# ICCAD 2019 CAD

# Contest

# Problem E: Rectilinear Polygon Operations for Physical Design

## Contents


0. Announcement……....P2

I. Introduction ……...….P3

II. Problem Description …….P3

III. Example of Input/Output Files ……P5

IV. Language ……..P6

V. Platform ……...……..P6

VI. Testcases ……...P6

VII. Evaluation ……P6

VIII. FAQ ……P8

# 0. Announcement

### October

- 2019-10--

### August

- 2019-08--

### July

- 2019-07--

### June

- 2019-06-21- The CHECKER of ProblemE is updated again.
- 2019-06-21- The FAQ of ProblemE is updated again.
- 2019-06-21- The CHECKER of ProblemE is updated.
- 2019-06-21- The FAQ of ProblemE is updated.
- 2019-06-19- ProblemE description and FAQ are updated.
- 2019-06-18- The CaseResult of ProblemE is updated.
- 2019-06-18- The FAQ of ProblemE is updated.
- 2019-06-14- The FAQ of ProblemE is updated.
- 2019-06-11- The CaseResult of ProblemE is removed temporarily.
- 2019-06-11- The FAQ of ProblemE is updated.
- 2019-06-04- The CaseResult of ProblemE is updated.

### May

- 2019-05-22- The FAQ of ProblemE is updated.
- 2019-05-08- ProblemE description and FAQ are updated.
- 2019-05-03- The FAQ of ProblemE is updated.

### April

- 2019-04-24- The FAQ of ProblemE is updated.
- 2019-04-23- The OpenCase_1 and OpenCase_2 of Problem E are updated.

### March

- 2019-03-18- The FAQ of ProblemE is updated.
- 2019-03-13- ProblemE is updated.
- 2019-03-07- ProblemE is updated.
- 2019-03-06- ProblemE is updated.

### Fabruary

- 2019-02-27- ProblemE is updated.
- 2019-02-01- ProblemE announced.

# Rectilinear Polygon Operations for Physical Design

## I. Introduction

Rectilinear polygon processing is a very common problem in Physical Design because there are many elements in the physical design flow that are represented by rectilinear polygons, such as macro blocks, cell pin shape, standard cell row regions, floorplan channels, ... and so on.

Therefore, in the physical design stage, it is often necessary to perform the merging, clipping, splitting, etc. processing on the rectilinear polygons, and obtain the results for further analysis or application. For example, "merge" operation is used to merge multiple rectilinear polygons connected to handle repeated descriptions of overlaps and reduce the number of rectilinear polygons to improve program execution efficiency. "Clip" operation is used to delete some special areas, and "split" operation is used to convert rectilinear polygons into rectangles to simplify the complexity of the problem.

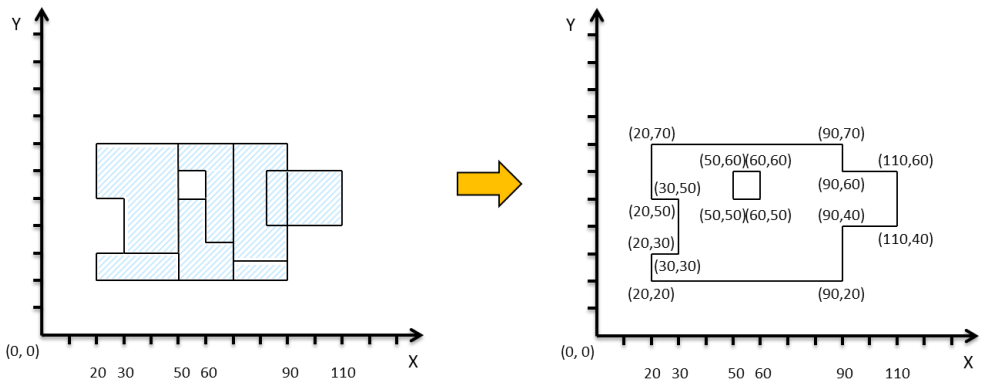
There are some references and methods for the basic processing of rectilinear polygons. But the key is that when applied to physical design flow, the number of polygons is often very large, which makes the program implementation more difficult.

## II. Problem Description

The program must be able to perform the following rectilinear polygon operations, including support the rectilinear polygon with “hole”.

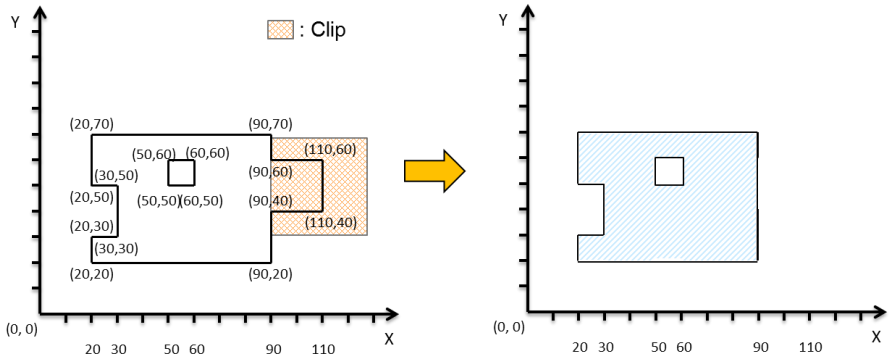
### A. Merge

After the “Merge” operation, all connected rectilinear polygons (including edges) must be merged into a rectilinear polygon, as shown below:



## B. Clip

The “clip” operation deletes the intersection between the original rectilinear polygons and the clip rectilinear polygons, and obtains one or more new rectilinear polygons, as shown below:

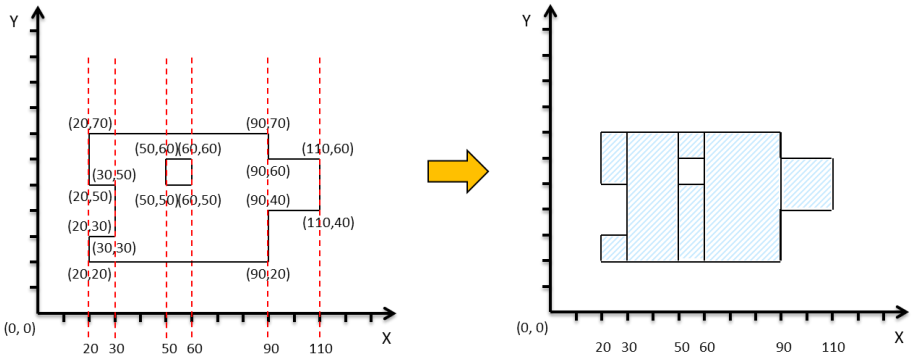



## C. Split

Convert the rectilinear polygon into multiple rectangle representations, and there must be no overlap in the results except the rectangular edges. However, if the collinear of edges between two of result rectangles have same start and end point, it should be removed. The function must contain these three types:

1. split_H: completed by **horizontal** cutting
2. split_V: completed by **vertical** cutting
3. split_O: Horizontal and vertical cuts can be mixed to cut out the **minimum rectangles** for the optimize target

The following is an example of "Split_V":




In testcases, a combination of various processes will be described. For example, “M C SH” means to perform Merge → Clip → Split_H for the input rectilinear polygon.

## III. Example of Input/Output Files

- Input File:

**OPERATION** M1 C1 M2 SV ;

**DATA MERGE** M1 ;
POLYGON 0 0 100 0 100 100 0 100 0 0 ;
POLYGON 100 0 200 0 200 100 100 100 100 0 ;
**END DATA**

**DATA CLIPPER** C1 ;
POLYGON 50 50 150 50 150 150 50 150 50 50 ;
**END DATA**

**DATA MERGE** M2 ;
POLYGON 0 100 200 100 200 200 0 200 0 100 ;
**END DATA**

- Output File:

**RECT** 0 0 50 200 ;
**RECT** 50 0 150 50 ;
**RECT** 50 100 150 200 ;
**RECT** 150 0 200 200 ;

- Description:

The first line of the input file describes the operation requirements, and please follow the order of the operations. The operation definition begins with the keyword "OPERATION" and each operation is separated by a space. In the example, "M1" represents a “merge” operation, and the polygons that need to be processed are described in the "Data M1" section below. “C1” represents a “clip” operation, and the polygons data are described in the “Data C1” section. “M2” continues to merge the polygons described by “DATA M2” base on the operation “M1 C1” result.

The "split" operation does not require the data section, but there are three keywords that represent different ways of cutting: "SV" means completion by vertical cutting, "SH" means completion by horizontal cutting, and "SO" means completion by optimized cutting. In the example, the last operation is to split the previous result by vertical cutting. The important rule

is that each merge operation must contain the previous results and the final operation will always be "split".

The data section of each merge or clip operation begins with the keyword "DATA" and continues with an operation. All of operation shapes are defined as rectilinear polygon points, such as X0 Y0 X1 Y1 and so on, and end with “END DATA”. The description of polygon may be clockwise or counterclockwise, and the last set of coordinates may not be the same as the first set of coordinates. Hole is not described in the test file, so there is no format for the description of the hole. But multiple polygon descriptions may cause holes, please consider when developing the program.

Since the last operation is always "split", the output file should only contain rectangles. Each rectangle should start with the keyword "RECT", and the points are the lower left X, the lower left Y, the upper right X, and the upper right Y sequentially.

## IV. Language

Please implement your program in C or C++. The binary file should be called as “myPolygon”. Please follow the following usage format:

**./myPolygon input_file output_file**

## V. Platform

OS: Linux
Compiler: gcc/g++

## VI. Testcases

5 open testcases can be downloaded
3 non-disclosure testcases

## VII. Evaluation

The score is divided into two stages. When the first stage score is the same, the second stage score is performed. The two-stage score is as follows:

**Stage1:**

The score is 25 for the open testcases and 75 for the non-disclosure testcases. If the program encounters compilation errors, crash (core dump),

runs on the test platform provided by the conference for more than 8 hours, the result incorrect, the score for this testcase will be 0. Additionally, if the last operation is SO, the number of result rectangles are must less than or equal to min(SV, SH), or the result is considered incorrect. Here min(SV,SH) represent an max base for normalizing and it will be calculated by scoring checker at first. When the first stage total score is the same, the second stage will be scored.

**Stage2:**

The second stage score includes the CPU time and the number of rectangles of the final result, with weights of 70% and 30%, respectively.

Suppose the testcase has m teams completed correctly, $T_i$ means the $i^{th}$ team time performance**,** and the score will be $St_i$. $N_i$ means the $i^{th}$ team number of result rectangles and the score will be $Sn_i$. Additionally, if the last operation is not SO, the CPU time performance will dominate all of score for that testcase.

**Normalize functions**

$$St_i = 1 - \frac{T_i - \min(T)}{\max(T) - \min(T)}$$

Where $T = (T_1, \dots, T_m)$ and $St_i$ is now $i^{th}$ team normalized CPU time score

$$Sn_i = 1 - \frac{N_i - \min(N)}{\min(SV, SH) - min(N)}$$

Where $N = (N_1, \dots, N_m)$ and $Sn_i$ is now $i^{th}$ team normalized number of result rectangle score, and $Sn_i$ is calculated by 0 if $Sn_i$ is less than 0.

**Total score function**

$$S_i = \sum_{C=1}^{5} \left(St_i^C * 70\%\right) + \left(Sn_i^C * 30\%\right)$$

Where C means case and $C = (C_1, \dots, C_5)$; $S = (S_1, \dots, Sm)$ and $S_i$ is now $i^{th}$ team total score

## VIII. FAQ

Q1. 題目描述中提到 "including support the rectilinear polygon with "hole"." 可否舉例說明？

A1. 意思是指，提供的 Testcase 中雖並不包含 hole 的資訊，但經過 merge 的 operation 後，可能會有 hole 的產生。
如題目 II. Problem Descriptio 中 Section A. Merge 的範例圖中，Testcase 所提供的，都會是 none hole polygon shape，但有可能會是 overlap 的。
經過 Merge operation 後所產生的圖形，是有可能帶有 hole 的資訊，意即參賽者要能夠處理 hole 的資訊。

Q2. "Merge" operation 中的圖示不清楚，看不出預期的 output。

A2. 題目中所描述的圖示，左邊圖示是指 Testcase input，右邊圖示是經過 merge operation 後的結果。
該題目並不會有這樣的 output，這邊是給參賽者提示，經過 merge operation 後，所產生的中繼圖形。

Q3. "Clip" operation 中的圖示不清楚，看不出預期的 output。

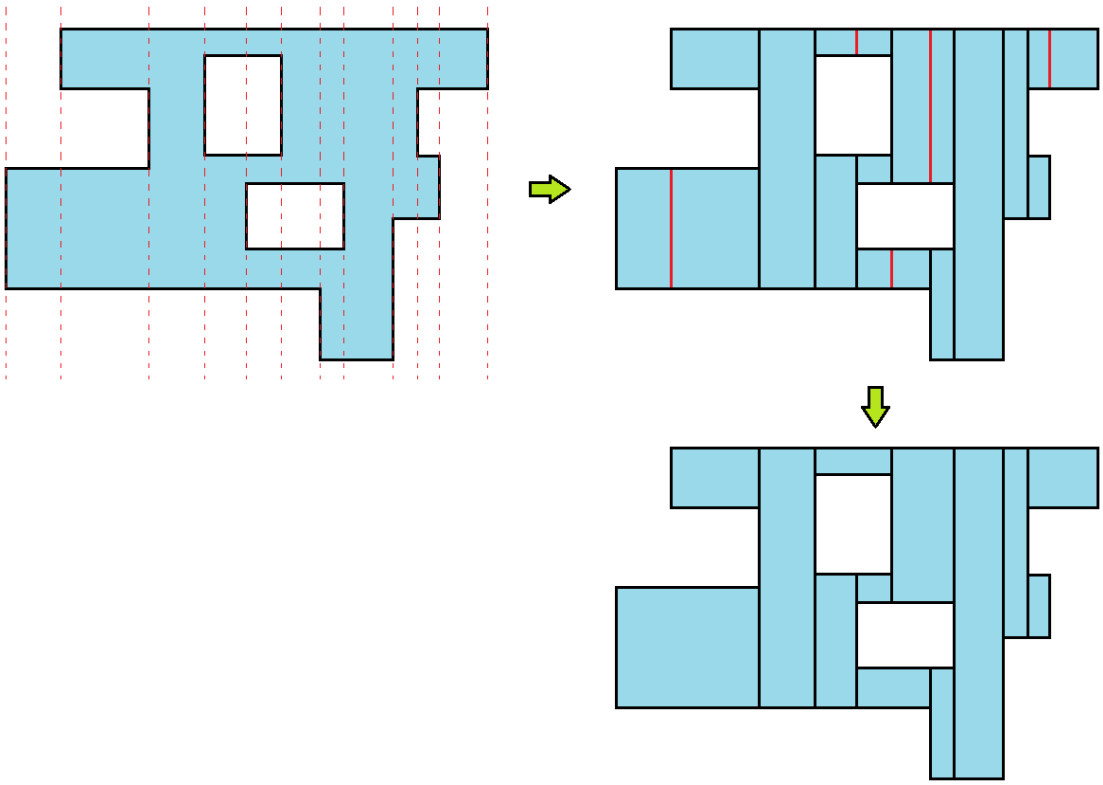
A3. 同上題，題目中所描述的圖示，左邊圖示是指 Testcase input，右邊圖示是經過 clip operation 後的結果。
該題目並不會有這樣的 output，這邊是給參賽者提示，經過 clip operation 後，所產生的中繼圖形。

Q4. "Split" operation 中的圖示不清楚，看不出預期的 output.

A4. 同上題，題目中所描述的圖示，左邊圖示是指 Testcase input，右邊圖示是經過 split operation 後的結果。
該題目所提供的 Testcase 最後一個 operation 皆會有 split operion，只是差別在於 Split_V, Split_H, Split_O。
預期的 output 格式以及範例，描述在 III. Example of input/output Files 中，該圖形只是給參賽者一個視覺化範例。

Q5. Split 的定義是否能再詳細些？Split_V 和 Split_H 是否是根據 "每一個" 轉角座標做 cutting line？

A5. Split_V：代表只以平行 Y 軸的方向做切線，起點應從圖形中存在之最小的 X 座標開始遞增，在遞增過程中。該中繼圖形若遇到任何 90 degree corner，即應對該圖形增加一切線，直到遞增至圖形中最大的 X 軸座標。然而，若存在一切線，使得兩個 Split 之後的 Rectangles，擁有相同的 Y 軸起點及終點之共線時，需將該共線移除。如下圖所示



Split_H：代表只以平行 X 軸的方向做切線，起點應從圖形中存在之最小的 Y 座標開始遞增，在遞增過程中。 該中繼圖形若遇到任何 90 degree corner，即應對該圖形增加一切線，直到遞增至圖形中最大的 Y 軸座標。然而，若存在一切線，使得兩個 Split 之後的 Rectangles，擁有相同的 X 軸起點及終點之共線時，需將該共線移除。

Q6. input file: polygon 的描述是否固定為順時針或逆時針方向順序給定? 或者兩個方向皆有可能？

A6. input file 中的 polygon 的描述，順時針方向與逆時針方向皆有可能。

Q7. output file: 最終的 rectangles 是否須按某種排序？

A7. output file 中的 rectangle 的描述，順時針方向與逆時針方向皆可。

Q8. Will the points describing the polygons in the input file be fixed counterclockwise?

A8. No. It will be clockwise or counterclockwise possibly. And these may both appear in one file.

Q9. Will the points describing the polygons in the input file must be close?

A9. Yes. The polygons will always be closed in the input file.

Q10. If the polygon of operation "merge" does not overlap with target polygon, do I need to store both of them or just keep the target polygon?
A10. You should keep all of intermediate polygons after each operation.

Q11. If the polygon of operation "clip" cuts the target polygon into multiple polygons, do I need to store them all?
A11. Same of above. You should keep all of intermediate polygons after each operation. Therefore, the output rectangles may be born of different target polygons.

Q12. What is the number range of the points describing the polygons in the input file?
A12. The input polygon number range will belong with type 'int64'.

Q13. Whether the number of the points describing the polygons in the input file will be decimal or just integer?
A13. The number of the points describing in the input file will be integer only.

Q14. If the polygon that operation "clip" does not overlap any of the target polygons, will the result be the same as the original target polygon? Or do I need to preserve the "clip" polygon?
A14. If the polygon that operation "clip" does not overlap any of the target polygons, the result will be the same as the original status.

Q15. 是否開放 multi-thread programming？
A15. 原則上，我們開放 multi-thread programming 技巧，但是在計分的時候，會控制 process thread 數量，使得 process 只能以 single thread 執行。

Q16. 如果有同學使用 multi-thread programming 呢？
A16. 基本上，我們還是開放 multi-thread 的 programming 技巧，但由於計分問題，本題意旨並非以 multi-thread 技巧為主軸，因此，在計分上，我們只讓 process running on single thread 的結果為主。

Q17. 承上。如若會扣分的話，就得確實檢查是否有使用 multi-thread programming 了——若未檢查出來，競賽的公平性可能將遭到質疑。
A17. 承上，因為開放了 multi-thread programming，所以即便是用了 multi-thread 技巧，皆不予以扣分，而公平性會用機器去限制 process，只能以 single thread 執行，因此達到公平性。

Q18. Can we use boost c++ libraries ?
A18. Yes. You can use C++ boost library.

Q19. 每個 case 的第一行 OPERATION 後面接的 M1 C1... ，與下面用 DATA 包住的各個 operation 給的順序會相同嗎？
A19. 順序不一定相同，因此，才會在每筆 DATA 前加上該筆資料的標註。

Q20. input file 有規定都是整數，請問會有負整數嗎？
A20. Input file 的 Polygon 皆以 int64 為範圍，因此，是有可能出現負整數的。

Q21. 第一階段評分的方式，跑八小時是指單筆測資還是所有測資？
A21. 係指單個 Case，每一個 Case 皆有八小時的時間範圍。

Q22. 第二階段評分的公式，Sni 的部分，分母為 min(SV,SH)是代表什麼意思？
A22. 第二階段評分 min(SV, SH) 係指，若最後的 Operation 出現 SO 時，則在程式驗證前，我們會以 SV 以及 SH 做為該 Case 最後的 Operation，先行計算出正確且固定的矩形輸出數量，最後再以上述計算之結果，從 SV 以及 SH 的矩形輸出數量中，取最小值為基準。
然而，倘若某 Case 中沒有出現 SO 時，代表該 Case 的輸出是有正確且固定的矩形數量，因此，將不會以該公式做為評分標準。

Q23. 為確保對題目的認知以及輸出的正確性，請問是否會提供 checker？
A23. 初步我們會先提供 Open case 1 & 2 的 output rectangles 正確數量，以供參賽者做初步驗證。之後，我們會再提供 Open case 1 & 2 的 checker 以供參賽者做後續的驗證。

Q24. 請問 input file 中的 polygon 都是只有 4 個頂點還是有可能會有 4 個頂點以上呢？
A24. Input file 當中的 Polygon 是有可能出現 4 個頂點以上的。

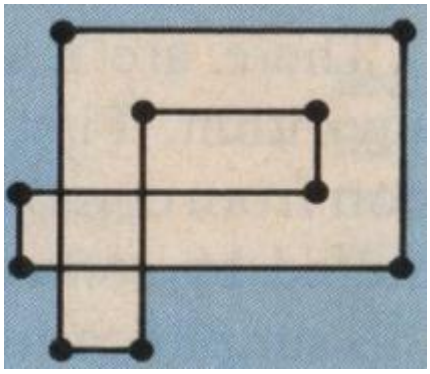
Q25. 是否能使用 open CV 的套件呢？
A25. 可以。

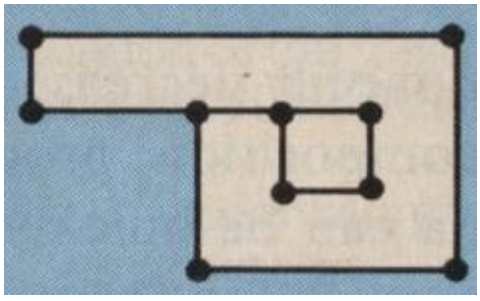
Q26. 於 Q22 的回覆(A22)中, SO 以 min(SV, SH)為基準。
這和題目描述 Sec. II-C 不太相同。
"split_O: Horizontal and vertical cuts can be mixed to cut out the minimum rectangles

for the optimize target"
原描述是混合 V 和 H 的 split 使得 rectangle 數目最小。
可否再次確認 SO 的評比方式？
A26. 首先，split_O 係指參賽者可以混合直切以及橫切的方式使得 rectangle 數目最小，且由上述說明可以推論 split_O 的結果數量應該小於或等於 split_H 及 split_V 的結果數量中較小者。
因此，在計分上，若該題最後的 operation 為 split_O 時，則參賽隊伍的結果數量需小於或等於 min(SV, SH)，否則該題會視為不正確，以 0 分計算, 此規則會再加入計分描述當中。
此處的 min(SV, SH) 係指該題最後以 SV 的結果數量以及 SH 的結果數量中，兩者取最小的結果數量, 以作為 normalize 的最大基準數。
最後，若該題最後的 operation 為 SV 或 SH 時 (非 SO)，則在計分方面，不會進入數量評分公式，該題所有的分數皆以 CPU time performance 為主。

Q27. 請問是否圖形都只會給直線四邊形呢？
還是會給多邊形，像是 20 邊形之類的？
A27. 該問題已經描述在題目當中，"Rectilinear Polygon" 即代表 Polygon 為多邊形，且每個 corner 皆為 90 度，因此，是有可能會出現多邊形的圖形，但每個轉角皆一定會是 90 度。

Q28. 請問 input data 有可能會有以下這兩種自己的 edge 互相重疊或交叉的情況嗎?





A28. Edge 重疊是有可能會出現的，但交叉的部分則不會出現。以 Edge 重疊的那張圖為例，中間部分應為 Hole，而不是填滿的實心。

Q29. 看到文檔說明後面的 Q&A 有說到，會先提供 Open case 1 & 2 的 output rectangles 正確數量， 以供參賽者做初步驗證。之後，會再提供 Open case 1 & 2 的 checker。不知道有沒有可能在近期公佈呢？時間也已經邁入六月，離繳交期限也不遠了，想趕快測試看看自己程式的正確性以作修正。
A29. Open Case 的數量已公布。Checker 的部分，經由與主辦單位確認後，預計在 6 / 21 前釋出 Checker 以利參賽隊伍做驗證。

Q30. 請問除了提供最後切出來的矩型數量之外，是否還會另外提供結果中所有矩型的座標點以供驗證？
A30. 會提供 Cheker 以利參賽隊伍做驗證，但不會提供完整座標點。

Q31. 關於日前公布的 Opencase 1 & 2 output rectangles 正確數量 (case1: 459, case2: 573780)，因為我們同學有好幾組隊伍用不同方法分頭做，都做出一模一樣的數量(case1: 449, case2: 586132)，跟正確數量差了一點，有點擔心可能是官方答案有問題（感覺機率不大？），不知道能不能提供 Open case 1 & 2 的 checker 讓我們看看問題出在哪裡呢？
A31. 關於該問題，我們會再做確認，若確認是答案有問題，將會立即修正以供參賽隊伍參考。
由於內部傳遞的資料有誤，我們已重新更正答案，以供參賽隊伍做參考。

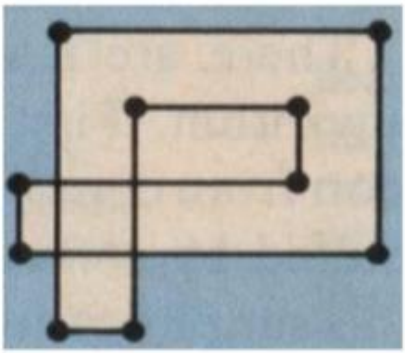
Q32. 我想詢問關於 Final Project 中的 OpenCase1，input 的相關問題。如附圖中看到，在網站中的規定中，說明 input 不會含有 hole，但重疊是可能發生的，但 Q28 下方的敘述又說，以「Edge 重疊的那張圖為例，中間部分應為 Hole」，我想請問他說的 Edge 重疊是指上面還下面那張圖？以及 input 到底會不會包含 Hole？另外其亦說 Edge 不會有交叉的情形，但在 OpenCase1 中有一個 input (如下圖)，出現了 Edge 交叉的情形，我想請問要如何解讀這個圖形？麻煩您了。

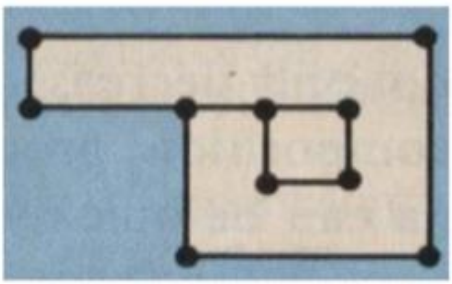
## VIII. FAQ

Q1. 題目描述中提到 "including support the rectilinear polygon with "hole"." 可否舉例說明？

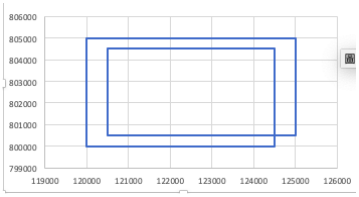
A1. 意思是指，提供的 Testcase 中雖並不包含 hole 的資訊，但經過 merge 的 operation 後，可能會有 hole 的產生。
如題目 II. Problem Descriptio 中 Section A. Merge 的範例圖中，Testcase 所提供的，都會是 none hole polygon shape，但有可能會是 overlap 的。
經過 Merge operation 後所產生的圖形，是有可能帶有 hole 的資訊，意即參賽者要能夠處理 hole 的資訊。

Q28. 請問 input data 有可能會有以下這兩種自己的 edge 互相重疊或交叉的情況嗎?





A28. Edge 重疊是有可能會出現的，但交叉的部分則不會出現。以 Edge 重疊的那張圖為例，中間部分應為 Hole，而不是填滿的實心。



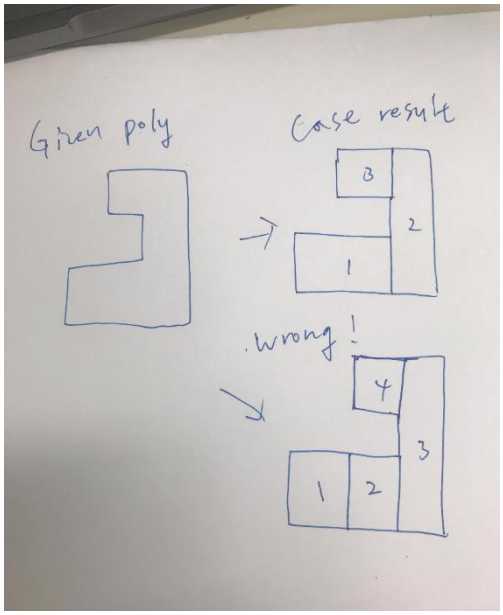

```
POLYGON 120000 800000 120000 805000 125000 805000 125000 800500 124500 800500 124500
804500 120500 804500 120500 800500 124500 800500 124500 800000 120000 800000 ;
```

A32. 第一個問題，"Edge 重疊是指上面還下面那張圖？"係指 Q28 所提出的問題中，下面的那張圖；Q28 上面那張圖中，Edge 互相交叉 (Cross edge) 的情況，在所有 Case 中皆不會出現。
第二個問題，"input 到底會不會包含 Hole"，題目說明中，不含有 Hole 的資訊，係指不會使用 Contour 或是 Hole 的表示方式，額外去定義該 Polygon 的內

部資訊，但並非說明該 Polygon 本身不會隱含有 Hole 的資訊。
第三個問題，Open case 1 input 所出現的，是 Vertices 重疊 (touching) 的情況，而非 "Cross edge" 的情況，因此是有機會出現的，在 Q28 提出的問題中，已說明。
另外，在所有的 Case 中，單一 Polygon 的描述中，Vertex 與 Edge 之間重疊 (touching)的情況是有可能出現的，但不會出現交叉的情況。

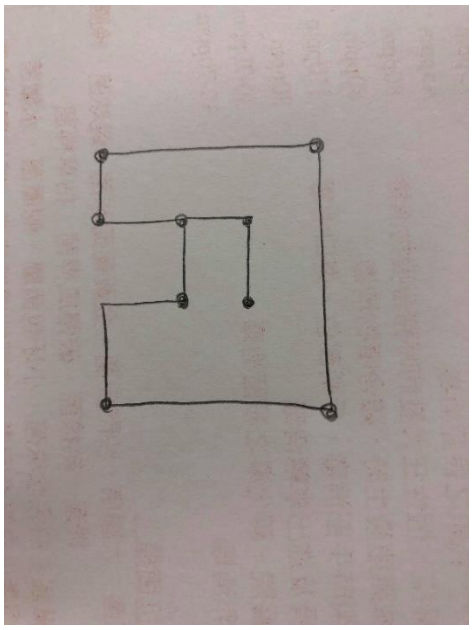
Q33. 若最終結果圖形包含數個獨立多邊形，根據 FAQ5 的回覆，會根據所有出現的轉角座標做水平線（或垂直線）切割。但根據 FAQ31 所列的參考答案，似乎是每個獨立多邊形是分開處理的（意即各自做水平或垂直切割），與 FAQ5 的回答不太一致，想再次確認 split_H 和 split_V 的定義。
A33. 由於當時在回覆 FAQ 5. 時，Document 轉檔上發生問題，導致參賽者看不到題目的圖案，因此，誤以為參賽者沒看懂題目圖形，才會使用題目上的圖形座標當作做回覆解釋，正確的結果應為 "針對每個獨立多邊形做分開處理"。這部分會再請主辦單位在 FAQ Q5 的地方，補充解釋之。

Q34. 根據 FAQ33 以及 FAQ5 的說明是每個 polygon 獨立 split，當遇到 90 度轉角時，將 polygon 以轉角座標切開，但還有一點點模糊之處，請見圖中的例子。
左邊的是一個 polygon，大會提供的答案是以右上圖中的方式 split，並非右下的方式。
簡單來說就是，若是 SV mode，切割線只有垂直線（平行 Y 軸），並且不會有兩塊相連的 Rectangle 可以以水平方式（相同 ymin ymax）互相合併（1、2 號）。
請確認是否右上才是正確結果，並請幫忙將題目描述對應修正。




A34. 已在 FAQ Q5 補充解釋。

Q35. 想請問 input 中是否會有如附圖所顯示的情況發生（vertex 重疊，在多邊形內卻只有一條邊）？在這種情況下，這些 edge 也算是多邊形的一條邊嗎？那又會有可能怎麼分割呢？



A35. 在 Input file 中，單一給定的 Polygon，不會出現附圖所顯示的情況。但如果經由參賽者 Merge operation 後所產生出該圖型時，參賽者應將 Polygon 內的這些多餘的 Vertex 移除掉。當移除上述 Vertex 後，切割問題即如題目所示。

Q36. 請問是直接把 compile 好的 binary 執行檔，放在我們自己帳號的主資料夾底下，名字取做 myPolygon（Problem E 文件中說的），這樣就可以了嗎？還是有需要附上程式碼跟 Makefile 讓你們來 compile 呢？

A36. 參賽者提供 Binary file 以及 Readme 即可。

Q37. 你好，剛剛我們測試 problem E 的 checker 時，發現有 permission denied 的警告，導致無法正常執行 checker，我們都有按照步驟執行，可是還是無法正常使用。

A.37 已請主辦單位更新文件。若依然有問題，請附上顯示的 Messages。