd1q4xh/VietSource.net_-_Graphics_in_Dev-C.zip/file>).

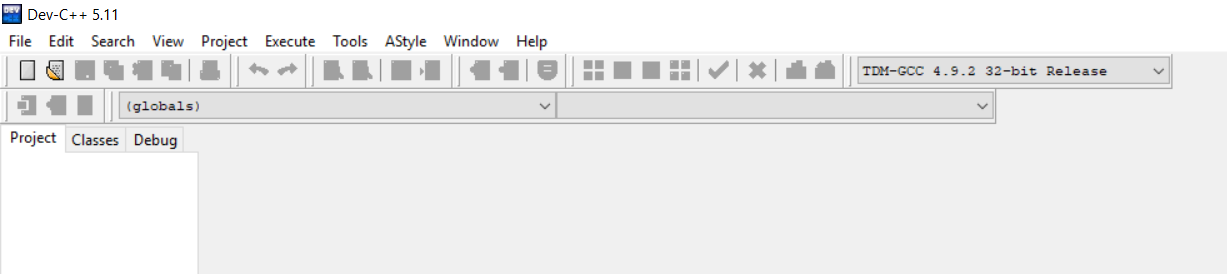
**Bước 3:** Sau khi down về bạn giải nén sẽ thấy một thư mục “VietSource.net – Graphics in Dev-C” trong thư mục có 5 file.

**Bước 4:** Copy – file *libbgi.a* vào thư mục*lib* và paste vào đường dẫn C:\Program Files\Dev-Cpp\MinGW64\lib.

**Bước 5:** Copy file *winbgim.h* và*graphics.h*vào thư mục *include*và paste vào đường dẫn C:\Program Files\Dev-Cpp\MinGW64\include.

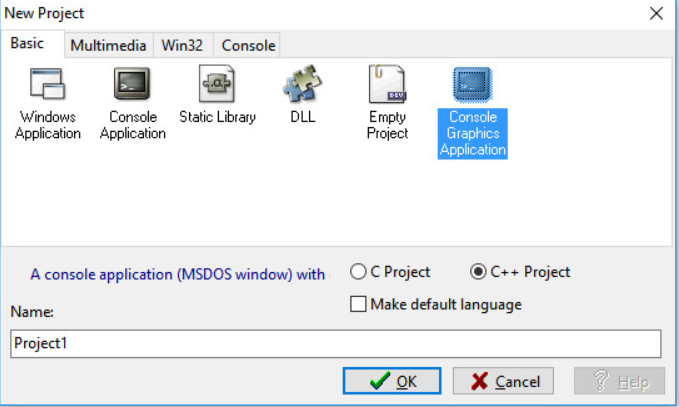
**Bước 6:** Copy 2 file *6-ConsoleAppGraphics.template* và file *ConsoleApp\_cpp\_graph.txt* vào thư mục *Templates* và paste vào đường dẫn C:\Program Files\Dev-Cpp\Templates.

**Bước 7:** Sau khi cài đặt Dev\_C++ và thư viện graphic theo hướng dẫn ở trên sau đó hãy mở Dev\_C ++ lên và chọn trình biên dịch là *TDM-GCC 32 bit Release*như khung màu đỏ hiển thị ảnh bên dưới**.**

****

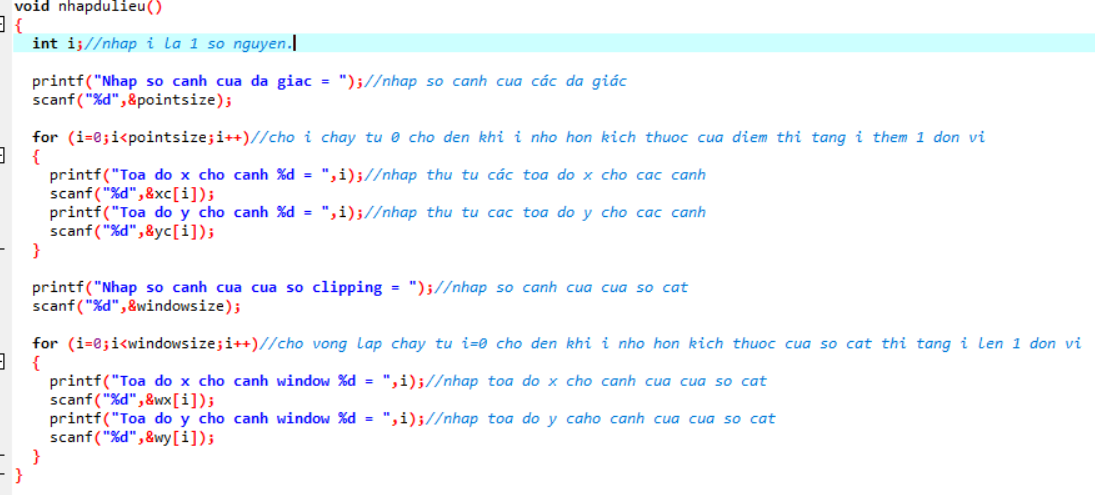
Hình 2.1. Lựa chọn trình biên dịch

**Bước 8:** Tạo một Project mới bằng cách chọn File → New → Project.Chọn Console Graphics Application sau đó chọn ô C++ Project,gõ tên Project vào ô Name và nhấn OK rồi chọn vị trí lưu trên máy tính (vùng màu đỏ).



Hình 2.2. Đặt tên cho project

## 2.2.Nhập dữ liệu



Hình 2.3.Nhập thông số từ bàn phím.

Nhập dữ liệu số cạnh của đa giác từ bàn phím.

Nhập vị trí tọa độ (x,y) cho các cạnh từ bàn phím.

Nhập số cạnh của cửa sổ cắt từ bàn phím.

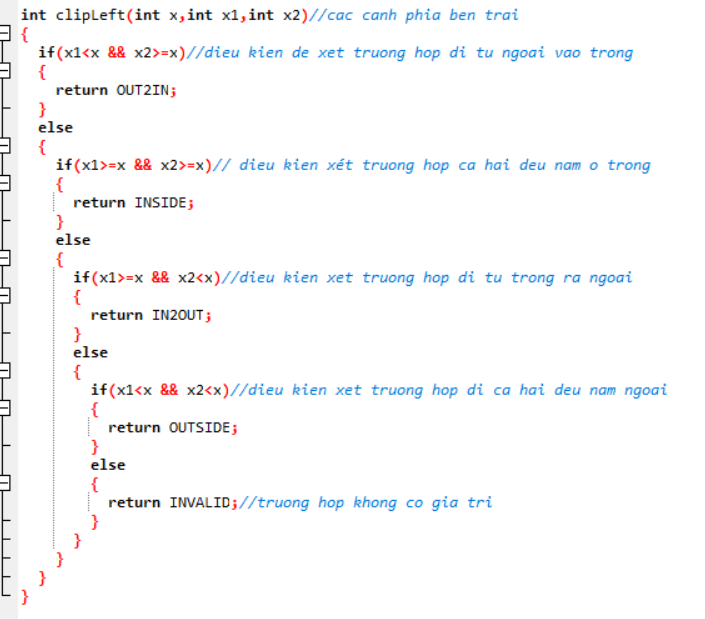
Nhập tọa độ(x,y) cho các cạnh của cửa sổ cắt từ bàn phím.

## 2.3. Thuật toán cắt xén đa giác

P1 (x1,y1), P2 (x2,y2) và cửa sổ cắt (hình chữ nhật ABCD, với các cạnh AB )cạnh trái), BC (cạnh trên), CD (Cạnh phải), AD (cạnh dưới). Vị trí tương đối của (P1,P2) so với với các cạnh của cửa sổ cắt thuộc 1 trong 4 trường hợp như sau:

Nếu đỉnh đầu tiên là bên ngoài cửa sổ đỉnh thứ hai là bên trong cửa sổ thì đỉnh thứ hai được thêm vào danh sách đầu ra. Các điểm giao nhau của ranh giới cửa sổ và mặt đa giác (cạnh) cũng được bổ sung vào dòng đầu ra. Nếu cả hai các đỉnh là bên trong ranh. giới cửa sổ thì chỉ có đỉnh thứ hai được thêm vào danh sách đầu ra. Nếu đỉnh đầu tiên là bên trong cửa sổ và thứ hai là một cửa sổ bên ngoài. Cạnh cắt cửa sổ sẽ được thêm vào danh sách đầu ra. Nếu cả hai đỉnh bên ngoài cửa sổ thì không có gì được thêm vào danh sách đầu ra.

2.3.1. Trường hợp cắt xén cạnh bên trái

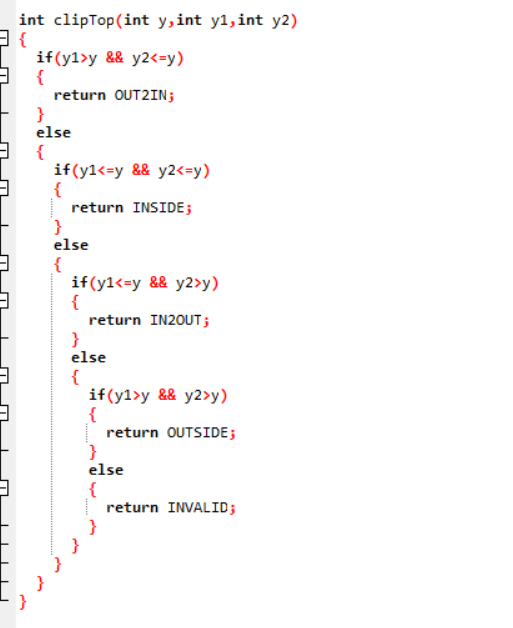


Hình 2.4. Xét các điều kiện để có thể xảy ra ở cửa sổ bên trái

|  |  |
| --- | --- |
| **Trường hợp 1**: Cửa sổ cắt ở phía bên trái và đoạn thẳng đi theo hướng từ ngoài vào trong. |  |
| **Trường hợp 2**:  Cửa sổ cắt nằm phía bên trái và đoạn thẳng nằm hoàn toàn bên trong cửa sổ cắt. |  |
| **Trường hợp 3**:Cửa sổ cắt ở phía bên trái và đoạn thẳng đi theo hướng từ trong ra ngoài |  |
| **Trường hợp 4**:Cửa sổ cắt nằm phía bên trái và đoạn thẳng nằm ngoài hoàn toàn cửa sổ cắt.Và cuối cùng sẽ in ra chiều của đoạn thẳng đó. |  |

Sau khi xén xong tất cả các cạnh của đa giác với cạnh trái của cửa sổ cắt, thì sao chép mảng tạm là kết quả vừa xén được, vào mảng cạnh của đa giác để làm dữ liệu cho lần xén tiếp theo với cạnh khác của đa giác.

**2.3.2. Trường hợp cắt cửa sổ bên trên**

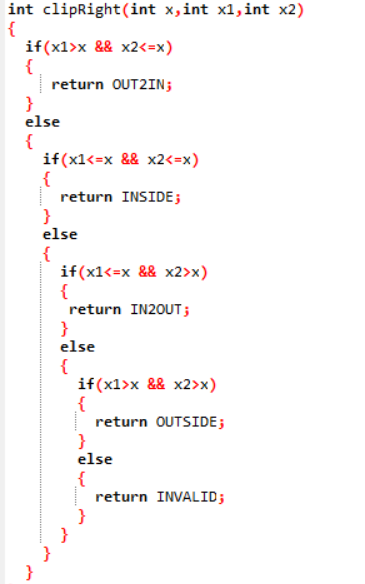


Hình 2.5.Xét các điều kiện có thể xảy ra với cửa sổ cắt bên trên

|  |  |
| --- | --- |
| **Trường hợp 1**: cửa sổ cắt ở phía bên trên và đoạn thẳng đi theo hướng từ ngoài vào trong. |  |
| **Trường hợp 2:**  *Cửa sổ cắt nằm phía bên trên và đoạn thẳng nằm hoàn toàn bên trong cửa sổ cắt* |  |
| **Trường hợp 3**: Cửa sổ cắt ở phía bên trên và đoạn thẳng đi theo hướng từ trong ra ngoài |  |
| **Trường hợp 4**: Cửa sổ cắt nằm phía bên trên và đoạn thẳng nằm ngoài hoàn toàn cửa sổ cắt. |  |

Sau khi xén xong tất cả các cạnh của đa giác với cạnh bên trên của cửa sổ cắt, thì sao chép mảng tạm là kết quả vừa xén được, vào mảng cạnh của đa giác để làm dữ liệu cho lần xén tiếp theo với cạnh khác của đa giác.

### 2.3.3.Trường hợp cắt xén của cửa sổ bên phải

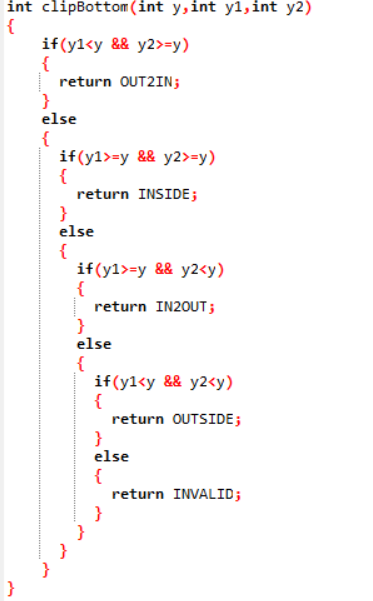


Hình 2.6. Xét các điều kiện có thể xảy ra với cửa sổ cắt bên phải

|  |  |
| --- | --- |
| **Trường hợp 1**: cửa sổ cắt ở phía bên phải và đoạn thẳng đi theo hướng từ ngoài vào trong. |  |
| **Trường hợp 2**:  Cửa sổ cắt nằm phía bên phải và đoạn thẳng nằm hoàn toàn bên trong cửa sổ cắt. |  |
| **Trường hợp 3**:Cửa sổ cắt ở phía bên phải và đoạn thẳng đi theo hướng từ trong ra ngoài |  |
| **Trường hợp 4**:Cửa sổ cắt nằm phía bên phải và đoạn thẳng nằm ngoài hoàn toàn cửa sổ cắt. |  |

Sau khi xén xong tất cả các cạnh của đa giác với cạnh bên phải của cửa sổ cắt, thì sao chép mảng tạm là kết quả vừa xén được, vào mảng cạnh của đa giác để làm dữ liệu cho lần xén tiếp theo với cạnh khác của đa giác.

### 2.3.4. Trường hợp cắt xén của cửa sổ bên dưới



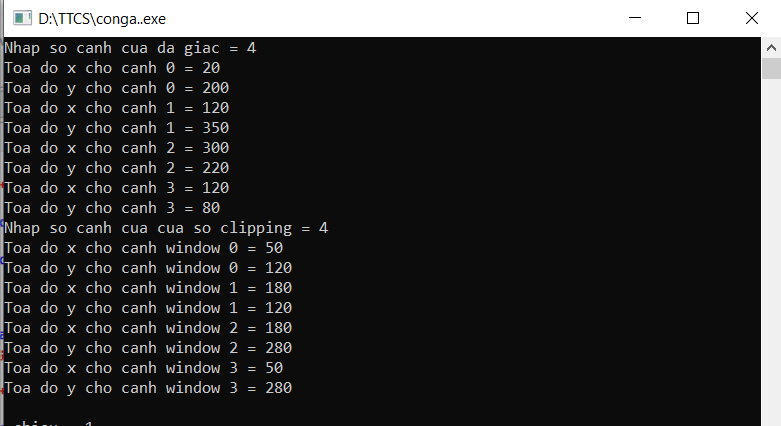
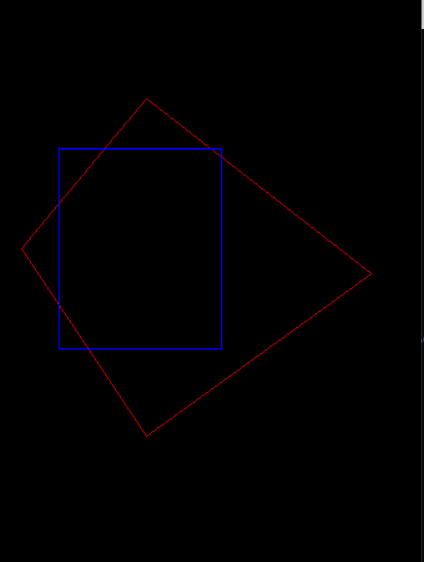
Hình 2.7. Xét các điều kiện có thể xảy ra với cửa sổ cắt bên dưới

|  |  |
| --- | --- |
| **Trường hợp 1**: cửa sổ cắt ở phía bên dưới và đoạn thẳng đi theo hướng từ ngoài vào trong. |  |
| **Trường hợp 2**:  Cửa sổ cắt nằm phía bên dưới và đoạn thẳng nằm hoàn toàn bên trong cửa sổ cắt. |  |
| **Trường hợp 3**:Cửa sổ cắt ở phía bên dưới và đoạn thẳng đi theo hướng từ trong ra ngoài |  |
| **Trường hợp 4**:Cửa sổ cắt nằm phía bên phải và đoạn thẳng nằm ngoài hoàn toàn cửa sổ cắt. |  |

Sau khi xét các điều kiện về chiều của đoạn thẳng và cắt xong được các cạnh mới ở cửa sổ phía bên dưới ta sẽ có danh sách vị trí của cạnh mới và gán lại số đỉnh mới được hình thành sau khi cắt xén.

# 3. KẾT QUẢ

## 3.1. Dữ liệu nhập từ bàn phím



Hình 3.1.Giao diện nhập từ bàn phím

## 3.2.Kết quả của xén tỉa đa giác

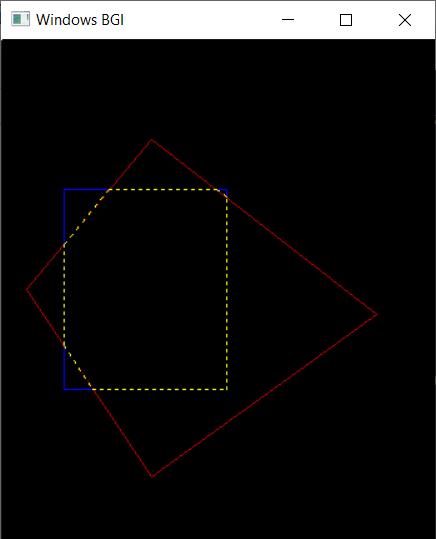
### 3.2.1. Xét trường hợp cắt xén từ bên trái

|  |  |  |  |
| --- | --- | --- | --- |
| Hình a | *Hình b* | *Hình c* | *Hình d* |

Hình 3.2. Các trường hợp cắt xén của đa giác

Trong đó cửa sổ cắt là hình chữ nhật có màu xanh, màu vàng nét đứt là đa giác bị cắt xén. Phần màu đỏ nét đứt trong hình(a) là vùng bị giới hạn bới cửa sổ cắt nằm ở phía bên trái. Phần màu đỏ nét đứt trong hình (b) là vùng bị giới hạn bởi cửa sổ cắt nằm ở phía bên trên. Phần màu đỏ nét đứt trong hình (c) là vùng bị giới hạn bởi cửa sổ cắt nằm ở phía bên phải.Phần màu đỏ nét đứt trong hình(d) là vùng bị giới hạn bởi cửa sổ cắt nằm phía bên dưới.

### 3.2.2. Kết quả đa giác sau khi cắt theo chiều từ trái→ trên→ phải→ dưới



Hình 3.3.Vùng màu vàng nét đứt là đa giác sau khi bị xén

# 4. THẢO LUẬN

Thuật toán này sẽ tiến hành xén đa giác lần lượt với các biên cửa sổ. Đầu tiên, đa giác sẽ được xén dọc theo biên trái của cửa sổ, kết quả sau bước này sẽ được dùng để xén tiếp biên phải, rồi cứ tương tự như vậy cho các biên trên, dưới,trái,phải. Sau khi xén hết với bốn biên của cửa sổ, ta được kết quả cuối cùng.

Thuật toán Sutherland-Hodgeman cho kết quả chính xác khi làm việc với các đa giác lồi, tuy nhiên với các đa giác lõm (thì phải nghiên cứu sau mới đưa ra được kết luận của thuật toán Sutherland-Hodgeman có cắt chính xác trong mọi trường hợp hay không).