ICCAD 2019 CAD

Contest

Problem E: Rectilinear Polygon Operations for Physical Design

Contents

0. Announcement

October

- 2019-10--

August

- 2019-08-28- The FAQ of ProblemE is updated.
- 2019-08-27- The FAQ of ProblemE is updated.
- 2019-08-27- The Open Case 4 of ProblemE is updated.
- 2019-08-26- The FAQ of ProblemE is updated.
- 2019-08-23- The Open Case 4 of ProblemE is updated.
- 2019-08-23- ProblemE description and FAQ are updated.
- 2019-08-22- ProblemE description and FAQ are updated.
- 2019-08-22- The FAQ of ProblemE is updated.
- 2019-08-18- The FAQ and Beta Report of ProblemE are updated.
- 2019-08-15- The FAQ of ProblemE is updated.
- 2019-08-12- The FAQ and Beta Report of ProblemE are updated.
- 2019-08-01- The FAQ of ProblemE is updated.

July

- 2019-07-31- The OpenCase_5 and FAQ of ProblemE are updated.
- 2019-07-26- The OpenCase_4 and FAQ of ProblemE are updated.
- 2019-07-26- The CaseResult and OpenCase_5 of ProblemE are updated.
- 2019-07-25- The FAQ of ProblemE is updated.
- 2019-07-25- The OpenCase_4 of ProblemE is updated again.
- 2019-07-25- The CaseResult and OpenCase_4 of ProblemE are updated.
- 2019-07-18- The FAQ of ProblemE is updated once again.
- 2019-07-18- The FAQ of ProblemE is updated again.
- 2019-07-18- The FAQ of ProblemE is updated.
- 2019-07-11- The FAQ of ProblemE is updated.
- 2019-07-10- The Alpha Report of ProblemE is updated.
- 2019-07-10- The FAQ of ProblemE is updated.

June

- 2019-06-28- The FAQ of ProblemE is updated.
- 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.

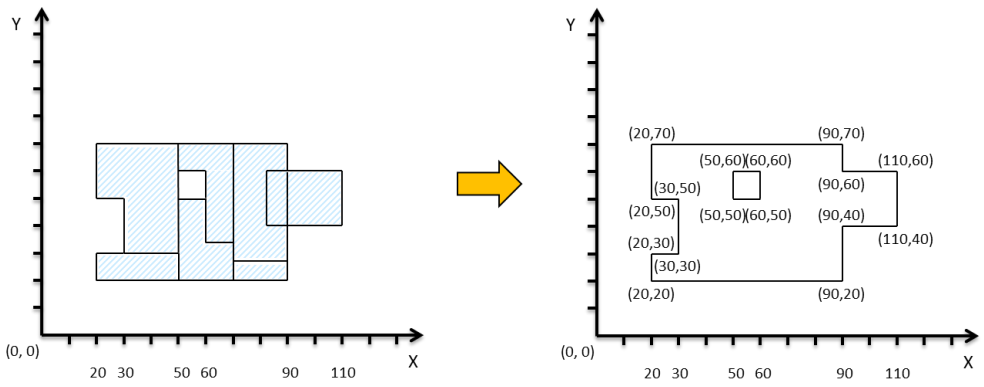
There are some references and methods for the basic processing of rectilinear polygons. But the key is that when applied to physical design flow, the number of polygons is often very large, which makes the program implementation more difficult.

II. Problem Description

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

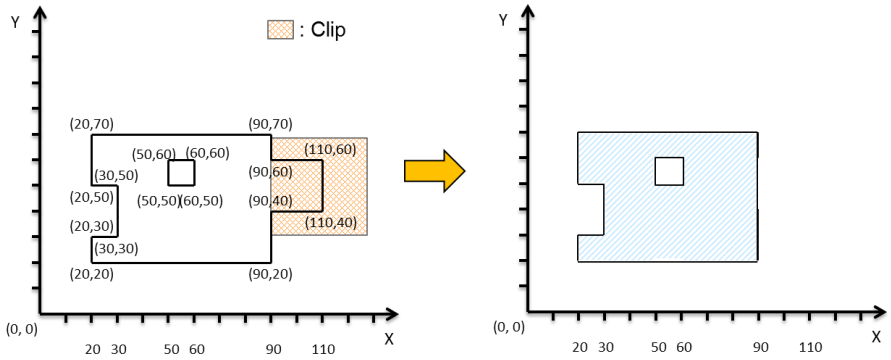
A. Merge

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:

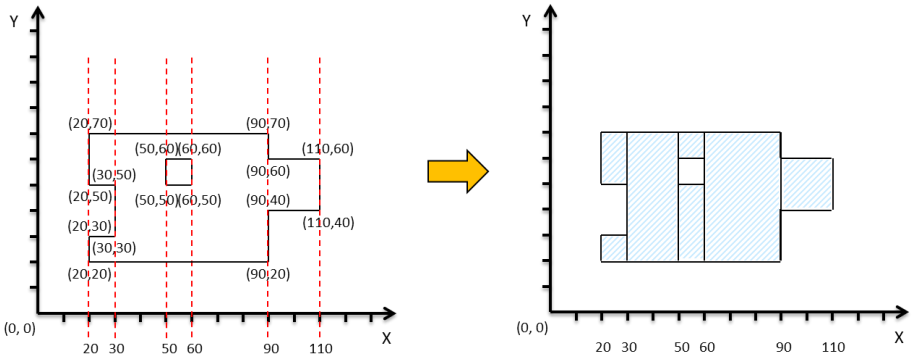


C. Split

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.

The score of each case shown below.

- Open Case 1: 1 points
- Open Case 2: 4 points
- Open Case 3: 5 points
- Open Case 4: 8 points
- Open Case 5: 7 points
- Non-disclosure Case 1: 20 points
- Non-disclosure Case 2: 25 points
- Non-disclosure Case 3: 30 points

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 , … , C_5); S = (S_1, ... , Sm) and S_i is now i^{th} team total score

VIII. Alpha Report

	Rank 1	Rank 2	Rank 3	Rank 4	Rank 5
Case 1 - Result	Pass	Pass	Pass	Pass	Pass
Case 1 - Time	0.05 sec.	0.16 sec.	0.11 sec.	0.10 sec.	2.31 sec.
Case 2 - Result	Pass	Pass	Pass	Pass	Pass
Case 2 - Time	17.83 sec.	42.51 sec.	111.93 sec.	20.76 sec.	1014.18 sec.
Case 3 - Result	Pass	Pass	Pass	Pass	Pass
Case 3 - Time	272.38 sec.	961.08 sec.	1437.43 sec.	12,826.96 sec.	25,307.17 sec.
Case 4 - Result	Failed - No Output	Failed - Error	Failed - Error	Failed - No Output	Failed - No Output
Case 4 - Time	x	x	x	x	x
Note					

IX. Beta Report

Rank	1	2	3	4	5
Open Case 1	PASS	PASS	PASS	PASS	PASS
TIME	0.05 sec.	0.06 sec.	0.06 sec.	200.06 sec.	0.12 sec.
Note					
Open Case 2	PASS	PASS	PASS	PASS	PASS
TIME	8.66 sec.	12.98 sec.	26.12 sec.	211.61 sec.	25.75 sec.
Note					
Open Case 3	PASS	PASS	PASS	PASS	PASS
TIME	11.93 sec.	34.34 sec.	122.63 sec.	221.09 sec.	748.20 sec.
Note					
Open Case 4	Failed	PASS	x	x	x
TIME	0.01 sec.	0.01 sec.	x	x	x
Note			Too Huge Memory	Segmentation fault	Segmentation fault
Open Case 5	PASS	x	x	x	x
TIME	1.39 sec.	x	x	x	x
Note		Segmentation fault	Segmentation fault	Segmentation fault	Segmentation fault
Close Case 1	x	x	PASS	PASS	PASS
TIME	x	x	664.37 sec.	366.16 sec.	13035.14 sec.
Note	No output	Segmentation fault			
Close Case 2	Failed	PASS	x	x	x
TIME	0.21 sec.	0.01 sec.	x	x	x
Note			Segmentation fault	Segmentation fault	Segmentation fault
Close Case 3	PASS	x	x	x	x
TIME	2.79 sec.	x	x	x	x
Note	SO Result: 138,000	Segmentation fault	Segmentation fault	Segmentation fault	Segmentation fault
Stage 1 Score	47	43	30	30	30

X. 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。

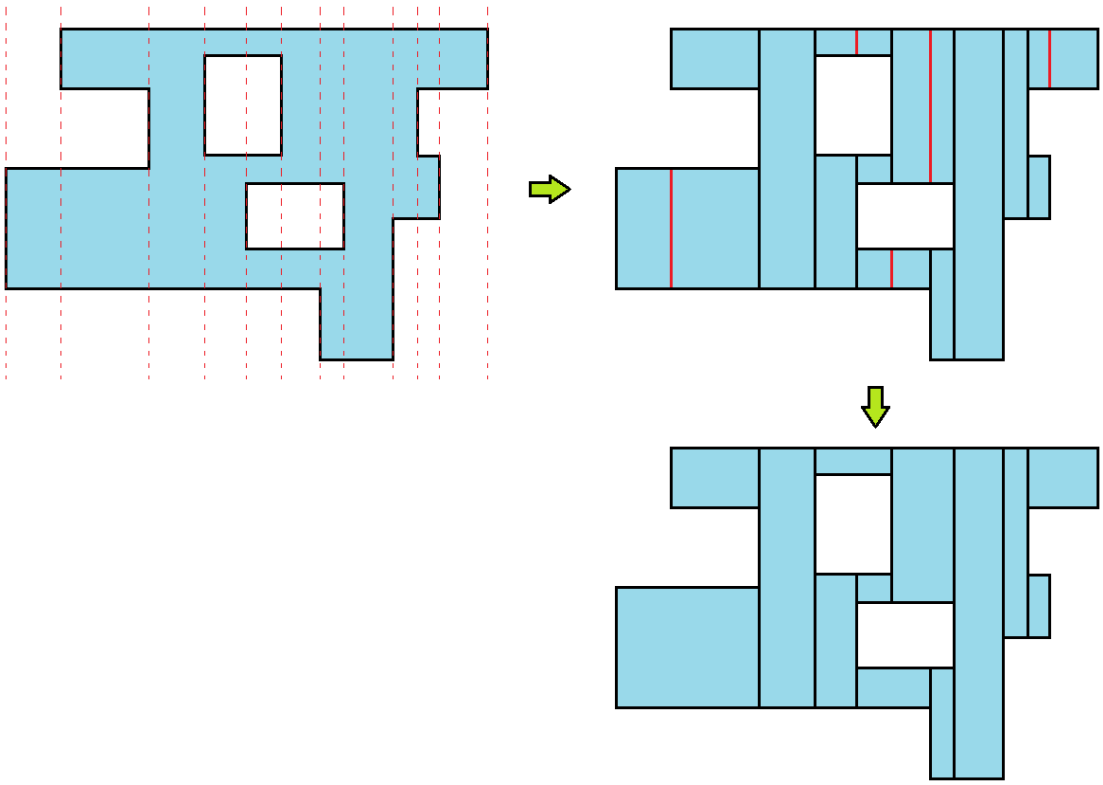
A3. 同上題，題目中所描述的圖示，左邊圖示是指 Testcase input，右邊圖示是經過 clip operation 後的結果。
該題目並不會有這樣的 output，這邊是給參賽者提示，經過 clip operation 後，所產生的中繼圖形。

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。

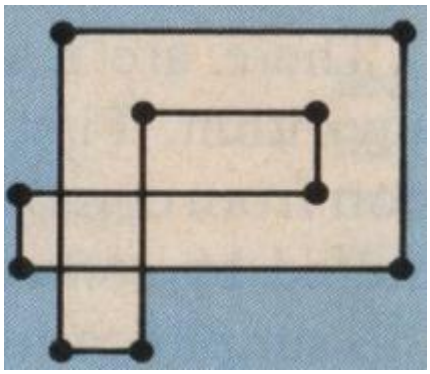
Q25. 是否能使用 open CV 的套件呢？
A25. 可以。

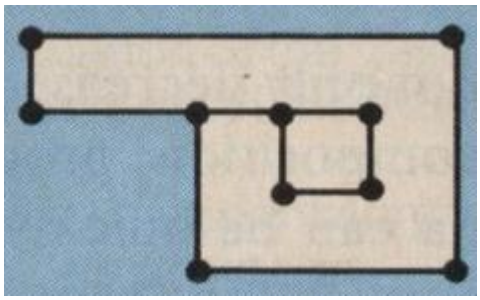
Q26. 於 Q22 的回覆(A22)中, SO 以 min(SV, SH)為基準。
這和題目描述 Sec. II-C 不太相同。

"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。

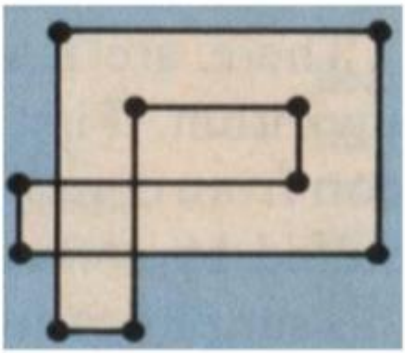
Q32. 我想詢問關於 Final Project 中的 OpenCase1，input 的相關問題。如附圖中看到，在網站中的規定中，說明 input 不會含有 hole，但重疊是可能發生的，但 Q28 下方的敘述又說，以「Edge 重疊的那張圖為例，中間部分應為 Hole」，我想請問他說的 Edge 重疊是指上面還下面那張圖？以及 input 到底會不會包含 Hole？另外其亦說 Edge 不會有交叉的情形，但在 OpenCase1 中有一個 input (如下圖)，出現了 Edge 交叉的情形，我想請問要如何解讀這個圖形？麻煩您了。

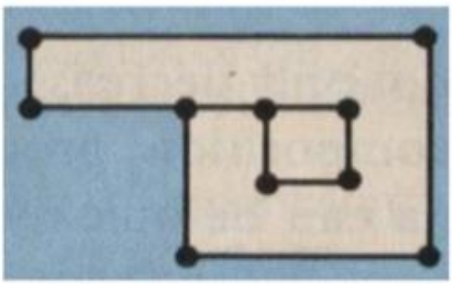
VIII. FAQ

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

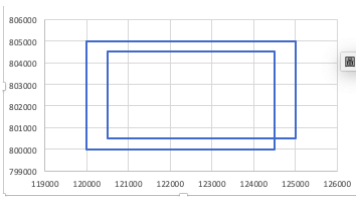
A1. 意思是指，提供的 Testcase 中雖並不包含 hole 的資訊，但經過 merge 的 operation 後，可能會有 hole 的產生。
如題目 II. Problem Descriptio 中 Section A. Merge 的範例圖中，Testcase 所提供的，都會是 none hole polygon shape，但有可能會是 overlap 的。
經過 Merge operation 後所產生的圖形，是有可能帶有 hole 的資訊，意即參賽者要能夠處理 hole 的資訊。

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 ;
```

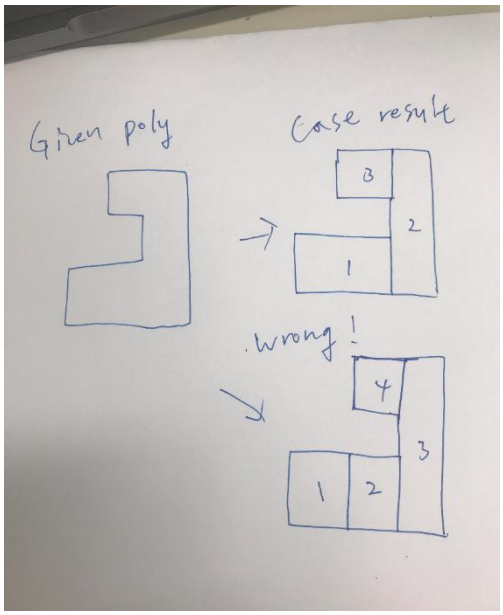
A32. 第一個問題，"Edge 重疊是指上面還下面那張圖？"係指 Q28 所提出的問題中，下面的那張圖；Q28 上面那張圖中，Edge 互相交叉 (Cross edge) 的情況，在所有 Case 中皆不會出現。

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。

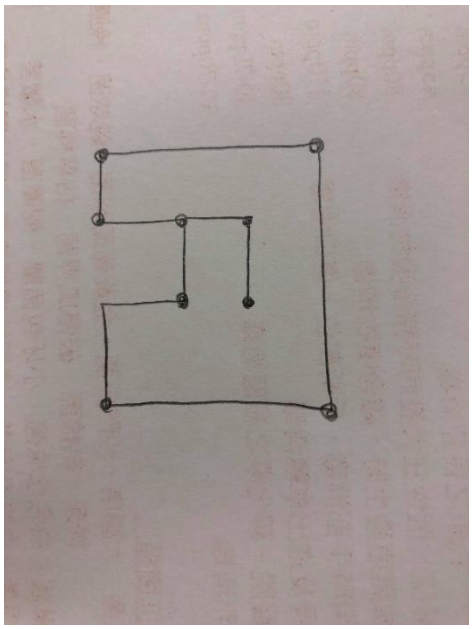
Q34. 根據 FAQ33 以及 FAQ5 的說明是每個 polygon 獨立 split，當遇到 90 度轉角時，將 polygon 以轉角座標切開，但還有一點點模糊之處，請見圖中的例子。
左邊的是一個 polygon，大會提供的答案是以右上圖中的方式 split，並非右下的方式。
簡單來說就是，若是 SV mode，切割線只有垂直線（平行 Y 軸），並且不會有兩塊相連的 Rectangle 可以以水平方式（相同 ymin ymax）互相合併（1、2 號）。

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。

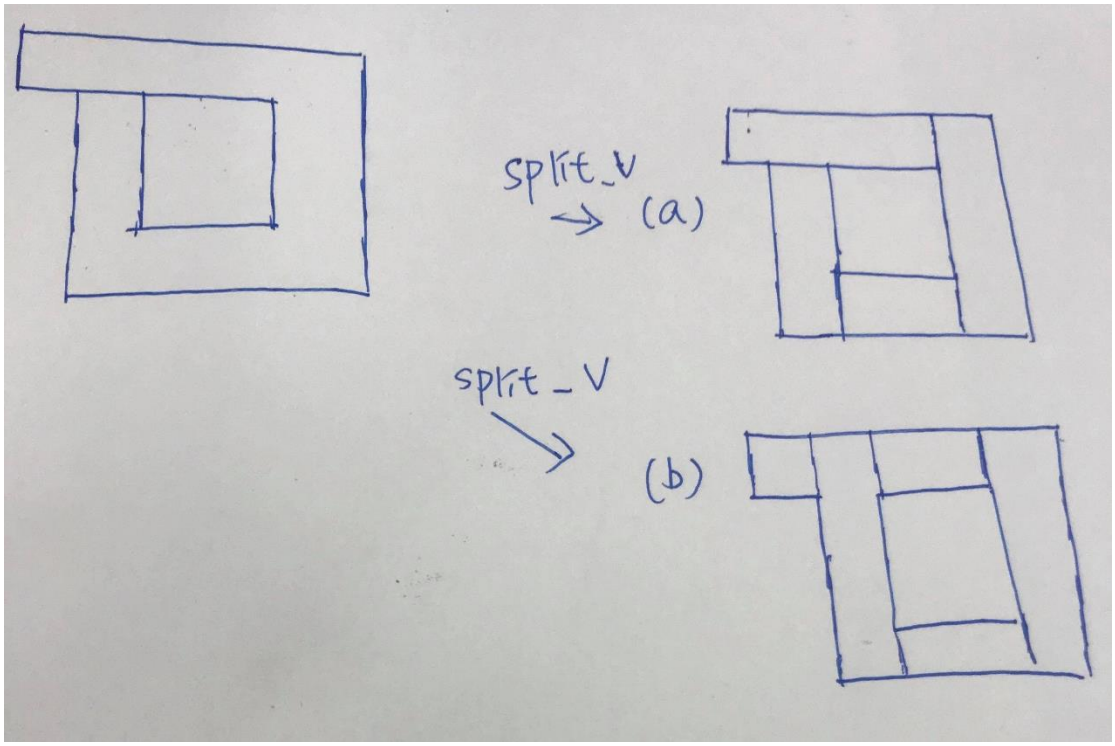
Q37. 你好，剛剛我們測試 problem E 的 checker 時，發現有 permission denied 的警告，導致無法正常執行 checker，我們都有按照步驟執行，可是還是無法正常使用。
A.37 已請主辦單位更新文件。若依然有問題, 請附上顯示的 Messages。

Q38. 在 Q32 的回答中
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。
以 Q32 所提到的 opencase1 的圖為例，要如何判斷中間的 polygon 是一個 hole，還是兩個 polygon 的 overlap？
A38. "要如何判斷中間的 polygon 是一個 hole，還是兩個 polygon 的 overlap", 以及在 Merge operation 過程中所產生之 Hole 的判斷與處理, 皆為該題目的一部分。

Q39. 狀況一、在同個 input file 內，Polygon 方向是否同時存在順時鐘與逆時鐘排序？
狀況二、在同個 operation 內，Polygon 方向是否同時存在順時鐘與逆時鐘排序？

請問：
若同個 input file 內，Polygon 會有不一樣的方向；那同個 operation 中，Polygon 方向是否相同？
A39.
1. 在同個 input file 內，Polygon 方向有可能同時存在順時鐘與逆時鐘排序。
2. 在同個 operation 內，Polygon 方向有可能同時存在順時鐘與逆時鐘排序。
3. 在同個 input file 內，Polygon 會有不一樣的方向，在同個 operation 中，方向也有可能不同。

Q40. 請問若 input file 中出現圖中左上多邊形自邊相交的情況，若要對他做 split，請問要移除相交的邊又或者是考慮相交的邊去做 split 呢?



A40. 若出現附圖左邊該圖形時，結果應該要為右下角 (b) 之結果。

Q41. 想請問題目中所述五個 public case 之中的剩下三個還沒公佈的 cases 會在什麼時候公佈，謝謝。

A41.

Open case 3：預計 7/21 前公布；

Open case 4：預計 7/28 前公布；

Open case 5：預計 7/31 前公布。

另外，各個 open case 的 checker 皆會在 7 月底前公布。

Q42. 請問 open case 3 , 4 ,5 會於何時公布？

A42. 同 Q41。

Q43. 請問執行 ./myPolygon input_file output_file 時
input_file 和 output_file 是否要加上附檔名嗎
例如: ./myPolygon input_file.txt output_file.txt

A43. 不需要附上附檔名。

Q44. 不好意思請問 A43. 不需要附上附檔名。
那我們可以要求附上嗎
例如: ./myPolygon input_file.txt output_file.txt

A44. 為了之後的 Evaluation，我們必須統一格式，且 "input_file" & "output_file" 是指任意檔案名稱，非固定名稱。

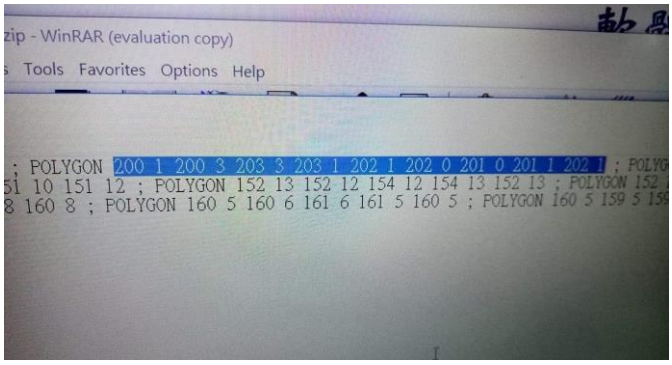
Q45. 想請問在 beta test 中會去測試 3 個不公開的隱藏測資嗎?能否得知隱藏測資跑出來的結果是 pass 或是 fail，謝謝。
A45. 我們已經在 Alpha test 當中，測試 2 個不公開的隱藏測資。在 Beta test 中，我們將會對所有的測資進行 Evaluation。

Q46. Based on the alpha test results, no team can successfully solve case 4 of problem E. I was wondering if organizer of problem E can release case4. (It would be good if it can be released before beta test). Without the case, it is quite difficult for contestants to debug.
Thank you very much for your kind help.

	Rank 1	Rank 2	Rank 3	Rank 4	Rank 5
Case 1 - Result	Pass	Pass	Pass	Pass	Pass
Case 1 - Time	0.05 sec.	0.16 sec.	0.11 sec.	0.10 sec.	2.31 sec.
Case 2 - Result	Pass	Pass	Pass	Pass	Pass
Case 2 - Time	17.83 sec.	42.51 sec.	111.93 sec.	20.76 sec.	1014.18 sec.
Case 3 - Result	Pass	Pass	Pass	Pass	Pass
Case 3 - Time	272.38 sec.	961.08 sec.	1437.43 sec.	12,826.96 sec.	25,307.17 sec.
Case 4 - Result	Failed - No Output	Failed - Error	Failed - Error	Failed - No Output	Failed - No Output
Case 4 - Time	x	x	x	x	x
Note					

A46. We will release similar case which like case 4. And the estimated releasing date will be the end of this week.

Q47. 不好意思
在 opencase4 裡面我們有發現非封閉的 polygon
還有一些線有交叉的問題



A47.
關於非封閉的 Polygon，已經請主辦單位做更新，感謝指出問題。
關於線交叉的問題, 若是線跟線產生交叉(Cross Edges)，且沒有在交叉點打上座標, 煩請指出有問題的地方，感謝！

若是頂點 (Vertex) 有在欲穿越的線 (Edge) 上標上座標 (touching) 後，Edge 再繼續往前走，則沒有違反題目的規則。

Q48. 我發現日前公布的 Opencase_4 在第 63 行的地方把"END DATA"打成"END"了，如下圖：

```
57 POLYGON 162 4 162 6 160 6 160 4 162 4 ;
58 POLYGON 162 1 162 2 160 2 160 1 162 1 ;
59 POLYGON 162 4 163 4 163 2 162 2 162 4 ;
60 POLYGON 162 8 163 8 163 6 162 6 162 8 ;
61 POLYGON 162 12 163 12 163 10 162 10 162 12 ;
62 POLYGON 205 5 206 5 206 4 205 4 205 5 ;
63 END
64
65 DATA CLIPPER C2 ;
66 POLYGON 160 8 160 9 159 9 159 8 160 8 ;
67 POLYGON 160 8 160 7 161 7 161 8 160 8 ;
68 POLYGON 160 5 160 6 161 6 161 5 160 5 ;
69 POLYGON 160 5 159 5 159 4 160 4 160 5 ;
70 END DATA
```

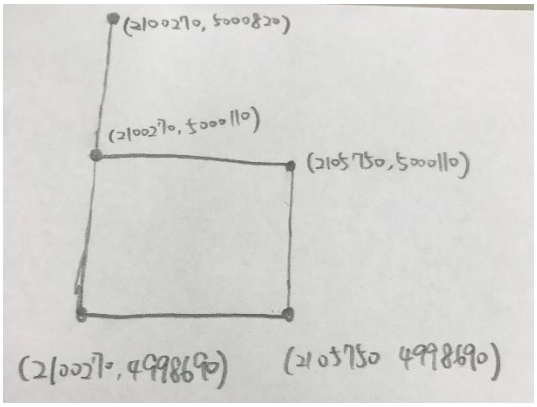
這樣會造成我們讀檔失敗，麻煩修正了～～

A48. 已更新，感謝指正。

Q49. case4 中的第 7、8、14 行的多邊形皆為非封閉且不完整的多邊形，麻煩再做修正。

A49. 我們檢查了 Case 4 的 7、8、14 行的多邊形，皆為封閉且完整的多邊形，並無問題中所描述之問題。

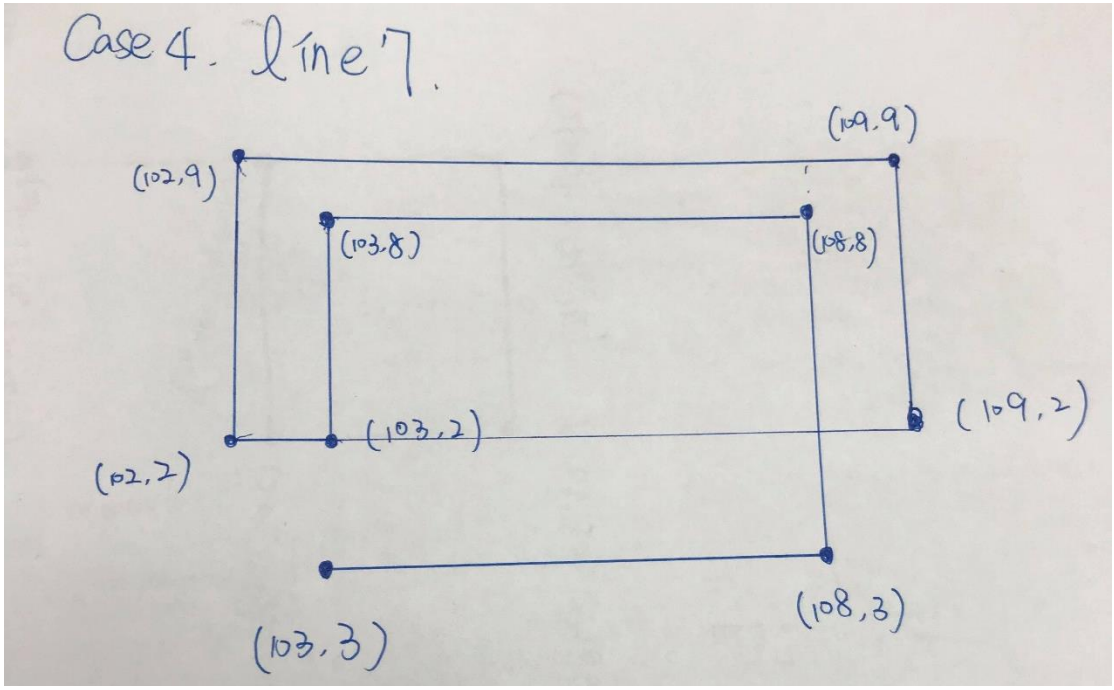
Q50. 在 CASE5 中出現以下多邊形，請問是合法的嗎?若是合法的請問若是要做 merge 或是 clipper 操作，是要自行判斷並且移除(2100270,5000820)這個點嗎?



A50. Case 5 已修正, 已請主辦單位更新。

Q51. 承 QA49，在 QA49 中回答道 CASE4 中的多邊形沒有問題，下圖為 CASE4 中第 7 行多邊形的樣子，所以 CASE 中有可能所給的第一點與最後一點為不同的狀況?
另一個問題是，此多邊形有很多無面積線段請問是要自行移除嗎例如點(103,3)及點(108,3)

```
7  POLYGON 103 3 108 3 108 8 103 8 103 2 102 2 102 9 109 9 109 2 103 2 ;
```

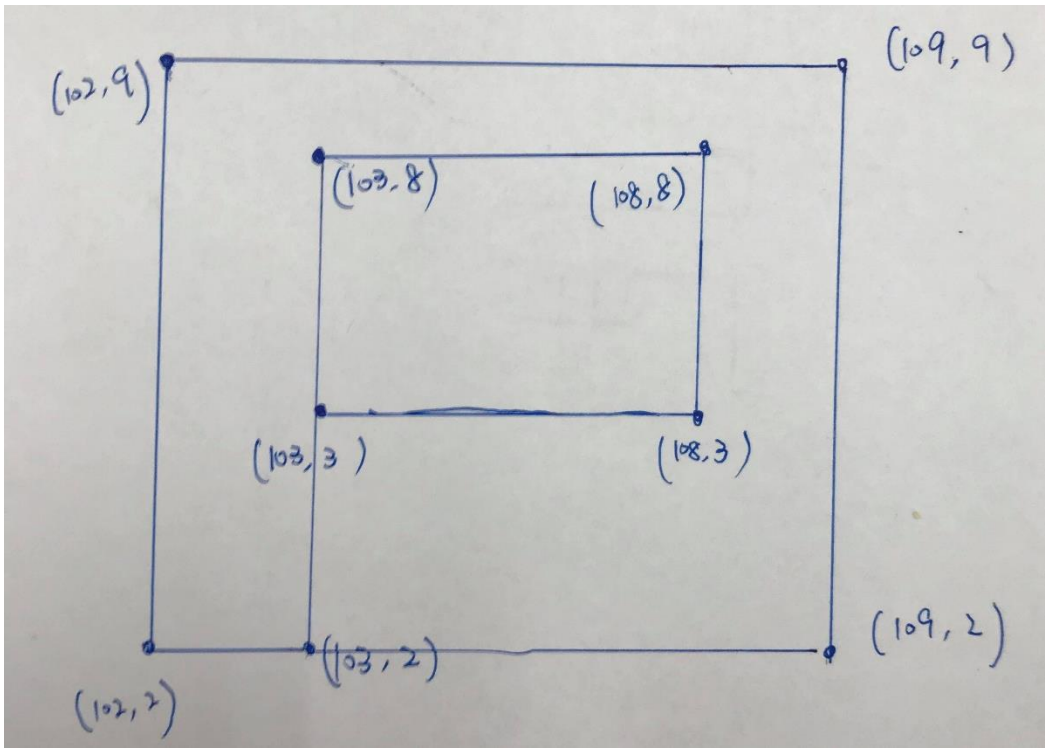


A51. 在 Document 中已有描述 "The description of polygon may be clockwise or counterclockwise, and the last set of coordinates may not be the same as the first set of coordinates"
另一個問題, Q51. 所附的圖本身即有誤, (103, 3) 與 (103, 2) 的位置顛倒了, 請再查看之。

Q52. 在 case4 出現以下多邊形

```
POLYGON 103 3 108 3 108 8 103 8 103 2 102 2 102 9 109 9 109 2 103 2 ;
```

圖形如下



在 FAQ47 中回答道，線會在欲穿越的線上再次打上座標，以此端邊形來看，是否應該在點 103 8 與 103 2 之間再次插入點 103 3 才合理?否則無法判斷此線穿越了線 103 3 108 3

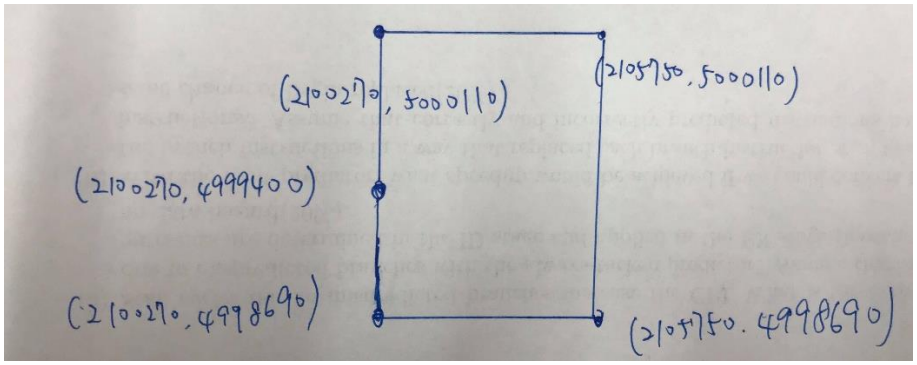
A52. 關於 Edge Cross 與 Vertex Touching 的問題，若是線跟線產生交叉 (Cross Edges) 時，應會在穿越的線上打上座標。

舉例來說，將該題的第一組 Vertex 修改成 (102, 3) 時，那在第四組 Vertex (103, 8) 的下一個座標點，就應該為 (103, 3)，而非 (103, 2)

然而，在 Case 4 中，在第四組 Vertex (103, 8) 與第五組 Vertex (103, 2) 之間，已存在一組之前有描述過的 Vertex 座標 (103, 3)，且該圖形描述為 Edge 與 Vertex 交疊 (Touching)，並非線與線交叉，在 Q32. 已回覆過，因此，該 Polygon 描述則已能判斷中間隱含著 Hole 之資訊。

倘若存在一 Polygon 之描述，為該 Polygon vertex 之倒述 (Reverse order)，即能解釋上述之說明。

Q53. 在 CASE5 第 10 行中出現以下 polygon



請問所提供的點不一定會剛好在多邊行的 corner 上，需要自行判斷刪除(2100270,4999400)這個點嗎？
A53. 若在 Polygon 中有出現 Redundant 的點，是需要自行判斷刪除的。

Q54. 想請問在測試結果中出現 NO output 的意思是包含 "程式執行結束後沒有任何 output(執行時間少於八小時)" 跟 "程式執行超過八小時" 兩種可能嗎?還是若程式執行超過八小時會以 time exceed 之類的詞彙來表達呢？
A54. No output 是指程式以正常或不正常(Segmentation fault) 的方式結束後，表示沒有任何 output 的描述。若程式執行超過八小時，會以 "Time out" 作為表示。

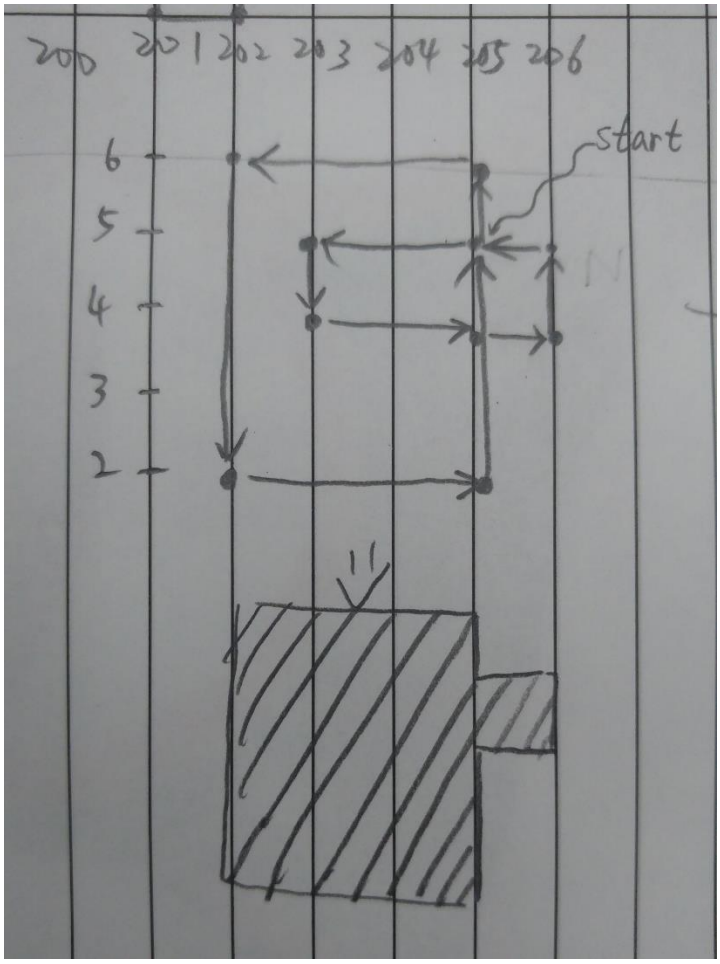
Q55. 在 problem E 中，若 CASE 的 split type 為 split_o，則最後的評分標準會有 70%是取決於最後的矩形數量，想請問可否公布 Beta test 中 top 5 最後 split type 為 split_o 的 case 各自所切出來的矩型數量方便參考?
A55. 已請主辦單位更新。

Q56. 由於 5 個 OPEN CASE 都沒有 split_o,想請問可否另外提供五個 opencase 最後 split type 轉換為 so 的 checker 又或者是另外提供一個 split type 為 split_o 的 case 並提供 checker 以供確認參賽者確認?
A56. 關於 Split_O 的測試方式，參賽者可以透過修改任一 open case 檔頭的最後一個 Operation，改成 'SO' 做測試。
檢驗結果的方式，則可以將 'SO' 的結果，另開一新檔，當作 input polygons (M1)，並將 Operation 定義為 'M1 SV' 或 'M1 SH' (視該題原本最後的 operation)。
再透過線上的 Checker 對該題檢驗之，則達到對 Split_O 的測試。

Q57. 關於三個 non-disclosure testcases，如果存在先前 open testcases 中皆未曾出現過的特殊形式 polygon，是否能分別舉出幾個相似的例子以供修正？
A57. 在五個 Open case 當中，已展現題目中所有類型的 Polygon。
在三個 Non-disclosure test cases 中的 Polygon 類型，皆與 Open case 雷同。

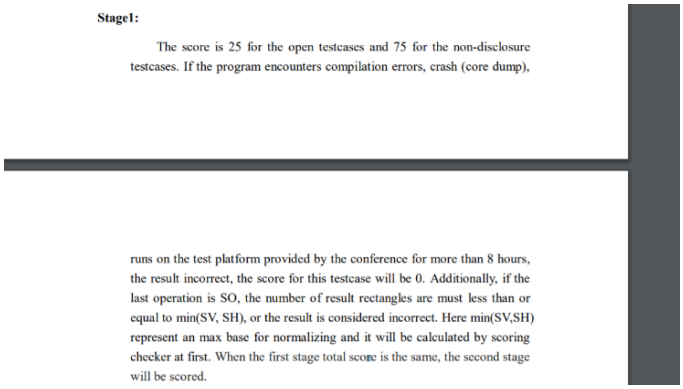
Q58. 想確認一下，在 case 4 的第 15 行，也就是
POLYGON 205 5 205 6 202 6 202 2 205 2 205 5 203 5 203 4 205 4 206 4 206 5 205 5 ;
畫出來會如下圖所示，想問這樣的形狀我們應該視作一個完整的側躺的"凸"字，中間交叉的部分並不代表任何洞的存在對嗎（因為都是同方向去

圍）？也就是我們處理完後，只會得到下圖下半斜線區域的那個多邊型，不知這樣理解是否有誤？



A58. 該爭議 Polygon pattern 將請主辦單位幫忙移除之。

Q59. 在 beta test 中有公布各組 stage 1 的成績，但從 stage 1 計算成績的描述看不出來實際的計算方式，如圖中所示，敘述中只提到在 compile 或執行有問題的情況下得 0 分，但沒有提到實際上得分的計算方式；由 beta test 前五名 stage 1 成績來看，在通過的 case 數量相同的狀況之下得分也不一定相同，想請問 stage 1 成績詳細的計算方式(例如是依據 runtime 或是只要通過就能取得分數)。



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.

A59. Stage 1 的成績計算上，並無參考 run time 來做計分評估，Stage 1 在 Timing 部分，只有在超過 8 小時後，該 Case 才會以零分計算。
若出現兩組參賽者在總分相同時, 才會進入 Stage 2 的評分 (即參考 run time 以及 Result)。
在 5 個 Open cases 中，總分占 25 分。在 3 個 Non-disclosure cases 中，總分占 75 分。
每一個 Case 的得分不盡然相同，實際每個 Case 的得分數如下

Open Case 1: 1 分
Open Case 2: 4 分
Open Case 3: 5 分
Open Case 4: 8 分
Open Case 5: 7 分
Non-disclosure Case 1: 20 分
Non-disclosure Case 2: 25 分
Non-disclosure Case 3: 30 分

該資訊會再請主辦單位補充至 document 中。

Q60. 由 A52.所述：
"關於 Edge Cross 與 Vertex Touching 的問題，若是線跟線產生交叉 (Cross Edges) 時，應會在穿越的線上打上座標。"
也就是說在此情況下會出現連續三個點在同一條直線上，例如 POLYGON 57 10 53 10 53 3 50 3 50 7 53 7 60 7 60 3 53 3 53 0 57 0 57 3 57 7 57 10 ;其中 53 7 這點代表穿越線的點，請問這種連續三個點在同一條直線上的情形是只會出現在這種狀況嗎？
A60. 根據該題提問的描述，若連續三個 同樣 的點出現在同一條直線上，以該題所有的 Cases 中，是只會出現在這種狀況的。
且 同樣 的點不會以相鄰的方式出現。例如. (0, 0) (1, 0) (1, 1) (1, 1) (0, 1) (0, 0), 所有的 Cases 中，不會出現上例這種 Polygon。
（針對該問題，在 A64.有詳細說明）

Q61. 剛剛看了最新 Q.58 的回應，說道"若 Polygon 之間有自旋出現 sub-region 時，會採用 Even-odd rule 去描述該 sub-region"。
然而 Even-odd rule 這個名詞似乎是過去說明文件中從未提及的，以下是我在 Alpha Test 之前就已經下載的文件截圖：

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:

當中並未提及 even-odd rule，用當時的文件按 ctrl-f 搜尋"odd"字眼，也沒有找到任何結果。

而在今日最新版的文件中，才在上圖第一段末多加了一句話，"Besides, if the polygon description has sub-regions, it should follow even-odd rule."。

首先，因為文件一開始有說"Hole is not described in the test file, so there is no format for the description of the hole."，而在 Q.32 中也已經說明：
「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。」
意指只能用合法的 polygon 用隱含的方式（如 Q.52 中用相依偎的 edge 去圍出 hole）去造出隱含的 hole。
然而 even-odd rule 自己就是一個 hole 的表示方式，只是比 Contour 麻煩一點，一樣可以造出各式各樣的 hole，這樣是否違反先前的說明呢？

其次，我想都已經過了 Alpha & Beta Test 了，才跟參賽者告知，其實必須使用 even-odd rule 才能正確通過我們的測資，這樣其實已經算是臨時的規則改變了，是否有點不妥？

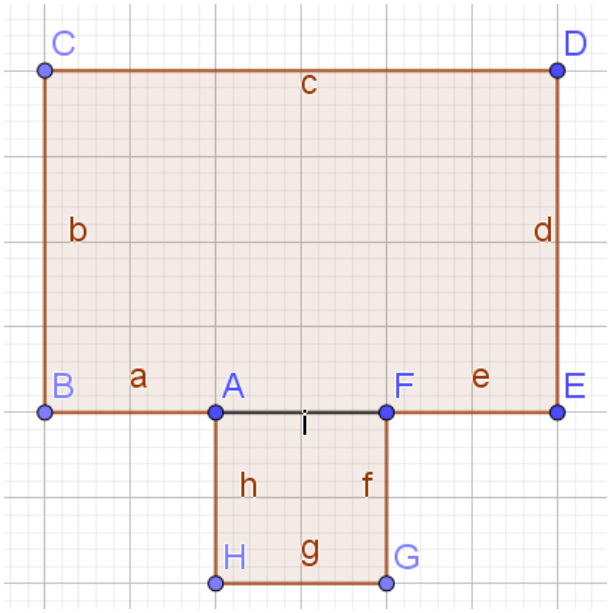
不知道有沒有可能把需要 even-odd rule 才能處理的 polygon 做移除呢？謝謝您。

A61. 我方描述該題題目時，原本已是基於此規則之下做描述，但在初始文件中確實缺乏對該規則的說明。
因此，經過我方討論之後，決議將有使用到 Even-odd rule 的爭議 polygon 移除之 (包含 Open case 以及 Non-disclosure case)。
該更動會影響到的 Open Case 為 Open Case 4，我們會再請主辦單位更新 Open Case 以及 Checker。

Q62. Opencase5 的 output 數量是否有問題？目前我們這組做出來的答案是 36,600 個。
A62. Open case checker 提供的 result 應該是正確的，若仍對 check result 有疑問，請提供 checker 的 error message，以便查看之，謝謝。

Q63. 看到最新的 FAQ，表示會移除使用 even-odd rule 的 polygon，但理論上所有有 hole 的 polygon 都算是用 even-odd rule 判斷的，請問到底怎麼樣的 polygon 是會出現在競賽中？Non-simple polygon 到底要怎麼處理？距離 Final subission 已經沒剩幾天了，**希望官方盡快給一個明確、完整的規則敘述。**(過去的敘述都明顯不足，例如之前表示會以 even-odd rule 判斷 polygon 的區域，題目也應該也要附上 even-odd rule 的說明與舉例)。

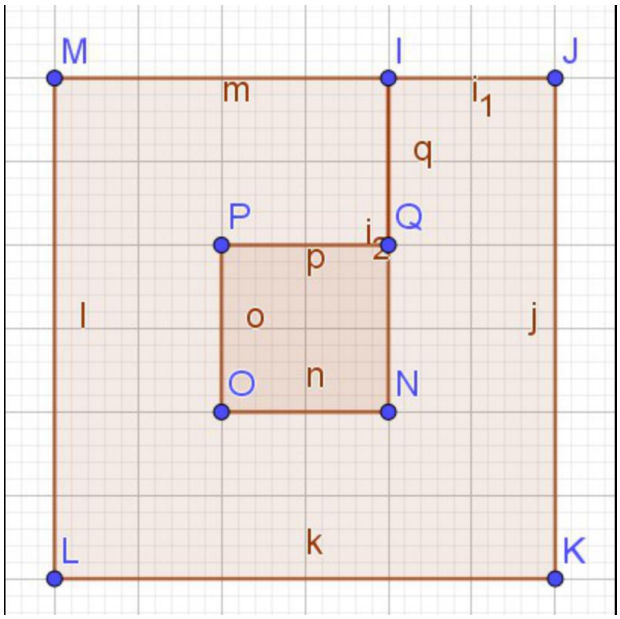
我在新的 opencase4 中仍有許多定義不明確的 polygon，以 line18 為例：
POLYGON 201 1 200 1 200 3 203 3 203 1 202 1 202 0 201 0 201 1 202 1 ;



請問如果不使用 even-odd rule 的話，這就是一個 T 字型的 polygon 嗎？所以圖中 AF 點間的線 i 就是直接忽略嗎？

又如 line8 中：

POLYGON 106 7 107 7 107 4 104 4 104 7 106 7 106 5 105 5 105 6 106 6 ;



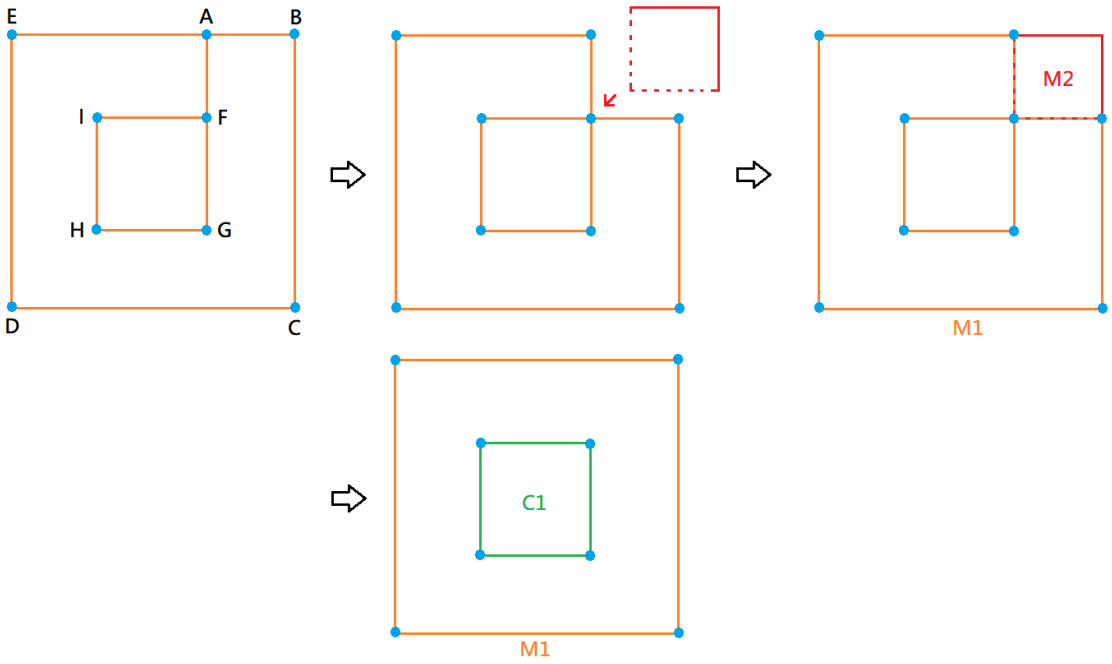
不使用 even-odd rule 的話，是否代表中間的 square 不用消去？這樣又是怎麼樣的一個圖形？

另外，我覺得 QA 中相關敘述都很不明確、有 ambiguous、不易理解，可否在未來改善？

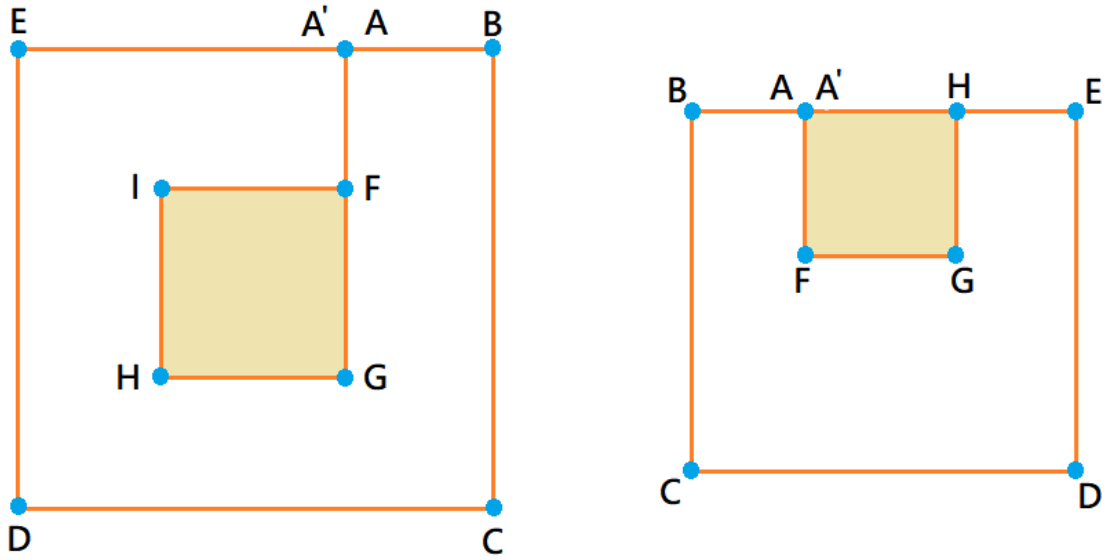
主辦單位到截止日期將至了還一直補充、更改題目，真的讓人覺得很辜負我們對這個競賽投入的心力，希望未來不會再這樣了。

A63. 致 Q63. 參賽者，我們一開始出題時，確實是以 even-odd rule 在思考該題題目，因此沒有顧及到其他可能延伸的問題，關於這部分我們深感抱歉！
競賽進入尾聲有參賽者反應我們才發現，確實先前沒有明確定義該題的 polygon 遵循規則，若單一 polygon 描述出現自我交疊等問題時，應該遵循 even-odd rule 或是 winding rule；因此，才將部分有疑慮的單一 polygon 描述改寫，目的是避免存在這樣的單一 polygon 描述，而導致有不同 rule 需要考慮的情形，也就是說我們簡化了 testcase，而非新增不同的規則，並將問題導正回 polygon operation，以彌補當初 polygon 遵循 rule 描述不仔細所造成的後續問題，這部分是我們的疏忽，再次抱歉！
我們會盡速將所有 case 中，有疑慮的 polygon 做修正，在 8/27 以前 release，以供參賽者做驗證。
以 Q63. 附圖一為例(Q63. 上圖)，在 Line 18 當中，AF 點間的線 i 可忽略。
以 Q63. 附圖二為例(Q63. 下圖)，我們將對該 polygon 做修正，在單一 polygon 描述當中，不再出現有需要遵循 even-odd rule 或是 winding rule 的爭議問題，而會以 operation 後的結果，去表現出該 pattern 想要測試的目的。

如下圖所示，原本出現的單一 polygon 描述(左上)，將會以右上或是右下的方式，透過 operation 後來呈現。



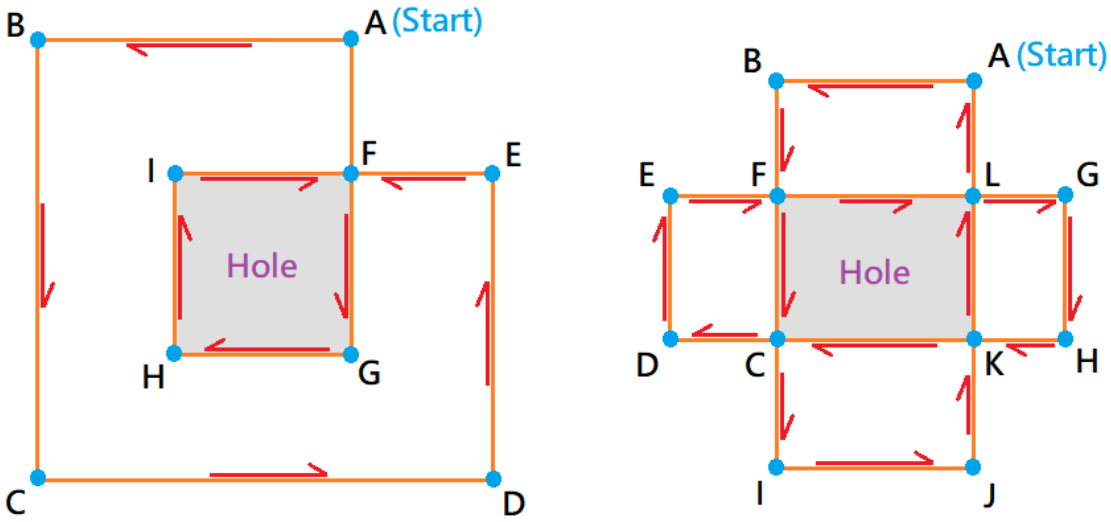
在定義上，該題 open 以及 non-disclosure 的 case 當中，non-simple polygon 出現的形式，*不會*出現下列類型的圖形



上圖(左)，由於 polygon 描述順序自 F 點開始，已落入了封閉的 polygon 中，會涉及到 even-odd rule 或是 winding rule 的爭議性，因此，不會出現在修正後所有 case 當中

上圖(右)，由於 polygon 描述的點，最終導致有 area 出現重疊的部分，會涉及到 even-odd rule 或是 winding rule 的爭議性，因此，不會出現在修正後所有 case 當中

而下圖的類型，由於給定的順序描述點，沒有落入已封閉的圖形中，因此 會出現 在 case 當中



因此，對於本題的 case 修正後，不會再出現有該爭議性之 polygon 描述，並將問題核心導正在 polygon operation；此外，修正後的 case 經測試後，不影響目前已發布之 beta test 的分數以及排名結果。

Q64. 請問 A60 的回答，根據該題提問的描述，若連續三個 同樣 的點出現在同一條直線上， 以該題所有的 Cases 中，是只會出現在這種狀況的。 且 同樣 的點不會以相鄰的方式出現。例如. (0, 0) (1, 0) (1, 1) (1, 1) (0, 1) (0, 0), 所有的 Cases 中，不會出現上例這種 Polygon。

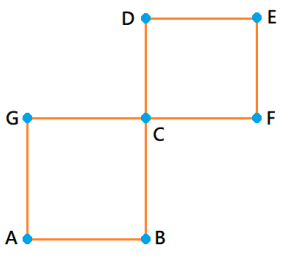
請問是否應該把紅字標明的" 同樣的"三個字去掉，如此才有回答到 Q60 所提問之" 連續三個點在同一條直線上的情形是否只在穿越線的情況下發生"之問題。

A64. 針對 A60. 回答中，補充詳細的說明如下。（會再請主辦單位補充更新 A60. 的回答）

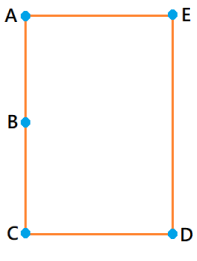
情況一. 若 同樣 的座標點出現在同一條直線上，在所有的 Cases 中，只會出現在 Vertex touching、edge cross 的情況，或是封閉 Polygon 的節點，如下左圖。

C 同樣的座標點出現了兩次，該座標點為 Vertex touching、edge cross；A 同樣的座標點出現了兩次，該座標點為封閉 Polygon 的節點。

情況二. 若 連續 的點（非 同樣 的座標點）出現在同一條直線上，如下面右圖，A-B-C-...，同 A53. 解答，這在 Case 當中是會出現的，參賽者若有需要，需自行 Simplified 該多餘的點。

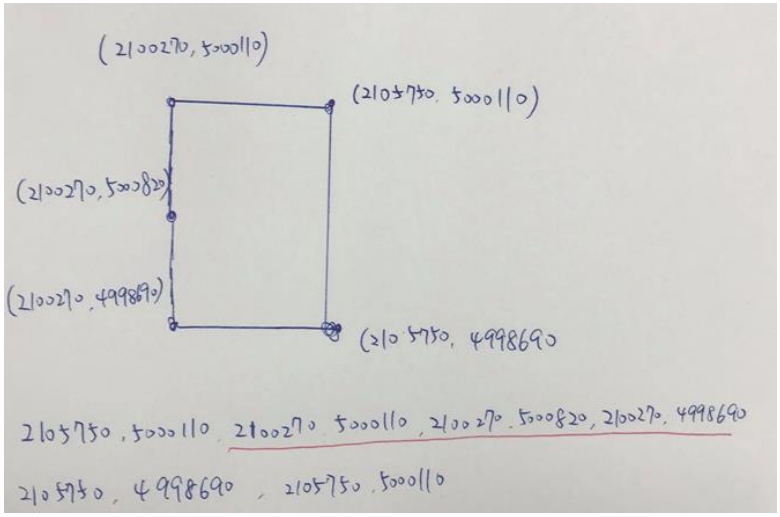


A, B, C, D, E, F, C, G, A



A, B, C, D, E, A

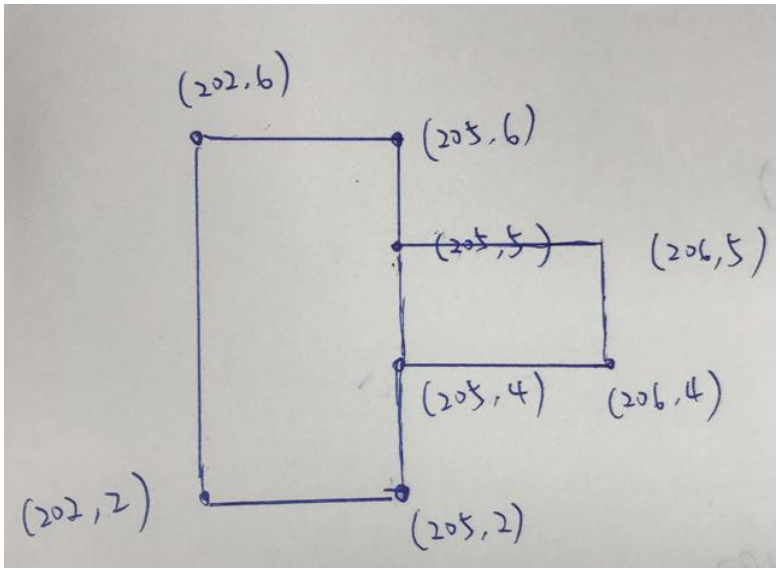
Q65. 而若 A60 有修改，則意思應該為，若出現連續三個點在同一直線上，則只會出現在線穿越的情況下，但在 Open_case5 第 10 行中出現以下多邊形。



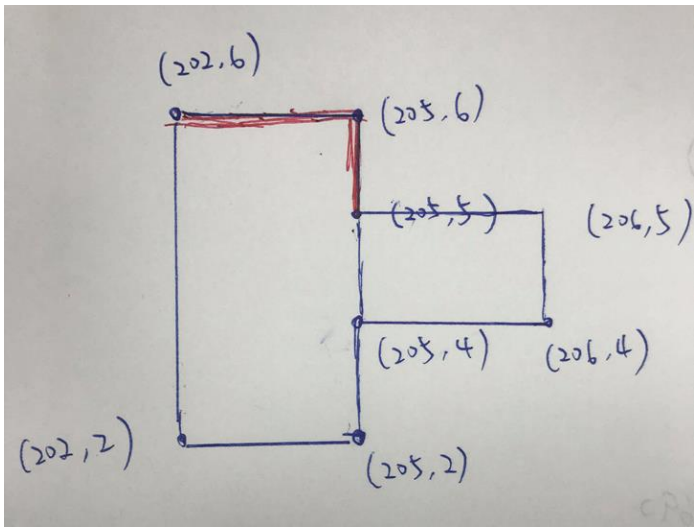
由 A53 我們得知此種描述是合法的，需要自行刪除 Redundant 的點，此種描述一樣有三個連續的點出現在同一線上的情況，但是並沒有任何線的穿越，請問是否應理解為 ” 若連續出現三個點在同一線上，在代表線的穿越或者是有 Redundant 的點 ” 這樣呢？

A65. 該題合併在 A64. 回答中。

Q66. 可否保證所有的 input polygon 的描述順序，不會重複描述 edge？
以 Open_Case4 的第 15 行為例: 原本的描述為 POLYGON 205 5 205 6 202 6 202 2 205 2 205 4 206 4 206 5 205 5 205 4 ;如下圖所示。



另一種會導致 edge 重複描述的順序例如 POLYGON 202 6 205 6 205 2 202 2 202 6 205 6 205 5 206 5 206 4 205 4; 如下圖所示



可以看到若改為此種順序描述，則在圖中多邊形畫紅色的線段被描述了兩次，請問會有這種強況嗎?或是能確保每條線段最多只會被描述一次？

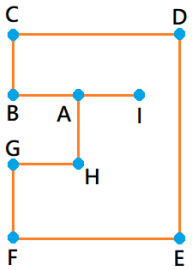
A66. 在所有的 Case 中，皆不會出現同樣的 Edge 重複描述之問題，意即不會出現問題中重複描述的例子。

Q67. 在 QA35 中回答到不會出現附圖的情況，我們的理解是，已封閉的 POLYGON 區域內不會再描述多餘的 line，但在 QA63 的上圖中卻出現了 POLYGON 區域內描述多餘的 line 的情況（edge I），而依據 A63 的回答，這似乎又要參賽者自行判斷刪除，這似乎又與 A35 所給的回答衝突？距離繳交期限只剩幾天了，如今才突然出現這種 case 是否不合理？可否定義清楚什麼狀況的多邊形會出現什麼不會出現或者趕緊修改 CASE。

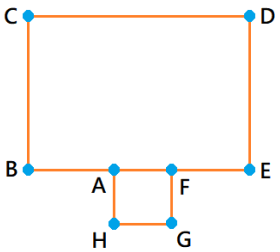
A67. 詳細的說明如下。

情況一. 在已封閉的 Polygon 中，出現了一 Edge，且該 Edge 的終點沒有與 Vertex 相連，在此情況下，如同 A35. 回答，該題所有的 Case 中，皆不會出現，如下左圖。（Edge A - I）

情況二. 在已封閉的 Polygon 中，出現了一 Edge，且該 Edge 的兩端 Vertex 皆有和其他 Vertex 相連，在該題原先的 Case 中，即有同樣類型的 Case（Open case 4），如下右圖（Edge A - F），但由於 even-odd rule 的疑慮，目前經 A63. 之問題修正後，已將涉及需要做 even-odd fill 的 Polygon 改寫。



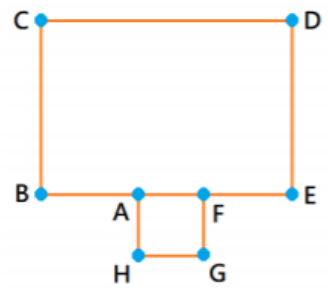
A, B, C, D, E, F, G, H, A, I



A, B, C, D, E, F, G, H, A, F

因此，無論是上例左圖或是右圖，皆不會出現在所有 Case 當中。

Q68. 根據最新的 FAQ67 所述，如下圖類型的 polygon 已被刪除，那請問我是否可以理解為「所有的 case 中皆不包含 edge touching 且 input 的起始點與終點必為同一點」？

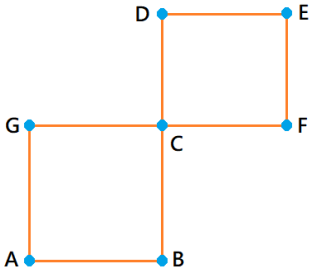


A. B. C. D. E. F. G. H. A. F

A68. Case 中依然會包含 Vertex touching 以及 Edge cross（如下圖圖例）。

由於 Polygon 不同的描述方式，而會有 Vertex touching 以及 Edge cross 不同的展現效果，然而，在圖例中的 Polygon 本質上，依然為同一個 Polygon 的描述。

在下例 Polygon 描述 1 當中，C 點為 Edge cross；在下例 Polygon 描述 2 當中，C 點則為 Vertex touching；且該類型的 Polygon 已出現在 7 月釋出的 Open case 4（Line 5）中。



1: A, B, D, E, F, C, G, A
2: A, B, C, D, E, F, C, G, A

在經過 A63. 修正後的所有 Case 中（Open case 以及 Non-disclosure case），**Polygon 起始點與終點必為同一點**。