TRƯỜNG ĐẠI HỌC NHA TRANG

**KHOA CÔNG NGHỆ THÔNG TIN**



**Đề tài: Cài đặt thuật toán xén tỉa đa giác bằng thuật toán Sutherland-Hogmand**

**GVHD : Ths. ĐOÀN VŨ THỊNH**

**SVTH : Nguyễn Thị Kim Nga**

**MSSV : 59131500**

**Lớp : 59CNTT-3**

Khánh Hòa, tháng 01 năm 2020

**MỤC LỤC**

[1. GIỚI THIỆU 1](#_Toc30090361)

[1.1. Thuật toán xén tỉa 2](#_Toc30090362)

[1.2. Kỹ thuật xén tỉa đoạn thẳng 2](#_Toc30090363)

[1.3. Kỹ thuật xén tỉa đa giác 5](#_Toc30090364)

[1.4. DevC/C++ 9](#_Toc30090365)

[1.5. Graphics.h 10](#_Toc30090366)

[2. PHƯƠNG PHÁP NGHIÊN CỨU 10](#_Toc30090367)

[2.1. Cài đặt Dev-C++ và Graphics.h 10](#_Toc30090368)

[2.2. Khai báo biến 11](#_Toc30090369)

[2.3. Nhập dữ liệu 11](#_Toc30090370)

[2.4. Vẽ cửa sổ cắt và đa giác 12](#_Toc30090371)

[2.5. Cài đặt thuật toán xém tỉa đa giác 12](#_Toc30090372)

[2.5.1. Trường hợp cắt cạnh bên trái 14](#_Toc30090373)

[2.5.2. Trường hợp cắt bên phải 15](#_Toc30090374)

[2.5.3. Trường hợp cắt bên trên 16](#_Toc30090375)

[2.5.4. Trường hợp cắt bên dưới 16](#_Toc30090376)

[2.5.5. Vẽ đa giác mới sau khi cắt 18](#_Toc30090377)

[3. KẾT QUẢ 18](#_Toc30090378)

[4. KẾT LUẬN 19](#_Toc30090379)

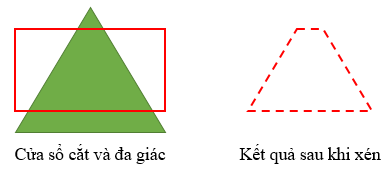
# 1. GIỚI THIỆU

Kỹ thuật đồ họa máy tính là phương tiện đa năng và mạnh nhất của giao tiếp giữa con người và máy tính, ở đó nghiên cứu, xây dựng và tập hợp các công cụ để kiến tạo, xây dựng, lưu trữ, xử lý các mô hình của đối tượng. Kỹ thuật đồ họa máy tính đã trải qua nhiều giai đoạn phát triển như: Thiết bị đồ họa đầu tiên là màn hình xuất hiện tại Đức (1959). SAGE – Semi Automatic Ground Environment System (1960) xuất hiện bút sáng thao tác với màn hình. Raster Graphics (1970). Xuất hiện các chuẩn công nghiệp (1980): PHIGS - Programmers Hierarchical Interactive Graphics Standard; HC/MI – Human Computer/Machine Interface. OpenGL API - Application Program Interface (1990). Với lịch sử hình thành và phát triển hơn nửa thế kỉ thì kỹ thuật đồ họa đã mang đến muôn vàn ứng dụng cho các lĩnh vực khác nhau. Hình 2.1 là ví dụ sống động cho ứng dụng kỹ thuật đồ họa trong việc xây dựng các nhân vật hoạt hình trong bộ phim Kungfu Panda.



Hình 1.1. Các nhân vật trong bộ phim hoạt hình Kungfu Panda (2008)

Trong rất nhiều các giải thuật cho đồ họa máy tính, tác giả đề xuất “Cài đặt thuật toán xén tỉa đa giác bằng thuật toán Sutherland - Hogman” để minh họa cho khái niệm xén tỉa là tiến trình xác định các điểm của một đối tượngkỹ thuật cụ thể. Hình 1.2 là ứng dụng kỹ thuật xén tỉa đa giác được cài đặt mặc định trong phần mềm PowerPoint 2013. Về cơ bản, kỹ thuật xén tỉa đa giác chính là vận dụng kỹ thuật xén tỉa đoạn thằng.

****

*Hình 2.2. Ví dụ về thuật toán xén tỉa đa giác trong phần mềm PowerPoint 2013*

Trong đó, vùng màu xanh là đa giác cần xén tỉa, vùng màu đỏ là cửa sổ cắt. Các đường nét đứt là kết quả hiển thị sau khi xén.

## 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.

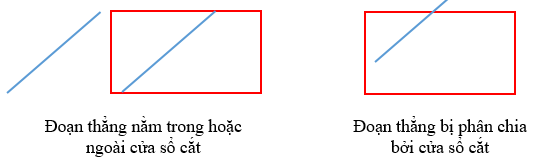
Đối tượng được xén tỉa có thể là đường thẳng, đa giác, hình tròn, ký tự hoặc đường cong không đều. Để xén tỉa được một đa giác trên ta phải cắt tỉa từng đoạn thẳng là mỗi cạnh của đa giác. Từ đó dẫn tới sự ra đời của thuật toán xén tỉa đoạn thẳng.

## 1.2. Kỹ thuật xén tỉa đoạn thẳng

Về xén tỉa đoạn thẳng có hai trường hợp:

Các đoạn thẳng không cắt cửa sổ xén tỉa: hoặc nằm trong hoàn toàn, hoặc nằm ngoài hoàn toàn cửa sổ cắt (Hình 1.3 trái).

Cửa sổ xén tỉa phân chia đoạn thẳng thành 2 phần: điểm cắt nằm trong và ngoài cửa sổ xén tỉa (Hình 1.3 phải).

****

Hình 1.3. Ví dụ minh họa về 2 trường hợp đoạn thẳng được xén tỉa

Xét trường hợp đoạn thẳng bị phân chia bởi cửa sổ cắt thì lại chia thành 4 trường hợp nhỏ nữa:

Trường hợp cắt cửa sổ bên trên (Hình 1.4.c).

Trường hợp cắt cửa sổ bên dưới (Hình 1.4.d).

Trường hợp cắt cửa sổ bên trái (Hình 1.4.a).

Trường hợp cắt cửa sổ bên phải (Hình 1.4.b).

|  |  |
| --- | --- |
| Hình 1.4.a Trường hợp cắt cửa sổ bên trái. | Hình 1.4.c Trường hợp cắt cửa sổ bên trên |
| Hình 1.4.b Trường hợp cắt cửa sổ bên phải. | Hình 1.4.c Trường hợp cắt cửa sổ bên dưới |

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;

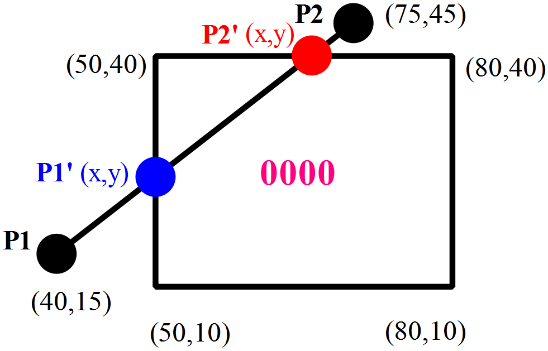
Bit code (ABRL) P1: 0001 #0

Bit code (ABRL) P2: 1000 #0

AND(P1,P2)=0000 nên P1P2 thỏa điều kiện cần xén

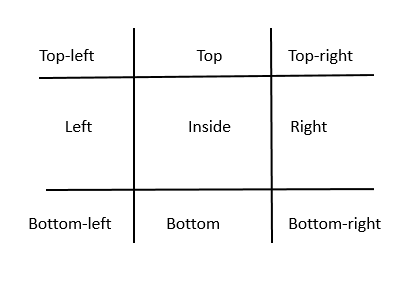
Xác định điểm cắt thứ nhất P1’(50,y):

Xác định điểm cắt thứ hai P2’(x,40):



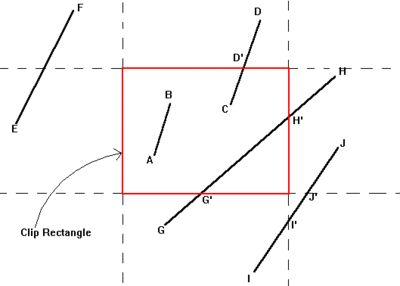
Hình 1.5. Xác định tọa độ các điểm cắt của đoạn thẳng P1 P2 và cửa sổ cắt.

Các giả thuật xén tỉa đoạn thẳng như: thuật toán Cohen-Sutherland là (1967) - một trong những thuật toán ra đời sớm nhất và thông dụng nhất. Thuật toán được phát triển trong quá trình mô phỏng chuyến bay của Daniel Cohen chuyên về mạng máy tính và Ivan Edward Sutherland người được coi là “cha đẻ của đồ họa máy tính”. Thuật toán sử dụng để cắt dòng. Bằng cách kéo dài các biên của cửa sổ, người ta chia mặt phẳng thành chín vùng gồm cửa sổ và tám vùng xung quanh nó (Hình 1.6).

****

Hình 1.6. Chia mặt phẳng thành chín vùng gồm cửa sổ và tám vùng xung quanh nó.

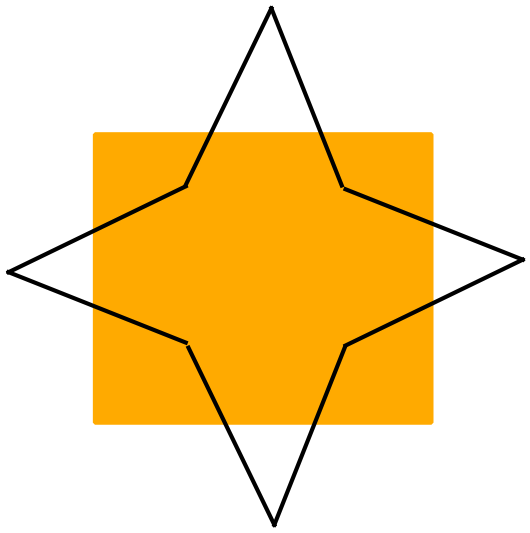
Hạn chế của thuật toán là chỉ áp dụng cho các cửa sổ hình chữ nhật và không áp dụng cho các cửa sổ hình lồi khác. Vì vậy thuật toán Liang-Barsky ra đời (1984). Thuật toán được phát triển bởi hai nhà khoa học You-Dong Liang và Brian A Barsky. Thuật toán này sử phương trình tham số của đường thẳng và bất đẳng thức mô tả phạm vi của cửa sổ cắt để xác định các giao điểm giữa đường thẳng và cửa sổ cắt. Với giao điểm này nó biết phần nào của đường nên được vẽ (Hình 1.7).



Hình 1.7. <http://algorithm-wiki.org/wiki2/images/thumb/4/48/At10.gif/400px-At10.gif>

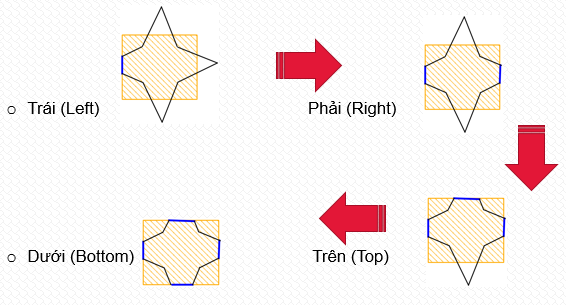
## 1.3. Kỹ thuật xén tỉa đa giác

Về giải thuật xén tỉa đa giác (Sutherland Hodgman). Thuật toán được phát triển vào năm 1974 bởi Ivan Sutherland và Gary W.Hodgman. Thuật toán này liên quan đến đa giác lồi và đa giác lõm. Khái niệm đa giác lồi là đa giác đường thẳng nối bất kỳ hai điểm bên trong nào của đa giác đều nằm trọn trong đa giác. Đa giác không lồi là đa giác lõm. Nó hoạt động bằng cách lần lượt mở rộng từng đoạn thẳng của đa giác lồi và chỉ chọn các đỉnh từ đa giác cần xén nằm ở phía có thể nhìn thấy. Nếu đa giác cần xén là đa giác lõm ở các đỉnh bên ngoài đa giác cắt, đa giác mới có thể có các cạnh trùng nhau (nghĩa là chồng chéo). 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 bên 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. Sau khi xén hết với bốn biên của cửa sổ, ta được kết quả cuối cùng.



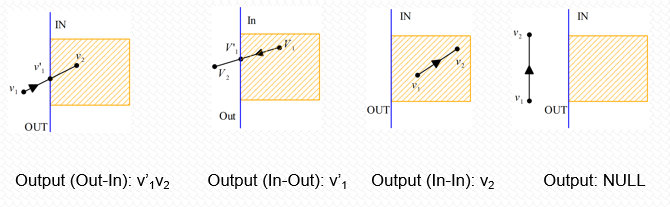
Hình 1.8. Cửa sổ cắt và đa giác

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 (Hình 1.9).

****

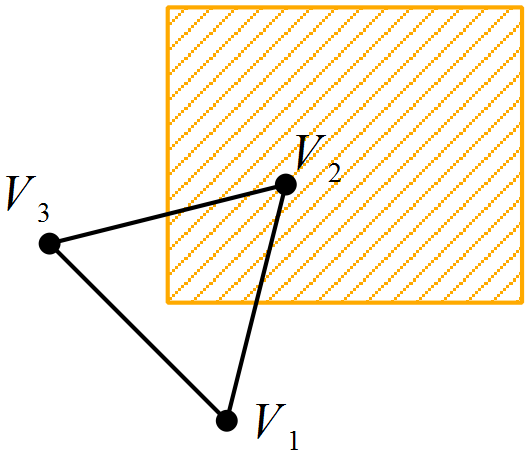
Hình 1.9. Minh họa thứ tự xén.

Xét 4 trường hợp (Hình 1.10).

****

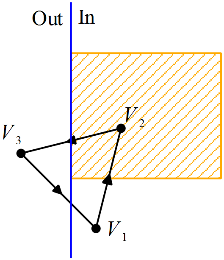
Hình 1.10. Minh họa xén tỉa trong từng trường hợp.

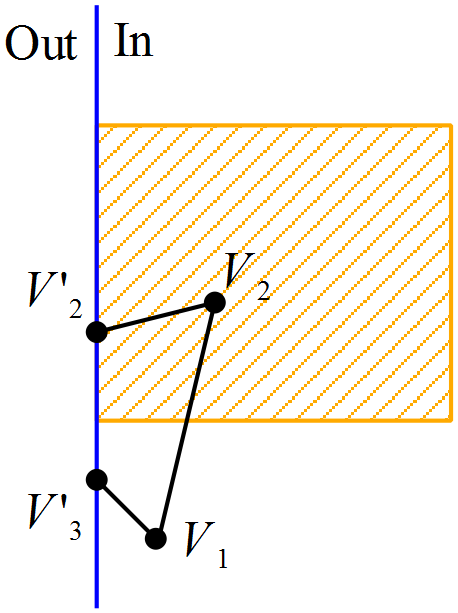
Cho đa giác dương sau và cửa sổ cắt, hãy xác định các đỉnh cắt (Hình 1.11).



Hình 1.11. Đa giác dương và cửa sổ cắt.

***Bước 0****:* Áp dụng quy luật được nêu ở mục 1.3 cho đa giác dương để xác định chiều của các cạnh.



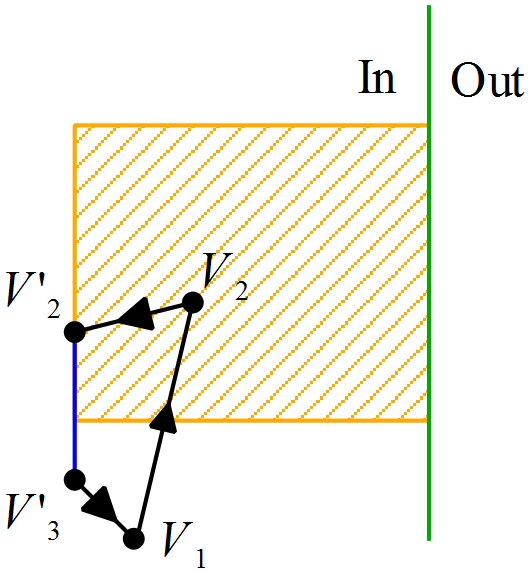


***Bước 1***: Cắt bên trái

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

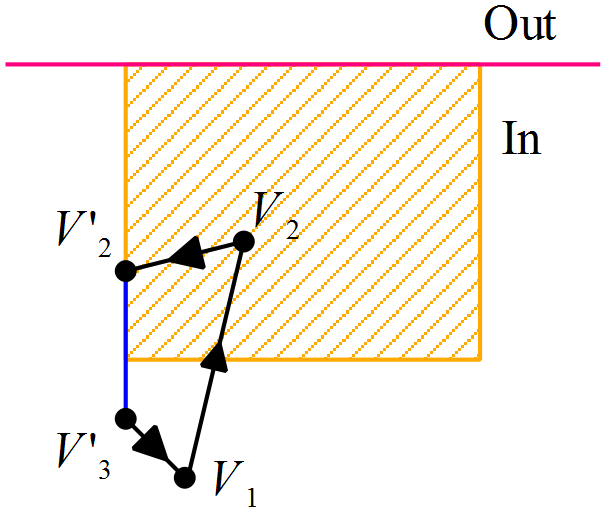
Xét cạnh V1-V2 (In-In): V2

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

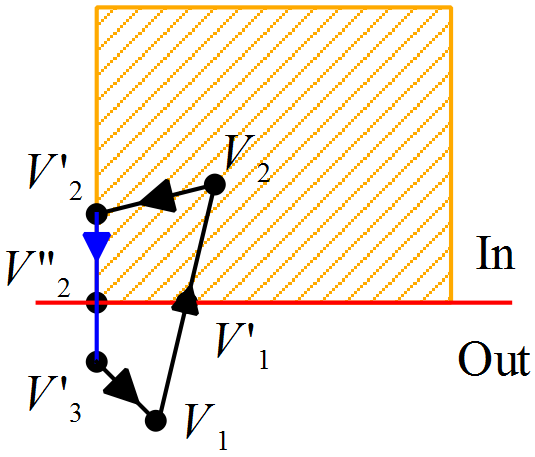


Xét cạnh V1-V2 (In-In): V2

Xét cạnh V2-V’2 (In-In): V’2

Xét cạnh V’2-V’3 (In-In): V’3

Xét cạnh V’3-V1(In-In): V1

***Bước 4***: Cắt bên dưới

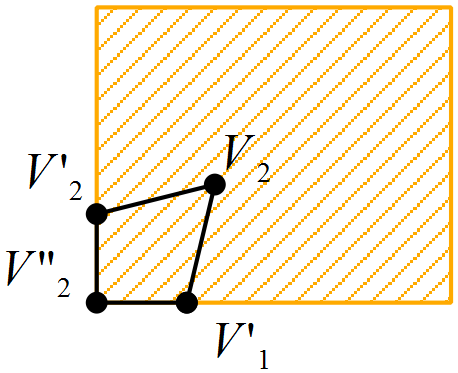
Xét cạnh V1-V2 (Out-In): V1’, V2

Xét cạnh V2-V’2 (In-In): V’2

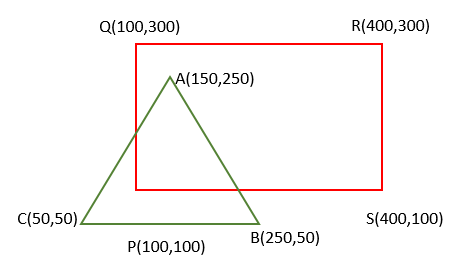
Xét cạnh V’2-V’3 (In-OUT): V2’’

Xét cạnh V’3-V1(Out-Out): NULL

***Bước 5***: Vẽ lại các cạnh theo thứ tự các điểm ở trên



Xét bài toán xén tỉa đa giác cụ thể với cửa sổ cắt PQRS lần lượt có tọa độ sau P(100,100), Q(100,300), R(400,300), S(400,100) và đa giác ABC với tọa độ A(150,250), B(250,50), C(50,50) (Hình 1.12).

****

Hình 1.12. Cửa sổ cắt PQRS và đa giác cần xén tỉa ABC.

Để xén tỉa đa giác ABC ta thực hiện xén tỉa như sau:

***Bước 1***: cắt bên trái cạnh PQ

Xét cạnh CA có chiều Out – In

Điểm C nằm bên trái PQ

Điểm A nằm bên phải PQ

Tìm giao điểm C’thuộc đường thằng CA

C’(100,150)

***Bước 2***: cắt bên dưới cạnh PS

Xét cạnh AB có chiều In – Out

Điểm A nằm trên PS

Điểm B nằm dưới PS

Tìm giao điểm A’ thuộc đường thẳng AB

Xét cạnh BC có chiều Out – Out: NULL.

Để thực hiện các thuật toán xén tỉa đoạn thẳng, xén tỉa đa giác cần sử dụng ngôn ngữ lập trình Dev-C++ và thư viện graphics. h.

## 1.4. DevC/C++

Dev-C++ 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). MinGW sử dụng GCC (the GNU g++ compiler collection) sử dụng cho cả hệ thống Windows và Linux. Hiện nay Dev-C++ là công cụ phát triển các ứng dụng C/C++ được úng dụng rộng rãi để dạy về lập trình cũng như để phát triển các ứng dụng mã nguồn mở. Mặc dù không có nhiều tính năng cao cấp như IDE thương mại khác (Visual Studio của Microsoft) nhưng DevCpp vẫn là một công cụ rất thích hợp trong môi trường giáo dục ở bậc đại học khi dạy và học các ngôn ngữ C/C++.

DevC/C++ phát triền trên nền Windows do vậy để lập trình giao diện đồ họa như BorlandC++ Builder 6.0 (https://taimienphi.vn/download-c-fundamentals-for-borland-c-builder-19020/60-phien-ban) gặp không ít trở ngại. Để giải quyết khó khăn này một thư viện mở rộng cho phép các lập trình viên thực

## 1.5. Graphics.h

Thư viện Graphicss.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 hai file graphics.h và graphics.lib để có thể sử dụng đucợ với ngôn ngữ C/C++ cũng như module graph nếu người dùng sử dụng ngôn ngữ Pascal. Bộ thư viện này đi kèm với IDE Borland C++ 3.1 (1992). Nó chứa một số hàm vẽ ảnh và cách tạo ảnh chuyển động, in chữ. Một trong những điểm mạnh của thư viện này là việc khởi tạo cũng như sử dụng rất đơn giản, vì vậy mặc dù ra đời rất lâu nhưng hiện tại vẫn có rất nhiều trường đại học sử dụng cho mục đích giảng dạy.**(**<https://github.com/thinhdoanvu/ComputerGraphics/tree/master/Coding/library>).

# 2. PHƯƠNG PHÁP NGHIÊN CỨU

## 2.1. Cài đặt Dev-C++ và Graphics.h

**Cài đặt Dev-C++ (**[**Download DevCpp-4.9.9.2\_Setup**](http://www.mediafire.com/download.php?ud31llsxsms360y)**).**

**Cài đặt thư viện Graphics.h vào Dev-C++**

***Bước 0:*** Tải thư viện và giải nén

***Bước 1***: copy 2 file 6-ConsoleAppGraphics và ConsoleApp\_cpp\_graph vào đường dẫn sau C:\Program Files\dev-Cpp\Templates.

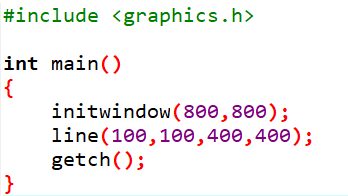
***Bước 2***: copy 2 file graphics và winbgim vào đường dẫn sau C:\Program Files\Dev-Cpp\MinGW64\x86\_64-w64-mingw32\include.

***Bước 3***: copy file libbgi.a vào đường dẫn sau C:\Program File\Dev-Cpp\MinGW6x86\_64-w64-mingw32\lib.

***Bước 4***: Mở Dev-C++ lên, chọn New Project – Console Graphics Application, đặt tên và lưu project.

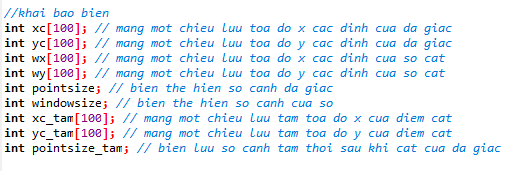
***Bước 5***: vào Tools - Compiler Option, chọn TDM-GCC 4.9.2 32 bit Release.

***Bước 6***: tiến hành code thử nghiệm đoạn code (Hình 2.1) vào project đã lưu.



Hình 2.1. Đoạn code demo cài đặt thư viện graphics.h vào Dev-C++

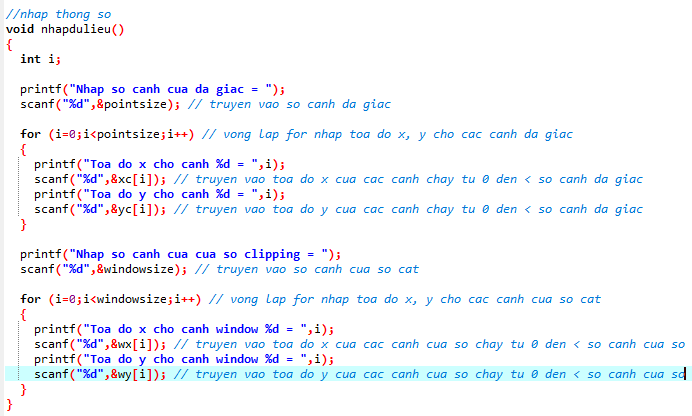
## 2.2. Khai báo biến

****

Hình 2.2 Khai báo các biến cho chương trình.

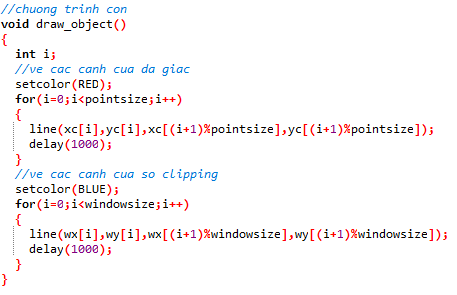
## 2.3. Nhập dữ liệu

Dữ liệu đầu vào bao gồm 2 tham số: (1) cửa sổ cắt và (2) đa giác cần cắt. Dữ liệu này được nhập từ bàn phím thông qua giao diện hỏi và đáp như hình 3.3.1. Danh sách các đỉnh của đa giác được lưu trữ trong mảng một chiều có tên xc và yc, các đỉnh của cửa sổ cắt được lưu trữ trong danh sách có tên wx và wy.

****

Hình 2.3. Các bước nhập dữ liệu từ bàn phím.

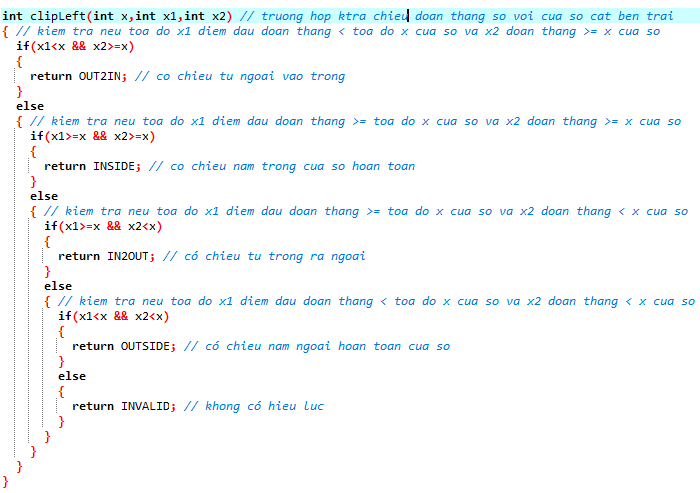
## 2.4. Vẽ cửa sổ cắt và đa giác

****

Hình 2.4 Các bước vẽ các cạnh đa giác và của sổ cắt.

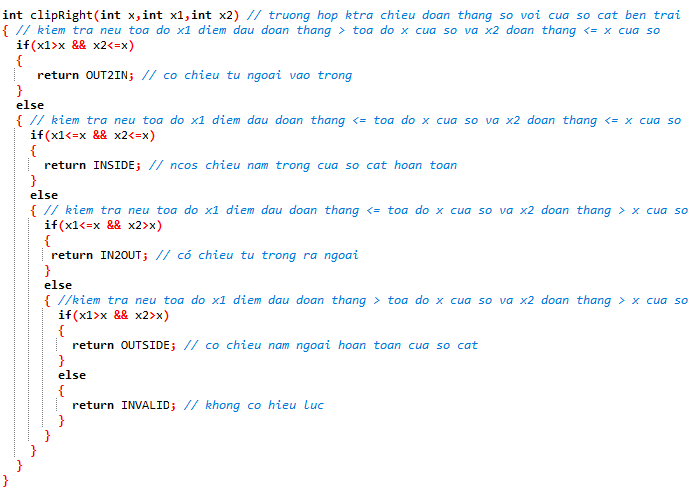
## 2.5. Cài đặt thuật toán xém tỉa đa giác

Đầu tiên kiểm tra chiều cạnh đa giác so với cửa sổ cắt bên trái.



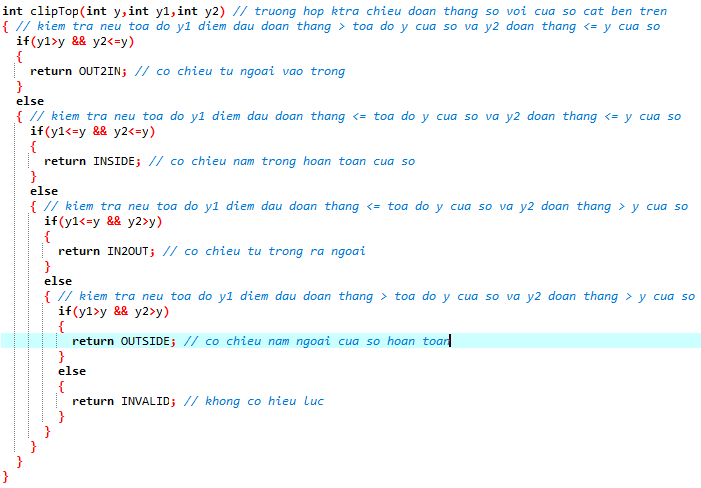
Hình 2.5. Các bước kiểm tra cạnh đa giác so với cửa sổ cắt bên trái.

Kiểm tra chiều các cạnh đa giác so với cửa sổ cắt bên phải.

****

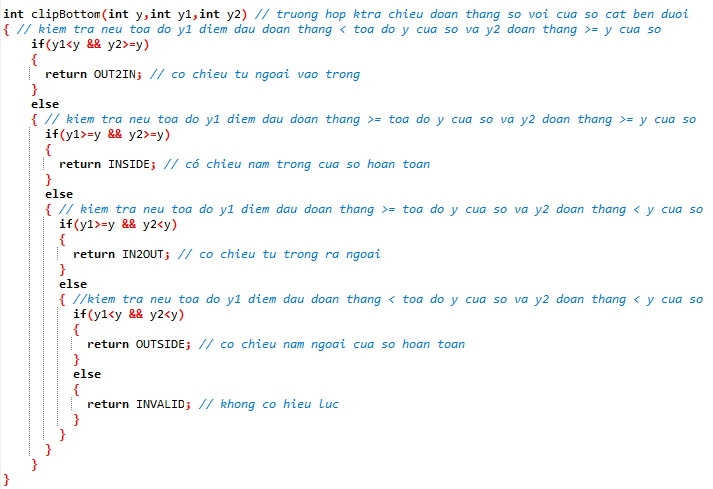
Hình 2.6. Các bước kiểm tra cạnh đa giác so với cửa sổ cắt bên phải.

Kiểm tra chiều các cạnh đa giác so với cửa sổ cắt bên trên.

****

Hình 2.7. Các bước kiểm tra cạnh đa giác so với cửa sổ cắt bên trên.

Kiểm tra chiều các cạnh đa giác so với cửa sổ cắt bên dưới

****

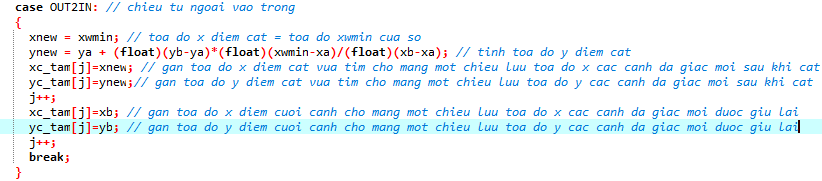
Hình 2.8. Các bước kiểm tra cạnh đa giác so với cửa sổ cắt bên dưới.

Bắt đầu xén tia đa giác theo các trường hợp sau:

### 2.5.1. Trường hợp cắt cạnh bên trái

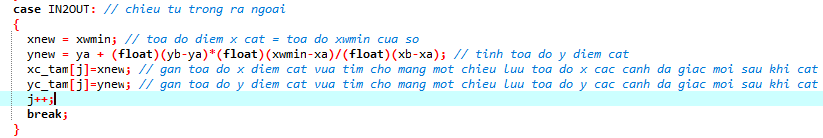
Có 4 trường hợp ứng với vị trí của cạnh đa giác so với cạnh trái của cửa sổ cắt:

Từ ngoài vào trong

****

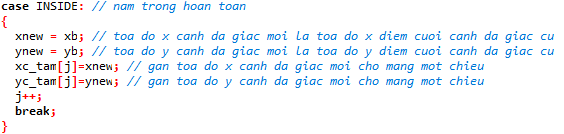
Hình 2.9. Chiều từ ngoài vào trong.

Từ trong ra ngoài

****

Hình 2.10. Chiều từ trong ra ngoài.

Trong hoàn toàn

****

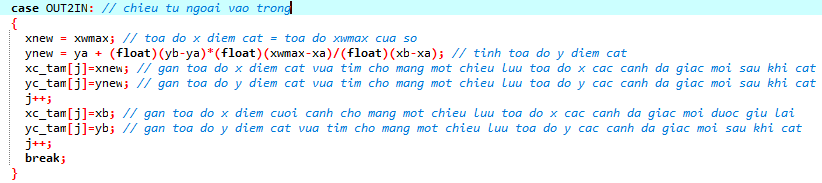
Hình 2.11. Nằm trong hoàn toàn.

Ngoài hoàn toàn không tiến hành cắt mà loại bỏ luôn.

### 2.5.2. Trường hợp cắt bên phải

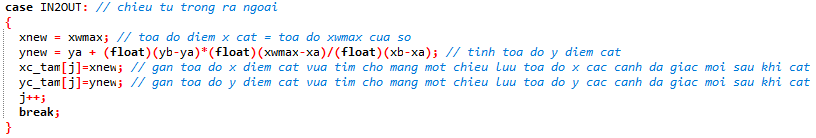
Có 4 trường hợp ứng với vị trí của cạnh đa giác so với cạnh phải của cửa sổ cắt:

Từ ngoài vào trong

****

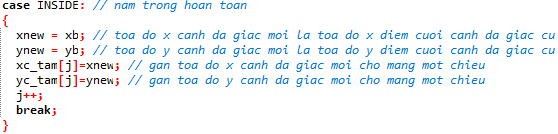
Hình 2.12. Chiều từ ngoài vào trong.

Từ trong ra ngoài

****

Hình 2.13. Chiều từ trong ra ngoài.

Trong hoàn toàn

****

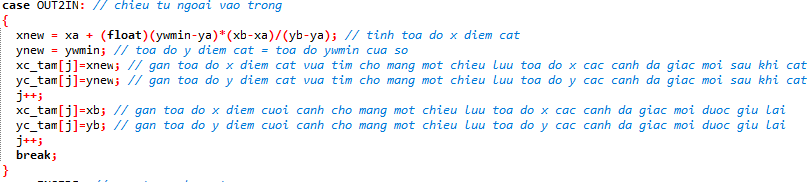
Hình 2.14. Nằm trong hoàn toàn.

Ngoài hoàn toàn không tiến hành cắt mà loại bỏ luôn.

### 2.5.3. Trường hợp cắt bên trên

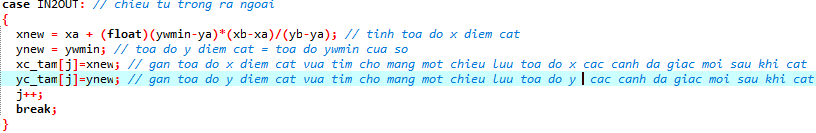
Có 4 trường hợp ứng với vị trí của cạnh đa giác so với cạnh trên của cửa sổ cắt:

Từ ngoài vào trong

****

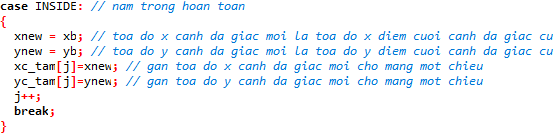
Hình 2.15. Chiều từ ngoài vào trong.

Từ trong ra ngoài

****

Hình 2.16. Chiều từ trong ra ngoài.

Trong hoàn toàn

****

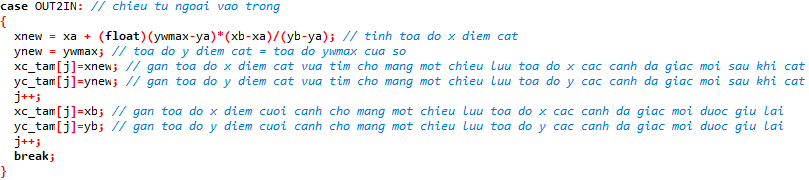
Hình 2.17. Nằm trong hoàn toàn.

Ngoài hoàn toàn không tiến hành cắt mà loại bỏ luôn.

### 2.5.4. Trường hợp cắt bên dưới

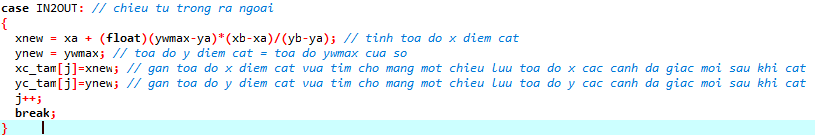
Có 4 trường hợp ứng với vị trí của cạnh đa giác so với cạnh dưới của cửa sổ cắt:

Từ ngoài vào trong

****

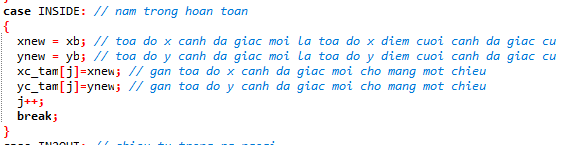
Hình 2.18. Chiều từ ngoài vào trong.

Từ trong ra ngoài

****

Hình 2.19. Chiều từ trong ra ngoài.

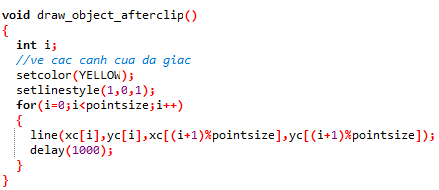
Trong hoàn toàn

****

Hình 2.19. Nằm trong hoàn toàn.

Ngoài hoàn toàn không tiến hành cắt mà loại bỏ luôn.

### 2.5.5. Vẽ đa giác mới sau khi cắt

****

Hình 2.20. Hàm vẽ đa giác mới sau khi xén tỉa.

# 3. KẾT QUẢ

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

Hình 3.1. Nhập dữ liệu bằng tay và kết quả hình ảnh đa giác

|  |  |  |  |
| --- | --- | --- | --- |
| a | B | C | d |

Hình 3.2. Các trường hợp xén tỉar đa giác

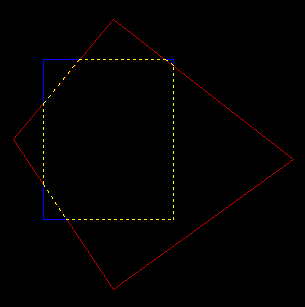
Trong đó đường màu xanh là cửa sổ cắt, đường màu vàng là đa giác.

Hình a có đường màu đỏ là đường xén tỉa bên trái của cửa sổ và đa giác.

Hình b có đường màu đỏ là đường xén tỉa bên phải của cửa sổ cắt và đa giác.

Hình c có đường màu đỏ là đường xén tỉa bên trên của cửa sổ cắt và đa giác.

Hình d có đường màu đỏ là đường xén tỉa bên dưới của cửa sổ cắt và đa giác.



Hình 3.3. Kết quả xén đa giác sau khi cắt xén.

# 4. KẾT LUẬN

Thuật toán xén tỉa đa giác sử dụng thuật toán Sutherland – Hogman được cài đặt theo yêu cầu mức một của đề tài thực tập cơ sở. Việc nhập dữ liệu được thông qua bàn phím. Các trường hợp xén tỉa đa giác đều được mô hình hóa và cài đặt, trong khuôn khổ báo cáo này, hạn chế của thuật toán Sutherland – Hogman đối với đa giác không lồi có đỉnh lõm nằm ngoài cửa sổ xén không thực hiện được.