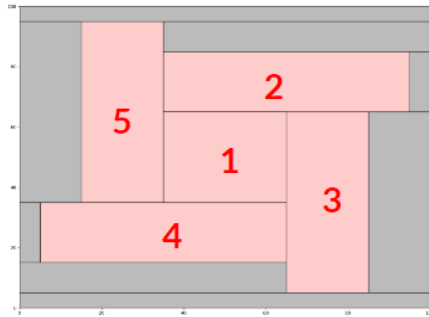# Lab1 – Corner Stitching

Lab1 Introduction

This is an exercise lab to review corner stitching data structure. You have to organize the space tiles as maximal horizontal stripes. This means that no space tile has other space tiles immediately to its right or left and every space tile must be as wide as possible.

Input


Example (case0.txt)

```
100 100           // outline width, outline height
P 35 35              // P, X coordinate of the point, Y coordinate of the point
1 35 35 30 30   // block index, coordinate of lower-left corner, block width, block height
2 35 65 60 20
3 65 5 20 60
4 5 15 60 20
5 15 35 20 60
P 35 35
```

Explanation of the Input

1. The first line gives the outline size and the coordinate of lower-left corner is (0,0).
2. The next line to the last line:
   - There are 2 kinds of Command: **Point_Finding** & **Block_Creating**.
   - **Point_Finding**: P, X coordinate of the point, Y coordinate of the point
   - **Block_Creating**: block index, coordinate of lower-left corner, block width, block height.

   Figure of the second line:

   (35,65) ┌──────┐ (65,65)
           │   1   │
   (35,35) └──────┘ (65,35)
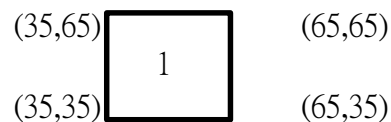
Note: * All the values in the input file are integers.
   * All the given blocks are rectangles.
   * All the given blocks will not overlap or beyond the outline.
   * Block indexes must be positive, but may not be consecutive.
   * Point_Finding's X coordinate range: [0 ~ outline width-1]
   * Point_Finding's Y coordinate range: [0 ~ outline height-1]
   * Block number: 1~25000
   * 1 <= Block Index <= INT_MAX-1
   * Outline_Width <= 200000, Outline_Height <= 200000

<u>Output</u>

<u>Output Format (output.txt)</u>

```
13     // number of tiles

1 4 0 // block index, the number of adjacent block tiles, the number of adjacent space tiles
2 3 3
3 3 3
4 3 3
5 3 3
0 0    // the lower-left corner's coordinate of tile where the point is located
35 35
```

<u>Explanation of the Output</u>

1. The first line outputs the number of tiles within the outline.
2. The next line to the fifth line outputs each block number, the number of block tiles adjacent to the block and the number of space tiles adjacent to the block in block index ascending order.
3. The seventh line to the last line outputs each lower-left corner coordinate of the tile where the point is located in input order. In addition, you need to output the block's lower-left coordinate at the time of point input, not the final situation.
4. Each point belongs to only one block. If the point appears on the edge or corner, you should output according to the following below rule:

<center>*"Right edge and top edge not belong to block"*</center>

<u>Example:</u>

- Referring to right picture, the red point is the corner of 4 blocks, but according to the above rules, the red dot only belongs to block3

<u>Note:</u>

- Lab1_Supplementary.pdf would help you better understand this Lab.

<u>Environment</u>

1. Linux (Please make sure your code is available on our linux server(compile & execute). If it does not work, you will get zero points!!)
2. Makefile should be provided

<u>Evaluation</u>

1. You **MUST WRITE YOUR OWN CODE**. Copying codes may result you to **FAIL** this course.
2. A script on checking similarity for codes will be performed. If two codes have high similarity. Both codes receive zero marks after confirmation.
3. For each case, the run time limit is up to 1 minute. It will be regarded as "failed" if you use more than 1 minute.
4. Naming rule.

   A.   Name of the binary after "make" – Lab1
   B.   Execution procedure: ./Lab1 [input] [output]
   C.   Not following specified naming rule will receive zero mark

3. Late submission will **NOT** be accepted.

4. Hidden cases will be evaluated.

<u>Submission</u>

<1>

Please put all required files in a folder, the name of the folder is your Student_ID.

    Student_ID
       |__Source code (.cpp, .h)
       |__Makefile

<2>

Use the below command to compress the folder in the linux environment, and the compressed file name should be the same as your Student_ID.

    tar cvf Student_ID.tar Student_ID

<u>Example</u>

<1>

```
└── 310510161
    ├── Lab1.cpp
    └── Makefile
```

<2>

```
21:19 zu00895077@vda04 [~/PDA_TA/Lab1] >$ ls
310510161/
21:19 zu00895077@vda04 [~/PDA_TA/Lab1] >$ tar cvf 310510161.tar 310510161
310510161/
310510161/Lab1.cpp
310510161/Makefile
21:21 zu00895077@vda04 [~/PDA_TA/Lab1] >$ ls
310510161/  310510161.tar
```

Upload Student_ID.tar to e3
(If your submission file is not .tar, you will get zero point!!)