

# DOCUMENTATION OF CPAR

KRISHNENDRA SHEKHAWAT  
DEPARTMENT OF MATHEMATICS,  
BITS PILANI, PILANI CAMPUS, RAJASTHAN, INDIA, 333031  
EMAIL: KRISHNENDRA.IITD@GMAIL.COM

## 1. PROCESSING LANGUAGE AND INTRODUCTION

*Processing* is an open source programming language and environment for people who want to create images, animations, and interactions (see [1-2]).

It was developed by *Casey Reas* and *Benjamin Fry*, both formerly of the Aesthetics and Computation Group at the MIT Media Lab. Software written using Processing is in the form of so-called *sketches*. These sketches are written in a specific *text editor*, which can have lots of tabs to manage different files.

After trying various other systems, we have written our code for the prototype CPAR in Processing. This code is subdivided into several components to make it more comprehensive and each component is written in a separate file (tab). CPAR has two external files and 16 tabs in total; one external file for input and another one for output. The 16 tabs are illustrated in Figure 1. In upcoming sections, we explain the functioning of each tab.

When we run CPAR, it generates GUI (see Figure 3) and a .jpg file having plus-shape arrangement and its dual graph as shown in Figure 2 corresponding to length and height of sub-rectangles in Table 1 and weighted adjacency matrix shown in Table 2.

## 2. NOTATIONS

Notations frequently used in the text are given as follows:

$A_T$ : a weighted adjacency matrix

Given spaces: sub-rectangles

$R^A$ : rectangular arrangement

$R_S^A$ : spiral-based  $R^A$

$(L^1, H^1), (L^2, H^2), (L^3, H^3), (L^4, H^4), (L^5, H^5)$ : length and height of central, left, upper, right and lower  $R_S^A$  respectively

$R_i$ :  $i^{th}$  sub-rectangle

TABLE 1. Length and height of given sub-rectangles

	$R_1$	$R_2$	$R_3$	$R_4$	$R_5$	$R_6$	$R_7$	$R_8$	$R_9$	$R_{10}$	$R_{11}$	$R_{12}$	$R_{13}$	$R_{14}$	$R_{15}$	$R_{16}$
length	90	48	90	54	40	50	45	30	72	105	90	50	60	66	120	28
height	60	32	60	81	60	75	30	20	108	70	60	75	90	99	80	42

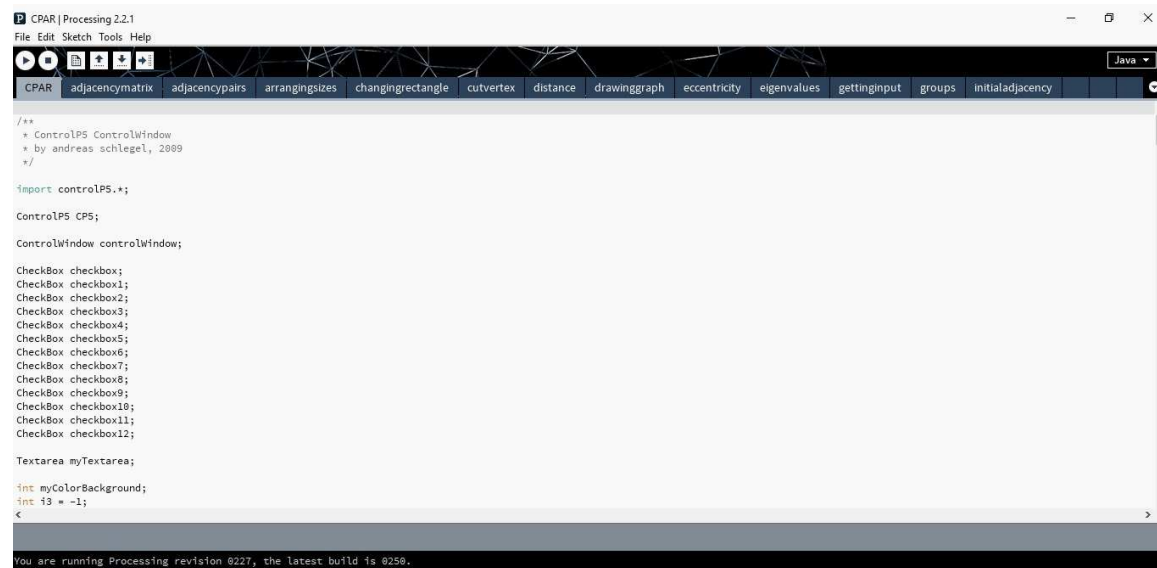


FIGURE 1. Screen of CPAR

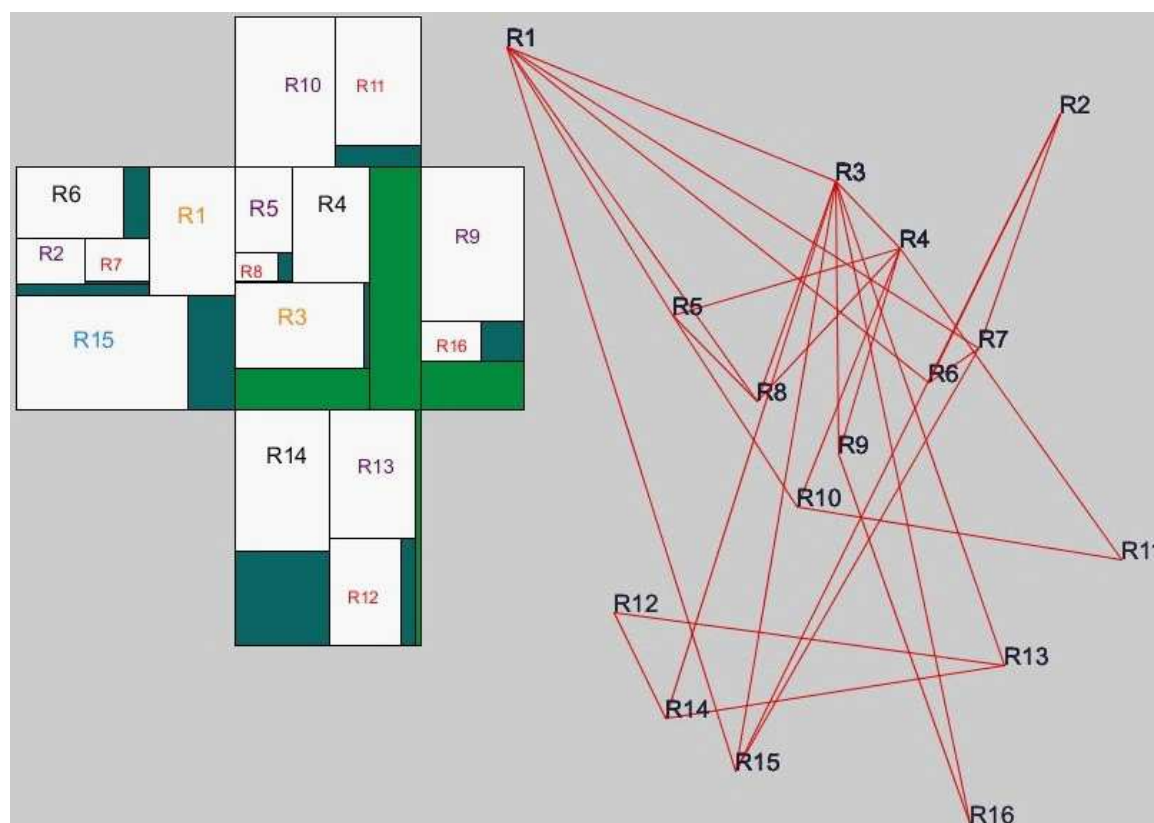
FIGURE 2. A  $P_S^A(16)$  and its dual graph generated using CPAR

TABLE 2. A weighted adjacency matrix of order 16

	$R_1$	$R_2$	$R_3$	$R_4$	$R_5$	$R_6$	$R_7$	$R_8$	$R_9$	$R_{10}$	$R_{11}$	$R_{12}$	$R_{13}$	$R_{14}$	$R_{15}$	$R_{16}$
$R_1$	0	8	6	6	8	6	9	6	4	5	3	2	2	2	8	6
$R_2$	8	0	6	6	8	6	9	6	4	5	3	2	2	2	8	6
$R_3$	6	6	0	8	6	8	6	9	4	4	3	6	6	4	4	6
$R_4$	6	6	8	0	6	8	6	9	4	4	3	4	4	4	4	6
$R_5$	8	8	6	6	0	6	9	10	2	2	2	2	2	2	4	2
$R_6$	6	6	8	8	6	0	10	9	6	2	2	2	2	2	4	2
$R_7$	9	9	6	6	9	10	0	6	2	2	2	2	2	2	4	2
$R_8$	6	6	9	9	10	9	6	0	6	6	4	4	4	4	4	6
$R_9$	4	4	4	4	2	6	2	6	0	8	6	6	6	6	6	9
$R_{10}$	5	5	4	4	2	2	2	6	8	0	10	4	4	4	6	4
$R_{11}$	3	3	3	3	2	2	2	4	6	10	0	2	2	2	4	4
$R_{12}$	2	2	6	4	2	2	2	4	6	4	2	0	8	10	2	9
$R_{13}$	2	2	6	4	2	2	2	4	6	4	2	8	0	10	2	9
$R_{14}$	2	2	4	4	2	2	2	4	6	4	2	10	10	0	2	6
$R_{15}$	8	8	4	4	4	4	4	4	6	6	4	2	2	2	0	2
$R_{16}$	6	6	6	6	2	2	2	6	9	4	4	9	9	6	2	0

$L_i$  and  $H_i$ : length and height of a  $R_S^A$  after drawing  $i^{th}$  sub-rectangle

$l_i$  and  $h_i$ : length and height of  $i^{th}$  sub-rectangle

$P_S^A$ : spiral-based plus-shape arrangement of order  $n$  i.e. having  $n$  sub-rectangles

MOI: moment of inertia

### 3. INPUT FOR CPAR

The input for the code is extracted from an external file *input.txt*. While writing the code, input.txt is kept external so that it can be more user-friendly. The file has the following 6 different inputs:

1. *A weighted adjacency matrix*

It gives the adjacency relation among all the sub-rectangles which need to be placed inside the plus-shape arrangement (refer to Table 2).

**Remark 1.** *The number of sub-rectangles( $n$ ) and a list of all the sub-rectangles are specified within the code in a tab input instead of Input.txt file.*

2. *The length and height of each sub-rectangle*

Refer to Table 1.

3. *Change of a sub-rectangle*

For obtaining  $P_S^A$ , we divide sub-rectangles into 5 groups. As an example, corresponding to Table 2, formed groups are as follows:

Group1:  $R_1, R_2, R_6, R_7, R_{15}$ ,

Group2:  $R_3, R_4, R_5, R_8$ ,

Group3:  $R_9, R_{16}$ ,

Group4:  $R_{10}, R_{11}$ ,

Group5:  $R_{12}, R_{13}, R_{14}$ .

Sub-rectangles  
R1(1) R2(2) R3(3) R4(4) R5(5) R6(6) R7(7) R8(8) R9(9)  
R10(10) R11(11) R12(12) R13(13) R14(14) R15(15) R16(16)

GROUPS  
Group 1: R1 R7 R15 R2 R6  
Group 2: R3 R8 R4 R5  
Group 3: R9 R16  
Group 4: R10 R11  
Group 5: R12 R14 R13

NOTE: There are 13 rows where a click is must. Therefore, click in each row in the top-down order.

Sub-rectangle Transfer  
If you want to move a sub-rectangle from a group A to a group B, then click for group number for groups A & B and then for sub-rectangle number (sub-rectangle number and group number are given above). If you do not want to move any sub-rectangle, click at 0 for sub-rectangle number.

1. Group A ☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5  
2. Group B ☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5  
3. Rectangle No. ☐ 0 ☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5 ☐ 6 ☐ 7 ☐ 8 ☐ 9 ☐ 10 ☐ 11 ☐ 12 ☐ 13 ☐ 14 ☐ 15 ☐ 16

POSITION OF GROUPS  
Choose position of each group one by one by clicking only once in each row and each column.

4. CENTRE ☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5  
5. LEFT ☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5  
6. ABOVE ☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5  
7. RIGHT ☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5  
8. BELOW ☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5

SPIRALS  
Choose SPIRAL for each group one by one by clicking only once in each row.

9. CENTRE Gr ☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5 ☐ 6 ☐ 7 ☐ 8  
10. LEFT Gr ☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5 ☐ 6 ☐ 7 ☐ 8  
11. ABOVE Gr ☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5 ☐ 6 ☐ 7 ☐ 8  
12. RIGHT Gr ☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5 ☐ 6 ☐ 7 ☐ 8  
13. BELOW Gr ☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5 ☐ 6 ☐ 7 ☐ 8

FIGURE 3. GUI for the CPAR

The formation of groups is done by using an algorithm but it can sometimes happen that one is not pleased with the formed groups. Therefore, we kept an option in the GUI (refer to Figure 3) which enables us to move a sub-rectangle from one group to another. For example, in Figure 4,  $R_1$  of Group1 is moved to Group3.

**Remark 2.** *In Processing an array always starts from zero, therefore in programming all numbering begins with zero.*

#### 4. The position of groups

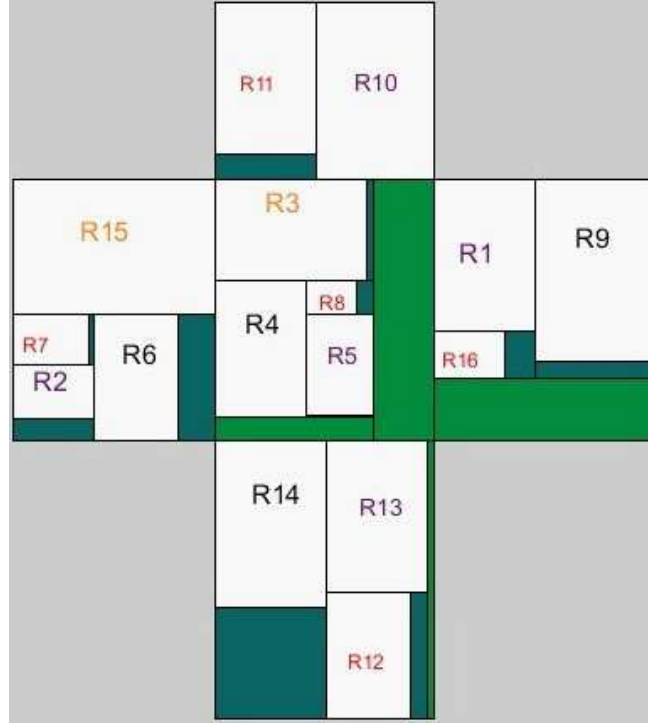


FIGURE 4. Moving a sub-rectangle from one group to other using CPAR

Five groups are required to form a plus-shape arrangement. These groups are formed on the basis of weighted adjacency matrix. There are only five positions for groups which can be selected directly from the GUI.

#### 5. Assigning a spiral for each group

A group can be constructed in any of the eight ways, we say it is constructed with any of the eight spirals. The eight spirals are shown in Figure 5. The spiral used for each group can be chosen directly from the GUI.

### 4. CPAR CONSTRUCTION

In this section all tabs of the code are explained one by one in the order in which they occur.

4.1. **Getting input.** In the tab *input*, we import the input from the external file *input.txt* by calling a function *input()*.

4.2. **Initial adjacency pairs.** In the tab *initialadjacency* by calling function *initialadjacency()* all the initial adjacency pairs are obtained. For details, refer to Section 6.

4.3. **Groups.** Once we have the initial adjacency pairs, by using the tab *groups*, the required groups and their members are obtained. For details, refer to Section 7.

4.4. **Change of a sub-rectangle.** The function of the tab *changingsub-rectangle* is to move a sub-rectangle from one group to another and simultaneously it revises the position of the sub-rectangle in the corresponding array.

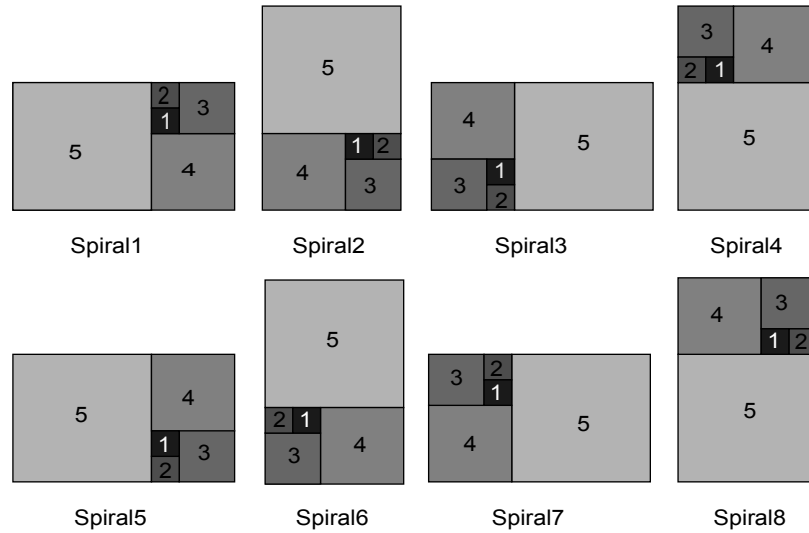


FIGURE 5. 8 different spirals

4.5. **Arranging the members of each group in ascending order.** The tab *arrangingsizes* considers each group one by one and then arranges its members in the increasing order according to their areas.

4.6. **Obtaining a spiral-based plus-shape arrangement( $P_S^A$ ).** In the tab *plussshape*, by calling function *plussshape()* the required  $P_S^A$  is constructed and displayed on the screen. This function has two parts, the first one does the necessary calculations while the second one deals with the construction of a  $P_S^A$ .

First part: **Interchanging the length and height of sub-rectangles and calculating the area of  $P_S^A$**

This part has many steps, some of which may call some other functions. The details of these newly-defined functions are provided later.

1. Set  $i = 0$
2. Consider the  $(i + 1)^{th}$  group

**Remark 3.** For all the upcoming steps, the 1<sup>st</sup>, 2<sup>nd</sup>, 3<sup>rd</sup>, 4<sup>th</sup> and 5<sup>th</sup> groups represent the central, left, upper, right and lower groups respectively. Let  $l_i$  and  $h_i$  be the length and height of  $i^{th}$  sub-rectangle.

3. Interchanging length and height of the 1<sup>st</sup> member of each group

This step is performed to reduce the area of  $P_S^A$ . It works well in most cases but sometimes it might work adversely.

For  $i = 0, 1$  or  $3$ , namely, for 1<sup>st</sup>, 2<sup>nd</sup> or 4<sup>th</sup> group if  $l_1 < h_1$ , we swap  $l_1$  and  $h_1$ . This is to reduce height of  $P_S^A$ .

For  $i = 2$  or  $4$ , namely, for 3<sup>rd</sup> or 5<sup>th</sup> groups, if  $l_1 > h_1$ , we swap  $l_1$  and  $h_1$ . This is to reduce length of  $P_S^A$ .

4. Interchanging length and height of the members of  $(i + 1)^{th}$  group

If *spiral1*, *spiral2*, *spiral5* or *spiral6* is used for the  $(i + 1)^{th}$  group then we call function *shape1()*. If *spiral3*, *spiral4*, *spiral7* or *spiral8* is used for the  $(i + 1)^{th}$  group then we call function *shape2()*.

The functions *shape1()* and *shape2()* swap the length and height of all the members of each group, if required to reduce the size of inner extra spaces.

5. *Calculating the length and height of the  $(i + 1)^{th}$  group*

All the functions defined in this step compute the length and height of corresponding groups. If *spiral1*, *spiral2*, *spiral5* or *spiral6* is used for the  $(i + 1)^{th}$  group and if

5.1.  $i = 0$ , we call function *LandH1.1()* otherwise for remaining spirals we call function *LandH2.1()*.

5.2.  $i = 1$ , we call function *LandH1.2()* otherwise for remaining spirals we call function *LandH2.2()*.

5.3.  $i = 2$ , we call function *LandH1.3()* otherwise for remaining spirals we call function *LandH2.3()*.

5.4.  $i = 3$ , we call function *LandH1.4()* otherwise for remaining spirals we call function *LandH2.4()*.

5.5.  $i = 4$ , we call function *LandH1.5()* otherwise for remaining spirals we call function *LandH2.5()*.

6. If  $i = 4$ , we go to the next step otherwise we increase  $i$  by one and go to step 2.

7. *Computing the area of  $P_S^A$*

Going through the details of all the functions used in the first part is lengthy and tedious; therefore to understand the concept of all these functions, we shall elaborate on the steps of only two functions, namely, *shape1()* and *LandH1.1()*. These two functions are given in Sections 8 and 9.

Second part: **Drawing  $P_S^A$**

1. Let  $i = 0$ .

2. Consider the  $(i + 1)^{th}$  group.

3. *Calculating the length and height of the inner extra spaces*

Let  $R_S^A$  represents the rectangular block used to generate  $P_S^A$ . The construction of a  $R_S^A$  is explained in [4]. If  $R_j$  is drawn to the left or right of  $R_S^A(j - 1)$  and  $l_j$  is greater than the length of  $R_S^A(j - 1)$ , we draw an inner extra space to the right of  $R_S^A(j - 1)$ . To obtain the starting point of  $(i + 1)^{th}$  group (e.g. second group), the length of the inner extra space is subtracted from  $x$ .

If *spiral1*, *spiral2*, *spiral5* or *spiral6* is used for the  $(i + 1)^{th}$  group, we call function *shape1.1()*.

If *spiral3*, *spiral4*, *spiral7* or *spiral8* is used for the  $(i + 1)^{th}$ , we call function *shape2.1()*.

Here the functions *shape1.1()* and *shape2.1()* calculate the length and height of some of the inner extra spaces of the  $(i + 1)^{th}$  group.

4. *Obtaining starting point of each group and drawing the outer extra spaces*

In this step first we compute the starting point of the  $(i + 1)^{th}$  group and then we draw an outer extra space if required.

For  $i = 0, 1, 2, 3, 4$ , we call functions *extra1.1*( $p_1, p_2$ ), *extra1.2*( $p_1, p_2, p_3, p_4$ ), *extra1.3*( $p_1, p_2, p_3$ ), *extra1.4*( $p_1, p_2$ ) or *extra1.5*( $p_1, p_2$ ) respectively. Here  $p_1, p_2, p_3$  and  $p_4$  are variables

which are passed to the corresponding function and the value of  $p_1, p_2, p_3$  and  $p_4$  may be different for each spiral.

5. *Drawing the  $(i + 1)^{th}$  group*

Corresponding to the *spiral1*, *spiral2*, *spiral3*, *spiral4*, *spiral5*, *spiral6*, *spiral7* and *spiral8*, we call functions *shape1.2()*, *shape2.2()*, *shape3.2()*, *shape4.2()*, *shape5.2()*, *shape6.2()*, *shape7.2()* and *shape8.2()* respectively.

Each of these functions draws the corresponding group following the corresponding spiral at the starting point obtained in step 4.

6. *If  $i = 4$  we move to the next step otherwise we go back to the second step.*

Again explaining all the functions is an extensive and verbose process so for clarification of the functions, *shape1.1()*, *extra1.2( $p_1, p_2, p_3, p_4$ )* and *shape1.2()* are discussed. These functions are given in Sections 10, 11 and 12 respectively.

**Remark 4.** *For the upcoming computations, it is not feasible to draw sub-rectangles again and again, therefore we allocate the sub-rectangles for the required computations. Allocating does not mean drawing, it means drawing virtually. Allocating the sub-rectangles instead of drawing them reduces the complexity of the code and the code consumes less time for displaying the final output.*

**4.7. Obtaining the final adjacency pairs.** The tab *adjacencypairs* computes the adjacency pairs among the components of each group and among members of different groups.

Function: **adjacencypairs()**

1. *Set  $i = 0$  and  $j = 0$  where  $i$  and  $j$  are variables.*

2. *Obtaining adjacency pairs of the first group*

To obtain adjacency pairs of two different groups, after allocating each member we calculate the length and height of corresponding  $R_S^A$ .

If  $j = 0$ ,

If *spiral1*, *spiral2*, *spiral5* or *spiral6* is used, the first and second sub-rectangle is drawn one above the other. By calling function *adjacencycal1()* we compute the length and height of  $R_S^A$  after allocating each member.

If *spiral3*, *spiral4*, *spiral7* or *spiral8* is used, the first and second sub-rectangle is drawn side by side. By calling function *adjacencycal2()* we compute the length and height of  $R_S^A$  after allocating each member.

If  $i = 0$ , to calculate adjacency pairs of the first group we call function *adjacency1()*.

3. *Obtaining adjacent pairs of  $(j + 1)^{th}$  group*

If  $i = j + 1$ ,

For  $(j + 1)^{th}$  group, after allocating each member we compute the length and height of corresponding  $R_S^A$  by calling any of the required functions *adjacencycal1()* or *adjacencycal2()*.

We compute adjacency pairs of  $(j + 1)^{th}$  group by calling function *adjacency1()*.

4. *Obtaining adjacency pairs among the first and second group*

If  $i = 0$ ,



4.1 If  $j = 0$ , we obtain those members (and their heights) of the first group which can be adjacent to some members of the second group by calling function  $\text{adjacencyHeight}(p_3, p_4)$ .

The values of  $p_3$  and  $p_4$  are different for each spiral. Since the second group is drawn to the left of the first group, adjacency among the members of these two groups is obtained by comparing their heights.

4.2 Obtaining the members (and their heights) of the second group which can be adjacent to the members of first group and then computing the adjacency pairs among these two groups

If  $j = 1$ , we first check which spiral is used and then call function  $\text{adjacencyHeight}(p_3, p_4)$  (the values of  $p_3$  and  $p_4$  are different for each spiral).

Afterwards we call function  $\text{findingadjacencies}(p_1, p_2)$ . This function computes the adjacency pairs among the members of two different groups. Here it computes the adjacency pairs among the members of the first and second group. The value of  $p_1$  and  $p_2$  can be same or different for different spirals.

#### 5. *Obtaining adjacent members of the first and third group*

In this case we check if  $i = 1, j = 0$  for the first group and  $j = 2$  for the third group. To obtain steps 5.1 and 5.2, we replace function  $\text{adjacencyHeight}(p_3, p_4)$  by function  $\text{adjacencyLength}(p_3, p_4)$  in the steps 4.1 and 4.2.

Since the third group is drawn above the first group, we compare the lengths of the members of these two groups and that is why we replaced  $\text{adjacencyHeight}(p_3, p_4)$  by  $\text{adjacencyLength}(p_3, p_4)$ .

#### 6. *Obtaining adjacent members of the first and fourth group*

In this case we check if  $i = 2, j = 0$  for the first group and  $j = 3$  for the fourth group. Steps 6.1 and 6.2 are same as the steps 4.1 and 4.2.

#### 7. *Obtaining adjacent members of the first and fifth group*

In this case we check if  $i = 3, j = 0$  for the first group and  $j = 4$  for the fifth group. Steps 7.1 and 7.2 are same as the steps 5.1 and 5.2.

8. If  $j < 4$ , we increase  $j$  by one and go to step 2. If  $j = 4$ , we increase  $i$  by one. If  $i < 4$ , we consider  $j = 0$  and go to step 2 otherwise stop the process.

Explaining all the functions is an extensive and verbose process so for clarification of the functions  $\text{adjacencycall}()$ ,  $\text{adjacency1}()$ ,  $\text{adjacencyHeight}(p_3, p_4)$  and  $\text{findingadjacency}(p_1, p_2)$  are discussed. All these functions are given in Sections 13, 14, 15 and 18 respectively.

**4.8. Obtaining covariants associated with the graphs.** The next five tabs are *adjacencymatrix*, *distance*, *cutvertex*, *eccentricity* and *MOI*. These tabs computes graph covariants associated with the obtained  $P_S^A$ . For the definition of graph terminology used in this section, refer [5].

##### 1. *Function adjacencymatrix()*

1.1 This function computes the *adjacency matrix* from obtained adjacency pairs.

1.2 From the adjacency pairs it calculates *degree of connectivity* of the graph.

1.3 From the adjacency matrix, it obtains the *degree* of each vertex of graph of  $P_S^A$  i.e.  $G_S^P$  and then the mean, standard deviation, maximum and minimum of all degrees.

##### 2. *Function distance()*

This function calculates the *distance* between any two vertices of  $G_S^P$  and then the mean, standard deviation, maximum and minimum of all distances.

3. *Function cutvertex()*

This function computes all the *cut vertices* and *cut pairs* of  $G_S^P$ .

4. *Function eccentricity()*

This function first provides the *eccentricity* of each vertex of  $G_S^P$  and then calculates the *diameter*, *radius* and *centre* of  $G_S^P$ . At the end, it computes the mean and standard deviation of all eccentricities.

5. *Function MOI()*

We compute moments of  $G_S^P$  relative to each vertex by the following two ways:

1. by considering the weight of each sub-rectangle equal to its area,
2. by considering the weight of each sub-rectangle as one unit.

The moments of inertia generally provide a more accurate measure of the centre of  $G_S^P$  than eccentricity, even when the graph is equipped only with the trivial weighting.

After having the first-order moments and the moments of inertia of  $G_S^P$  relative to each vertex, we compute the mean, standard deviation, maximum and minimum of all moments.

**4.9. Obtaining eigenvalues.** From the tab *eigenvalue*, we obtain the eigenvalues of the adjacency matrix and its characteristic polynomial, by using inbuilt library *Jama*. Afterwards the maximum and minimum of all eigenvalues are computed.

**4.10. Drawing the graph.** Using the tab *graph*, we draw the  $G_S^P$  on the same screen on which  $P_S^A$  is displayed.

**4.11. Print.** Using the tab *print*, all the results are displayed in an external file *output.txt*. A list of these results is given in next Section. In this tab the following calculations are made:

1. Using the inbuilt library *jgraphT*, we obtain the *chromatic number* of  $G_S^P$ .
2. By means of the paths for calculating distances, we compute a *shortest path* between each pair of vertices of  $G_S^P$  using the Floyd's algorithm given in Section 19.
3. We compute whether  $G_S^P$  is *bipartite* or not.

**4.12. Calling all functions.** In the tab *CPAR* the function *setup()* calls all the main functions defined in Sections 4.1 to 4.11.

## 5. OUTPUT OF CPAR

When we run CPAR, a  $P_S^A$  and its graph are displayed on the screen. In addition, some important covariants, like the area of  $P_S^A$ , spirals used for the central, left, upper, right and lower  $R_S^A$ , the minimum and maximum moments of inertia, the radius and diameter of  $G_S^P$  are also displayed. At the same time, a new file output.txt is obtained, which contains the following results:

1. The length and height of each sub-rectangle
2. All the five groups and their members
3. The number of inner and outer extra spaces
4. The area of  $P_S^A$
5. The total area of all the extra spaces

6. The adjacency matrix
7. The number of edges
8. The degrees of all sub-rectangles, their mean, standard deviation, dispersion, maximum and minimum.
9. The eigenvalues of adjacency matrix and the corresponding polynomial. Also, the minimum and maximum of all eigenvalues.
10. Whether the  $G_S^P$  is bipartite or not
11. The distance matrix and the mean, standard deviation, dispersion, maximum and minimum of all distances
12. A shortest path between each pair of sub-rectangles
13. All the cut vertices and cut pairs
14. The eccentricities of all sub-rectangles, their mean, standard deviation, and dispersion. The radius, diameter and centre of  $G_S^P$
15. The moments and their mean, standard deviation, dispersion, maximum and minimum.
16. The chromatic number

5.1. **Libraries.** To run CPAR, the following libraries are required:

#### 7.1 jgraphT

This library is used to calculate the chromatic number.

#### 7.2 Jama

This library is used to compute the eigenvalues.

### 6. INITIAL ADJACENCY PAIR ALGORITHM

This algorithm calculates the initial adjacency pairs from a given  $A_T$ .

1. Let  $A_T = [a_{ij}]_{n \times n}$ ,  $M = \max\{a_{ij}\}$  where  $i = 1, \dots, n$ ;  $j = 1, \dots, n$  and  $n$  be the number of sub-rectangles. Initially  $M = 10$ ,  $j = 1$ .
2. Consider the  $j^{th}$  row.
3. If it corresponds to any sub-rectangle which is covered in any of the obtained initial adjacency pairs we skip this row otherwise we obtain all the pairs of sub-rectangles corresponding to number  $M$  in the  $A_T$  and consider them as adjacency pairs.
4. If all the sub-rectangles are covered in the obtained adjacency pairs, terminate the algorithm; otherwise go to the next step.
5. Increase  $j$  by one.
6. If  $j < n + 1$ , go to step 2.
7. If  $j = n + 1$ , reduce  $i$  by one, consider  $j = 1$  and go to step 2.

### 7. ALGORITHM FOR THE GROUPING OF SUB-RECTANGLES

This algorithm computes groups from the initial adjacency pairs. Here in particular, the algorithm is given for obtaining five groups which will be used to obtain a  $P_S^A$ . If the number of groups is greater or less than five, the initial adjacency pairs will require updating. Therefore this algorithm does not only obtain five groups, but also revises the initial adjacency pairs. Here are the steps of the algorithm:

1. Let the number of groups be  $i$  and initially  $i = 1$ .
2. Obtaining the 1<sup>st</sup> member of the  $i^{th}$  group
  - a. Consider each sub-rectangle one by one from the given list of sub-rectangles.

b. Select the sub-rectangle which does not exist in any of the groups obtained so far.

c. Now regard this sub-rectangle as the  $1^{st}$  member of the  $i^{th}$  group.

**Note:** To start the process of forming groups, we consider the  $1^{st}$  sub-rectangle as the  $1^{st}$  member of the  $1^{st}$  group.

3. Forming the  $i^{th}$  group

a. Among the adjacency pairs, we find those sub-rectangles which are adjacent to the  $1^{st}$  member of the group.

b. Then we include these sub-rectangles as members of the group.

c. If newly included members are adjacent to other sub-rectangles from the initial adjacency pairs, we add those sub-rectangles to the group.

d. We repeat Step 3.c until the remaining sub-rectangles from the initial adjacency pairs are adjacent to any other member of the group.

e. When all members along with their adjacent sub-rectangles are included in the group we stop the process.

4. Review all the remaining sub-rectangles. If the number of sub-rectangles in the given list is equal to the number of all sub-rectangles included in the groups (i.e., if all the sub-rectangles are included in the groups)

a. Then proceed to step 5

b. Otherwise increase  $i$  by one and go to step 2 to form another group.

5. To obtain a plus-shape tiling five groups are required, therefore

a. If  $i = 5$ , we stop.

b. If  $i < 5$ , we go to step 6 to increase the number of groups.

c. If  $i > 5$ , we go to step 7 to reduce the number of groups.

6. When the number of groups is less than 5 ( $i < 5$ )

a. We search for the group having the maximum number of sub-rectangles as members.

b. If there is more than one group we consider the one which comes first.

c. Let this group be named  $G$ .

d. We look in the  $A_T$  for a pair of elements of  $G$  with minimum weight. If there is more than one pair we consider the one which comes first.

e. Let the sub-rectangles from this pair be  $(R_i, R_j)$ .

f. Now we update the initial adjacency pairs by deleting all those pairs which have any member in common with  $G$ .

g. Split  $G$  into two parts, so that  $i$  gets increased by one. Splitting has the effect of forming two new groups:

(i).  $G_1$  which contains  $R_i$

(ii).  $G_2$  which contains  $R_j$ .

h. To find the members of  $G_1$  and  $G_2$ , we look at the weight of each member of  $G$  corresponding to  $R_i$  and  $R_j$ .

i. If the obtained weight of any member corresponding to  $R_i$  is greater than the weight corresponding to  $R_j$  then this sub-rectangle forms an adjacency pair with  $R_i$  and we consider it as a member of  $G_1$  otherwise it forms an adjacency pair with  $R_j$  and we consider it as a member of  $G_2$ .

j. Repeat step 6.i until  $G_1 \cup G_2 = G$ .

k. Now  $G$  is replaced by  $G_1$  and  $G_2$ . Also,  $i$  got increased by one.

1. Go to step 5.
7. When the number of groups is greater than 5 ( $i > 5$ )
  - a. From among the  $i$  groups, we choose two having a minimum number of members.
  - b. Let these groups be named  $G_1$  and  $G_2$ .
  - c. Combine  $G_1$  and  $G_2$  to form a new group.
  - d. We look in the  $A_T$  for a pair of elements of  $G_1$  and  $G_2$  with maximum weight. If there is more than one pair we consider the one which comes first.
  - e. Consider this pair to be an adjacency pair.
  - f. Now  $G_1$ ,  $G_2$  together form a new group. Also,  $i$  got reduced by one.
  - g. Go to step 5.

**Note:** The steps 6.h, 7.d, 7.e, 7.f are meant to update the initial adjacency pairs. They have nothing to do with the formation of groups.

## 8. FUNCTION SHAPE1()

This function swaps the length and height of members of a group when *spiral1*, *spiral2*, *spiral5* or *spiral6* is used for the corresponding group.

1. When  $R_2$  is going to be allocated above  $R_1$ 
  - 1.1 Calculating the area of extra spaces  
 If  $\ell_1 > \ell_2$ , then  $A_O = (\ell_1 - \ell_2) \times h_2$  otherwise  $A_O = (\ell_2 - \ell_1) \times h_1$   
 If  $\ell_1 > h_2$ , then  $A_I = (\ell_1 - h_2) \times \ell_2$  otherwise  $A_I = (h_2 - \ell_1) \times h_1$ .
  - 1.2 Interchanging the length and height (if required)  
 If  $A_O > A_I$ , then swap  $\ell_2$  and  $h_2$ , i.e.,  
 $\text{temp} = \ell_2, \ell_2 = h_2, h_2 = \text{temp}$ .
  - 1.3 Calculating  $L_2$  and  $H_2$   
 Initially  $L_1 = l_1, H_1 = h_1$ . Now  $L_2 = \max(\ell_2, \ell_1), H_2 = H_1 + h_2$ .
2. When  $R_i$  is going to be allocated to the left or right of  $R_{i-1}$   
 We are calculating the heights of  $R_S^A$  only because either  $H_i \geq h_i$  or  $H_i < h_i$  but  $L_i$  is simply  $L_{i-1} + l_i$ . Also, for further calculations, when  $R_i$  is allocated to the left or right of  $R_{i-1}$ , only  $H_i$  will be used.
  - 2.1 Calculating  $H_i$   
 $H_i = H_{i-2} + h_{i-1}$ .
  - 2.2 Calculating the area of extra spaces  
 If  $H_i > h_i$ ,  $A_O = (H_i - h_i) \times \ell_i$  otherwise  $A_O = (h_i - H_i) \times L_{i-1}$   
 If  $H_i > \ell_i$ ,  $A_I = (H_i - \ell_i) \times h_i$  otherwise  $A_I = (\ell_i - H_i) \times L_{i-1}$ .
  - 2.3 Interchanging the length and height (if required)  
 If  $A_O > A_I$ , then swap  $\ell_i$  and  $h_i$ .
  - 2.4 Updating  $H_i$   
 If  $h_i > H_i$ , then  $H_i = h_i$ .
3. When  $R_i$  is going to be allocated above or below  $R_{i-1}$ 
  - 3.1 Calculating  $L_i$   
 $L_i = L_{i-2} + \ell_{i-1}$ .
  - 3.2 Calculating the area of extra spaces  
 If  $L_i > \ell_i$ ,  $A_O = (L_i - \ell_i) \times h_i$  otherwise  $A_O = (\ell_i - L_i) \times H_{i-1}$   
 If  $L_i > h_i$ ,  $A_I = (L_i - h_i) \times \ell_i$  otherwise  $A_I = (h_i - L_i) \times H_{i-1}$ .
  - 3.3 Interchanging the length and height (if required)

If  $A_O > A_I$ , then swap  $\ell_i$  and  $h_i$ .

3.4 Updating  $L_i$

If  $\ell_i > L_i$ , then  $L_i = \ell_i$ .

4. Keep repeating steps 2 and 3 until all members of the corresponding group are allocated.

## 9. FUNCTION *LandH*1.1()

This function calculates the length and height of the first group when *spiral1*, *spiral2*, *spiral5* or *spiral6* is used for the corresponding group.

1. If the number of sub-rectangles in the 1<sup>st</sup> group is greater than one

1.1 If the number of sub-rectangles in this group is an even number then

$$L^1 = L_{n-1} \text{ and } H^1 = H_{n-1} + h_n$$

In this case if *spiral1*, *spiral2*, *spiral5* or *spiral6* is used, then  $R_n$  will be allocated above or below  $R_S^A(n-1)$ . Therefore after allocating  $R_n$ , the length of the group will be  $L_{n-1}$  but for the height,  $H_{n-1}$  will be augmented by  $h_n$ .

1.2 If the number of sub-rectangles in this group is an odd number then

$$L^1 = L_{n-1} + l_n \text{ and } H^1 = H_{n-1}$$

In this case if *spiral1*, *spiral2*, *spiral5* or *spiral6* is used, then  $R_n$  will be allocated to the left or right of  $R_S^A(n-1)$ . Therefore after allocating  $R_n$ , the height of the group will be  $H_{n-1}$  but for the length,  $L_{n-1}$  will be augmented by  $l_n$ .

2. If the number of sub-rectangles in the 1<sup>st</sup> group is equal to one then  $L^1$  and  $H^1$  are  $\ell_1$  and  $h_1$  respectively.

## 10. FUNCTION *SHAPE*1.1()

This function computes the length or height of the inner extra space corresponding to each member of every group.

1. Calculating  $L_2$  and  $H_2$

Initially  $L_1 = l_1$ ,  $H_1 = h_1$ . Now  $L_2 = \max(\ell_2, \ell_1)$ ,  $H_2 = H_1 + h_2$ .

2. Calculating the length of inner extra space after allocating  $R_2$

$$\text{length}(\text{inner extra space}) = |\ell_2 - \ell_1|$$

This value will be used while drawing the corresponding extra space.

3. When  $R_i$  is going to be allocated to the left or right of  $R_{i-1}$

3.1  $H_i = H_{i-2} + h_{i-1}$ .

3.2 If  $h_i > H_i$  then  $\text{height}(\text{inner extra space}) = h_i - H_i$  otherwise we consider it as 0 (the explanation for considering the height of inner extra space only when  $h_i > H_i$  is given in upcoming function).

4. When  $R_i$  is going to be allocated above or below  $R_{i-1}$

4.1  $L_i = L_{i-2} + \ell_{i-1}$ .

4.2 If  $\ell_i > L_i$ , then  $\text{length}(\text{inner extra space}) = \ell_i - L_i$  otherwise we consider it as 0.

5. Keep repeating steps 3 and 4 until all members of the corresponding group are covered.

## 11. FUNCTION *EXTRA*1.2( $p_1, p_2, p_3, p_4$ )

This function calculates the starting point of the second group and draw an outer extra space below the first or the second group.

1. Obtaining the starting point of the second group

Let  $(x, y)$  is the upper left corner of the first group and initially the starting point of the second group.

When a member  $R_i$  of the second group is drawn to the right of  $R_S^A(i-1)$ , it overlaps with some members of the first group (e.g. if *spiral2* is used for the second group, its second member is drawn at position  $(x + l_1, y)$ ; clearly this member overlaps with some members of the first group). Therefore we deduct the lengths of all  $R_i$ , drawn to the right of corresponding  $R_S^A(i-1)$ , from  $x$  to obtain the starting point of second group.

In function `extra1.2( $p_1, p_2, p_3, p_4$ )`,  $p_2$  represents the first value of  $i$  for which  $R_i$  is drawn to the right of  $R_S^A(i-1)$  and after  $R_{p_2}$  every fourth member (if exist) is drawn to the right of  $R_S^A$ . Using  $p_2$ , we compute the lengths of all  $R_i$  drawn to the right of corresponding  $R_S^A(i-1)$  and subtract them from  $x$ .

Also  $R_1$  of the second group overlaps with some members of the first group. Therefore corresponding  $l_1$  is subtracted from  $x$ . For some spirals,  $R_2$  is drawn to the left or to the right of  $R_1$  (e.g. *spiral2*, see Figure 3). In this case,  $p_1 = 0$  and  $l_1$  is subtracted from  $x$ . For some spirals,  $R_2$  is drawn above or below  $R_1$  (e.g. *spiral1*, see Figure 3). In this case  $p_1 = 1$  and  $L_2$  is deducted from  $x$ .

For the second group, we require  $(x, y)$  should be its upper right corner. When a member  $R_i$  of the second group is drawn above  $R_S^A(i-1)$ ,  $(x, y)$  would not remain upper right corner of the second group. To obtain this position, the heights of all those  $R_i$  which are drawn above corresponding  $R_S^A(i-1)$ , are added to  $y$ .

Here  $p_3$  represents the first value of  $i$  for which  $R_i$  is drawn above  $R_S^A(i-1)$  and after  $R_{p_3}$  every fourth member (if exist) is drawn above some  $R_S^A$ .

If a member  $R_j$  of the second group is drawn to the left or right of  $R_S^A(j-1)$  and  $l_j$  is greater than the length of  $R_S^A(j-1)$ , we draw an inner extra space to the right of  $R_S^A(j-1)$ . This extra space is a virtual part of  $R_S^A(j-1)$  and it virtually increases the length of  $R_S^A(j-1)$ . To obtain the starting point of the second group, the length of the inner extra space is subtracted from  $x$ . The length of inner extra spaces has already been obtained in the previous function.

Here  $p_1$  also represents the first value of  $i$  for which  $R_i$  is drawn above or below  $R_S^A(i-1)$  where  $l_i$  is greater than the length of  $R_S^A(i-1)$ . For this case we have considered the upper and lower sides only, therefore every second member after  $R_{p_1}$  is drawn either above or below some  $R_S^A$ . Therefore using  $p_1$  the length of all the inner extras are obtained and subtracted from  $x$ .

After all these calculations, obtained value of  $x$  and  $y$  gives the starting point of the second group.

**Remark 5.** *We have not considered the members  $R_i$  whose height is greater than the height of  $R_S^A(i-1)$  because the inner extra space is always drawn below a  $R_S^A$ . And to obtain the starting point, the heights of only those spaces which are drawn above some  $R_S^A$  are subtracted from  $y$ .*

*In case of the third group, we have not considered the cases when the length of members  $R_i$  is greater than the length of  $R_S^A(i-1)$  because in these cases, inner extra space is always drawn to the right of  $R_S^A(i-1)$ . And to get the starting point, the lengths of only those  $R_i$  which are drawn to the left of  $R_S^A(i-1)$  are added to  $x$ .*

Similarly for the first, fourth and fifth groups, any of the cases when the length or the height of  $R_i$  is greater than the length (or the height) of  $R_S^A(i-1)$ , have not considered.

## 2. Drawing an outer extra space

If  $H^1 > H^2$ , we draw an outer extra space below the second group such that its lower left vertex is the upper left vertex of the extra space. The length and height of this extra space is  $L^2$  and  $H^1 - H^2$  respectively having position  $(x - L^2, y + H^2)$ .

If  $H^1 < H^2$ , we draw an outer extra space below the first group such that its lower left vertex is the upper left vertex of the extra space. The length and height of this extra space is  $L^1$  and  $H^2 - H^1$  respectively having position  $(x, y + H^1)$ .

**Remark 6.** A particular colour is used for all the outer extra spaces to distinguish them from others spaces. Also after drawing an outer extra space, the number of outer extra spaces is increased by one so that in the end the total number of outer extra spaces is achieved.

## 12. FUNCTION SHAPE1.2()

This function is used to draw all the members of any group when *spiral1* is used for the group. Suppose  $i^{th}$  group is going to be drawn and its starting point is  $(x, y)$ .

### 1. Drawing $R_1$

$R_1$  is drawn with the length and height  $\ell_1$  and  $h_1$  respectively at position  $(x, y)$ .

If the number of members of  $i^{th}$  group is greater than one, move to the next step otherwise stop here.

**Remark 7.** After drawing each member, its name is printed at its centre and a particular colour is used for all the members of each group to distinguish them from the extra spaces.

### 2. Drawing $R_2$ and the inner extra space

$R_2$  is drawn with the length and height  $\ell_2$  and  $h_2$  respectively at position  $(x, y - h_2)$ .

If  $\ell_2 < \ell_1$ , then an inner extra space is drawn to the right side of  $R_2$  with the length and height obtained in the function *shape1.1()* at position  $(x + \ell_2, y - h_2)$ .

If  $\ell_2 > \ell_1$ , then an inner extra space is drawn to the right  $R_1$ , with the length and height obtained in the function *shape1.1()* at position  $(x + \ell_1, y)$ .

**Remark 8.** A particular colour is used in all the inner extra spaces to distinguish them from other spaces. Also after drawing an inner extra space the number of inner extra spaces is increased by 1 so that in the last the total number of inner extra spaces will be achieved.

### 3. Calculating $L_2$ and $H_2$

Initially  $L_1 = \ell_1$ ,  $H_1 = h_1$ . Now  $L_2 = \max(\ell_2, \ell_1)$ ,  $H_2 = H_1 + h_2$ .

### 4. Obtaining a position for $R_3$

Since upper left vertex of  $R_3$  should be upper right vertex of  $R_S^A(2)$ , we subtract  $h_2$  from  $y$ , i.e.  $y = y - h_2$ , to obtain position of  $R_3$ .

### 5. Drawing $R_i$ to the right of $R_S^A(i-1)$

Add  $L_i$  to  $x$  to obtain position of  $R_i$ . We draw  $R_i$  with length  $\ell_i$  and height  $h_i$  at position  $(x, y)$ .



If  $h_i < H_i$ , we draw an inner extra space at position  $(x, y + h_i)$  with length  $\ell_i$  and height  $H_i - h_i$ . If  $h_i > H_i$ , we draw an inner extra space at position  $(x - L_i, y + H_i)$  with length  $L_i$  and height  $h_i - H_i$ .

In this case we update  $H_i$  by  $H_i = h_i$ .

6. *Drawing  $R_i$  below  $R_S^A(i - 1)$*

Subtract  $L_i$  from  $x$  and add  $H_i$  to  $y$  to obtain position of  $R_i$ . We draw  $R_i$  with length  $\ell_i$  and height  $h_i$  at position  $(x, y)$ .

If  $\ell_i < L_i$ , we draw an inner extra space at position  $(x + \ell_i, y)$  with length  $L_i - \ell_i$  and height  $h_i$ .

If  $\ell_i > L_i$ , we draw an inner extra space at position  $(x + L_i, y - H_i)$  with length  $\ell_i - L_i$  and height  $H_i$ .

In this case we update  $L_i$  by  $L_i = \ell_i$ .

7. *Drawing  $R_i$  to the left side of  $R_S^A(i - 1)$*

Subtract  $\ell_i$  from  $x$  and subtract  $H_i$  from  $y$  to obtain position of  $R_i$ . We draw  $R_i$  with length  $\ell_i$  and height  $h_i$  at position  $(x, y)$ .

If  $h_i < H_i$ , we draw an inner extra space at position  $(x, y + h_i)$  with length  $\ell_i$  and height  $H_i - h_i$ . If  $h_i > H_i$ , we draw an inner extra space at position  $(x + \ell_i, y + H_i)$  with length  $L_i$  and height  $h_i - H_i$ .

In this case we update  $H_i$  by  $H_i = h_i$ .

8. *Drawing  $R_i$  above  $R_S^A(i - 1)$*

Subtract  $h_i$  from  $y$  to obtain position of  $R_i$ . We draw  $R_i$  with length  $\ell_i$  and height  $h_i$  at position  $(x, y)$ .

If  $\ell_i < L_i$ , we draw an inner extra space at position  $(x + \ell_i, y)$  with length  $L_i - \ell_i$  and height  $h_i$ . If  $\ell_i > L_i$ , we draw an inner extra space at position  $(x + L_i, y + h_i)$  with length  $\ell_i - L_i$  and height  $H_i$ .

In this case we update  $L_i$  by  $L_i = \ell_i$ .

9. *Keep repeating the steps 5, 6, 7 and 8 until all the members are drawn.*

### 13. FUNCTION ADJACENCYCAL1()

This function calculates the length (or the height) of  $R_S^A$  after allocating each sub-rectangle for each group when *spiral1*, *spiral2*, *spiral5* or *spiral6* is used for the corresponding group.

1. Initially  $L_1 = l_1$ ,  $H_1 = h_1$ . Now  $L_2 = \max(\ell_2, \ell_1)$ ,  $H_2 = H_1 + h_2$ .

2. When  $R_i$  is going to be allocated to the left or right of  $R_{i-1}$

$H_i = H_{i-2} + h_{i-1}$ . If  $h_i > H_i$  we have  $H_i = h_i$ .

3. When  $R_i$  is going to be allocated above or below  $R_{i-1}$  member

$L_i = L_{i-2} + \ell_{i-1}$ . If  $\ell_i > L_i$  we have  $L_i = \ell_i$ .

4. Keep repeating steps 2 and 3 until all the members of corresponding group are covered.

### 14. FUNCTION ADJACENCY1()

This function calculates the adjacency pairs among each group.

1. *Obtaining adjacency of the first member with other members of the same group*

1.1. If  $n = 2$  then  $R_1$  will be adjacent to  $R_2$ .

1.2. If  $n = 3$  then  $R_1$  will be adjacent to  $R_2$  and  $R_3$ .

1.3. If  $n > 3$  then  $R_1$  will be adjacent to  $R_2, R_3, R_4$  and  $R_5$ .

2. *Obtaining adjacency with every next member*

Starting from  $R_2$ , each  $R_i$  will be adjacent to every  $R_{i+1}$  till  $i < n$ .

3. *Obtaining adjacency with every third next member*

Starting from  $R_2$ , each  $R_i$  will be adjacent to every  $R_{i+3}$  till  $i < (n - 2)$ .

4. *Obtaining adjacency with every fourth next member*

Starting from  $R_2$ , each  $R_i$  will be adjacent to every  $R_{i+4}$  till  $i < (n - 3)$ .

## 15. FUNCTION ADJACENCYHEIGHT( $p_3, p_4$ )

This function calculates the members (and their heights) of the first and second group (resp. fourth group) which can be adjacent to each other.

There are four cases for the number of members of a group, namely  $n = 1$ ,  $n = 2$ ,  $n = 3$  or  $n > 3$ . For  $n > 3$ , there are four sub-cases namely  $n \equiv 1 \pmod{4}$ ,  $n \equiv 2 \pmod{4}$ ,  $n \equiv 3 \pmod{4}$  and  $n \equiv 0 \pmod{4}$ . We represent all these cases using  $p_3$ .

We know that at most three members of the first group can be adjacent to at most three members of the second or the fourth group. We represent these members by  $r_1, r_2$  and  $r_3$  and their heights by  $s_1, s_2$  and  $s_3$ .

Here  $p_4 = 1$  stands for the members (and their heights) of the first group which can be adjacent to the members of the second group (resp. fourth group) when *spiral1*, *spiral2*, *spiral3* or *spiral4* (resp. *spiral5*, *spiral6*, *spiral7* or *spiral8*) is used for the first group.

$p_4 = 2$  represents the members (and their heights) of the first group which can be adjacent to the members of the second group (resp. fourth group) when *spiral5*, *spiral6*, *spiral7* or *spiral8* (resp. *spiral1*, *spiral2*, *spiral3* or *spiral4*) is used for the first group.

$p_4 = 3$  corresponds to the members (and their heights) of the second group (resp. fourth group) which can be adjacent to the members of the first group when *spiral5*, *spiral6*, *spiral7* or *spiral8* (resp. *spiral1*, *spiral2*, *spiral3* or *spiral4*) is used for the second group (resp. fourth group).

$p_4 = 4$  symbolizes the members (and their heights) of the second group (resp. fourth group) which can be adjacent to the members of the first group when *spiral1*, *spiral2*, *spiral3* or *spiral4* (resp. *spiral5*, *spiral6*, *spiral7* or *spiral8*) is used for the second group (resp. fourth group).

Let  $e_1, e_2$  and  $e_3$  are the members of the first group and  $g_1, g_2$  and  $g_3$  represent their heights respectively. If three members of a group (which can be adjacent to other members of any other group) are drawn one above the other such that  $e_3$  at top,  $e_2$  in middle and  $e_1$  at bottom. Let  $f_1, f_2$  and  $f_3$  represents members of the second or the fourth group and  $h_1, h_2$  and  $h_3$  represents their height respectively. Similarly if  $f_1, f_2$  and  $f_3$  are drawn one above the other then we have  $f_3$  at top,  $f_2$  in middle and  $f_1$  at bottom. For example, in Figure 2 for the left  $R_S^A, R_5, R_8, R_3$  are  $f_3, f_2, f_1$  respectively.

1. Set  $p_1 = p_3 - 1$ .

2. If  $p_1 = 1$  (here  $n = 1$ )

In this case, only one member of the first group can be adjacent to one member of the second or the fourth group.

Here  $r_1$  is  $R_1$  and  $s_1 = h_1$ . If  $p_4 = 1$  or  $2$ , we have  $e_1 = r_1$ ,  $g_1 = s_1$ . If  $p_4 = 3$  or  $4$ , we have  $f_1 = r_1$ ,  $h_1 = s_1$ .

3. If  $p_1 = 2$

3.1 If  $n = 1$ , this step is same as Step 2.

3.2 If  $n = 2$  we have  $r_2$  is  $R_1$  with  $s_2 = h_1$ ,  $r_1$  is  $R_2$  with  $s_2 = h_1$ .

If  $p_4 = 1$  we have  $e_1 = r_1$ ,  $e_2 = r_2$ ,  $g_1 = s_1$ ,  $g_2 = s_2$ .

If  $p_4 = 2$  we have  $e_1 = r_2$ ,  $e_2 = r_1$ ,  $g_1 = s_2$ ,  $g_2 = s_1$ .

If  $p_4 = 3$  we have  $f_1 = r_1$ ,  $f_2 = r_2$ ,  $h_1 = s_1$ ,  $h_2 = s_2$ .

If  $p_4 = 4$  we have  $f_1 = r_2$ ,  $f_2 = r_1$ ,  $h_1 = s_2$ ,  $h_2 = s_1$ .

4. If  $p_1 = 3$

4.1 If  $n = 1$ , this step is same as Step 2.

4.2 If  $n = 2$  we have  $r_1$  as the first member,  $s_1 = H_1$ . The cases for  $p_4 = i$ ,  $i = 1, \dots, 4$  are same as in Step 2.

4.3 If  $n = 3$  we have  $r_2$  as the first member,  $s_2 = H_1$ ,  $r_1$  as the third member,  $s_2 = h_3$ . The cases for  $p_4 = i$ ,  $i = 1, \dots, 4$  are same as in Step 3.2.

5. If  $n > p_1$

5.1 If  $n \equiv p_3 \pmod{4}$  ( $(n \text{ congruent to } p_3 \text{ modulo } 4)$ ) then  $r_1$  is  $R_n$ ,  $s_1 = H_n$ . The cases for  $p_4 = i$ ,  $i = 1, \dots, 4$  are same as in Step 2.

5.2 If  $n \equiv p_3 + 1 \pmod{4}$  then  $r_2$  is  $R_n$ ,  $s_2 = h_n$ ,  $r_1$  is  $R_{n-1}$ ,  $s_1 = H_{n-1}$ . The cases for  $p_4 = i$ ,  $i = 1, \dots, 4$  are same as in Step 3.2.

5.3 If  $n \equiv p_3 + 2 \pmod{4}$  then  $r_2$  is  $R_{n-1}$ ,  $s_2 = h_{n-1}$ ,  $r_1$  is  $R_{n-2}$ ,  $s_1 = H_{n-2}$ . The cases for  $p_4 = i$ ,  $i = 1, \dots, 4$  are same as given in Step 3.2.

5.4 If  $n \equiv p_3 + 3 \pmod{4}$  then  $r_3$  is  $R_{n-2}$ ,  $s_3 = h_{n-2}$ ,  $r_2$  is  $R_{n-3}$ ,  $s_2 = H_{n-3}$  and  $r_1$  is  $R_n$ ,  $s_1 = h_n$ .

If  $p_4 = 1$  we have  $e_1 = r_1$ ,  $e_2 = r_2$ ,  $e_3 = r_3$ ,  $g_1 = s_1$ ,  $g_2 = s_2$ ,  $g_3 = s_3$ .

If  $p_4 = 2$  we have  $e_1 = r_3$ ,  $e_2 = r_2$ ,  $e_3 = r_1$ ,  $g_1 = s_3$ ,  $g_2 = s_2$ ,  $g_3 = s_1$ .

If  $p_4 = 3$  we have  $f_1 = r_1$ ,  $f_2 = r_2$ ,  $f_3 = r_3$ ,  $h_1 = s_1$ ,  $h_2 = s_2$ ,  $h_3 = s_3$ .

If  $p_4 = 4$  we have  $f_1 = r_3$ ,  $f_2 = r_2$ ,  $f_3 = r_1$ ,  $h_1 = s_3$ ,  $h_2 = s_2$ ,  $h_3 = s_1$ .

6. If  $p_4 = 1$  or  $p_4 = 2$  we have  $e_4 = n$ . If  $p_4 = 3$  or  $p_4 = 4$  we have  $f_4 = n$ . The values of  $e_4$  and  $f_4$  will be used in the upcoming functions.

**Example 1.** Refer to Figure 2 where the members of the first group can be adjacent to the members of the second group, we have  $p_4 = 2$  and  $p_3 = 4$ . From Step 5.2, we have  $r_1 = R_2$ ,  $r_2 = R_{15}$ , hence  $e_1 = R_{15}$ ,  $e_2 = R_2$ .

Before moving to the function `findingadjacencies( $p_1, p_2$ )`, consider function

`adjacency( $p_1, p_2, p_3, p_4, p_5, p_6, p_7, p_8, p_9$ )` which is going to be used in the function `findingadjacencies( $p_1, p_2$ )`.

At most three members of the first group can be adjacent to at most three members of another group, therefore there are nine possibilities to be considered for computing adjacency among the members of different groups. All these nine possibilities are given in the function `adjacency( $p_1, p_2, p_3, p_4, p_5, p_6, p_7, p_8, p_9$ )`.

#### 16. FUNCTION `ADJACENCY( $p_1, p_2, p_3, p_4, p_5, p_6, p_7, p_8, p_9$ )`

If  $p_1 = 1$ ,  $p_2 = 1$ ,  $p_3 = 1$ , we consider  $e_1$  is adjacent to  $f_1$ ,  $f_2$ ,  $f_3$  respectively.

If  $p_4 = 1$ ,  $p_5 = 1$ ,  $p_6 = 1$ , we consider  $e_2$  is adjacent to  $f_1$ ,  $f_2$ ,  $f_3$  respectively.

If  $p_7 = 1$ ,  $p_8 = 1$ ,  $p_9 = 1$ , we consider  $e_3$  is adjacent to  $f_1$ ,  $f_2$ ,  $f_3$  respectively.

As said before 1, 2 or 3 members of the first group can be adjacent to 1, 2 or 3 members of any other group. These possibilities are expressed by following functions:

$adjacentrects11()$ ,  $adjacentrects21()$ ,  $adjacentrects31()$ ,  $adjacentrects12()$ ,  $adjacentrects22()$ ,  $adjacentrects32()$ ,  $adjacentrects13()$ ,  $adjacentrects23()$ ,  $adjacentrects33()$ .

For example function  $adjacentrects31()$  represents that only one member of the first group can be adjacent to at most three members of any other group. For these functions, adjacency pairs are obtained by comparing the length or height of the members of corresponding groups.

Here it is not possible to go through all these nine functions, therefore we consider only one of them. For an illustration, we discuss the steps of function  $adjacentrects22()$ .

## 17. FUNCTION ADJACENTRECTS22()

1. If  $(h_2 + h_1) \leq g_2$  then  $f_1, f_2$  is adjacent to  $e_2$  and we call function  $adjacency(0, 0, 0, 1, 1, 0, 0, 0, 0)$  to obtain adjacency pairs  $(f_1, e_2)$  and  $(f_2, e_2)$ .

2. If  $h_2 < g_2$  and  $h_1 > (g_2 - h_2)$  then  $f_2$  is adjacent to  $e_2$  and  $f_1$  is adjacent to  $e_1$ ,  $e_2$ . Here we call function  $adjacency(1, 0, 0, 1, 1, 0, 0, 0, 0)$  to obtain corresponding adjacency pairs.

3. If  $h_2 > g_2$  and  $h_2 < (g_2 + g_1)$  then  $f_2$  is adjacent to  $e_1, e_2$  and  $f_1$  is adjacent to  $e_1$ . Therefore, we call function  $adjacency(1, 1, 0, 0, 1, 0, 0, 0, 0)$ .

4. If  $h_2 \geq (g_2 + g_1)$  then  $f_2$  is adjacent to  $e_1, e_2$  and we call function  $adjacency(0, 1, 0, 0, 1, 0, 0, 0, 0)$ .

5. If  $h_2 = g_2$  then  $f_2$  is adjacent to  $e_2$  and  $f_1$  is adjacent to  $e_1$ . Here we call function  $adjacency(1, 0, 0, 0, 1, 0, 0, 0, 0)$ .

Now we discuss function  $findingadjacency(p_1, p_2)$ .

From functions  $adjacencyLength(p_3, p_4)$  or  $adjacencyHeight(p_3, p_4)$ , there are four cases for the values of  $f_4$  and  $e_4$ , namely  $f_4 > i$  and  $e_4 > j$  where  $i = 0, \dots, 3$ ,  $j = 0, \dots, 3$ . For obtaining adjacency pairs, it is required to consider  $e_4$  and  $f_4$  together. Therefore, in total there are sixteen possibilities.

In function  $findingadjacency(p_1, p_2)$ ,  $p_1 = 2$  and  $p_2 = 1$  represents the case  $f_4 > 2$ ,  $e_4 > 1$ . These sixteen possibilities are divided into following four parts.

In the first part we consider  $f_4 \leq p_1$  and  $e_4 \leq p_2$  where the number of sub-cases is  $p_1 \times p_2$ . For example for  $f_4 \leq 2$  and  $e_4 \leq 1$ , we consider the sub-cases  $f_4 = 1$  and  $e_4 = 1$ ,  $f_4 = 2$  and  $e_4 = 1$ .

In the second part we consider  $f_4 > p_1$  and  $e_4 \leq p_2$  where the number of sub-cases is  $4 \times p_2$ . For example for  $f_4 > 2$  and  $e_4 \leq 1$ , we consider the sub-cases  $f_4 \equiv 1 \pmod{4}$  and  $e_4 = 1$ ,  $f_4 \equiv 2 \pmod{4}$  and  $e_4 = 1$ ,  $f_4 \equiv 3 \pmod{4}$  and  $e_4 = 1$ ,  $f_4 \equiv 0 \pmod{4}$  and  $e_4 = 1$ . In general, for this part we call function  $adjacentrects1(p_3, p_1)$ . For example for  $f_4 > 2$  and  $e_4 \leq 1$ , we have  $p_3 = 1$ ,  $p_1 = 2$ .

In the third part we consider  $f_4 \leq p_1$  and  $e_4 > p_2$  where the number of sub-cases is  $p_1 \times 4$ . In general, for this part we call function  $adjacentrects2(p_3, p_1)$ . For example for  $f_4 \leq 2$  and  $e_4 > 1$ , we have  $p_3 = 2$ ,  $p_1 = 1$ .

In the fourth part, we consider  $f_4 > p_1$  and  $e_4 > p_2$  where the number of sub-cases is  $4 \times 4 = 16$ .

It is not possible to go through all the sixteen cases, therefore for demonstration we discuss only one case.

Suppose the first and second group is drawn using *spiral1*. This is the case  $f_2 > 2$  and  $e_4 > 0$  which implies that  $p_1 = 2$  and  $p_2 = 0$ .

#### 18. FUNCTION FINDINGADJACENCY(2, 0)

For  $p_1 = 2$  and  $p_2 = 0$ , we first consider the case  $f_4 \leq 2$  and  $e_4 > 0$  and then consider the case  $f_4 > 2$  and  $e_4 > 0$ .

1.  $f_4 \leq 2$  and  $e_4 > 0$

If  $p_1 = 2$  and  $p_2 = 0$  we call function `adjacentrects2(2, 0)`. For this particular example, the function `adjacentrects2(2, 0)` has the following steps:

1.1 If  $f_4 = 1$  and  $e_4 \equiv 1 \pmod{4}$  we call function `adjacentrects11()`. As an example, function `adjacentrects22()` has already been discussed (see Section 17).

1.2 If  $f_4 = 1$  and ( $e_4 \equiv 2 \pmod{4}$  or  $e_4 \equiv 3 \pmod{4}$ ) we call function `adjacentrects21()`.

1.3 If  $f_4 = 1$  and  $e_4 \equiv 0 \pmod{4}$  we call function `adjacentrects31()`.

1.4 If  $f_4 = 2$  and  $e_4 \equiv 1 \pmod{4}$  we call function `adjacentrects12()`.

1.5 If  $f_4 = 2$  and ( $e_4 \equiv 2 \pmod{4}$  or  $e_4 \equiv 3 \pmod{4}$ ) we call function `adjacentrects22()`.

1.6 If  $f_4 = 2$  and  $e_4 \equiv 0 \pmod{4}$  we call function `adjacentrects32()`.

2.  $f_4 > 2$  and  $e_4 > 0$

2.1 If ( $f_4 \equiv 0 \pmod{4}$  or  $f_4 \equiv 1 \pmod{4}$ ) and  $e_4 \equiv 1 \pmod{4}$  we call function `adjacentrects12()`.

2.2 If ( $f_4 \equiv 0 \pmod{4}$  or  $f_4 \equiv 1 \pmod{4}$ ) and ( $e_4 \equiv 2 \pmod{4}$  or  $e_4 \equiv 3 \pmod{4}$ ), we call function `adjacentrects22()`.

2.3 If ( $f_4 \equiv 0 \pmod{4}$  or  $f_4 \equiv 1 \pmod{4}$ ) and  $e_4 \equiv 0 \pmod{4}$  we call function `adjacentrects32()`.

2.4 If  $f_4 \equiv 2 \pmod{4}$  and  $e_4 \equiv 1 \pmod{4}$  we call function `adjacentrects13()`.

2.5 If  $f_4 \equiv 2 \pmod{4}$  and ( $e_4 \equiv 2 \pmod{4}$  or  $e_4 \equiv 3 \pmod{4}$ ) we call function `adjacentrects23()`.

2.6 If  $f_4 \equiv 2 \pmod{4}$  and  $e_4 \equiv 0 \pmod{4}$  we call function `adjacentrects33()`.

2.7 If  $f_4 \equiv 3 \pmod{4}$  and  $e_4 \equiv 1 \pmod{4}$  we call function `adjacentrects11()`.

2.8 If  $f_4 \equiv 3 \pmod{4}$  and ( $e_4 \equiv 2 \pmod{4}$  or  $e_4 \equiv 3 \pmod{4}$ ) we call function `adjacentrects21()`.

2.9 If  $f_4 \equiv 3 \pmod{4}$  and  $e_4 \equiv 0 \pmod{4}$  we call function `adjacentrects31()`.

#### 19. FLOYD'S ALGORITHM

This algorithm (cf. Pemmaraju and Skiena [6], Chapter 8) is used to obtain the distance and a shortest path between any two vertices.

The algorithm works by updating two matrices,  $D_k$  and  $Q_k$ ,  $n$  times for an  $n$ -vertex graph. The matrix  $D_k$ , in any iteration  $k$ , gives the value of the shortest distance among all pairs of vertices  $(i, j)$  as obtained till the  $k^{th}$  iteration. The matrix  $Q_k$  has  $q_{ij}^k$  as its elements. The value of  $q_{ij}^k$  gives the immediate predecessor vertex from vertex  $i$  to vertex  $j$  on the shortest path as determined by the  $k^{th}$  iteration. The starting matrix  $D_0$ , with entries  $d_{ij}^0$ , is defined as follows:

$d_{ij}^0 = 1$  if  $i \neq j$  and vertex  $i$  is adjacent to vertex  $j$

$d_{ij}^0 = \infty$  if  $i \neq j$  and vertex  $i$  is not adjacent to vertex  $j$

$$d_{ij}^0 = 0 \text{ if } i = j$$

The entries  $q_{ij}^0$  of the predecessor matrix  $Q_0$  are defined as follows:  $q_{ij}^0 = i$ , for  $i \neq j$ , i.e., for every pair of distinct vertices  $(i, j)$ , the immediate predecessor of vertex  $j$  on a shortest path leading from vertex  $i$  to vertex  $j$  is (temporarily) assumed to be vertex  $i$ . After defining  $D_0$  and  $Q_0$  the following steps are used repeatedly to determine  $D_n$  and  $Q_n$ .

Step 1 : Set  $k = 1$

Step 2 : The entries  $d_{ij}^k$  of the shortest path matrix  $D_k$  are defined by:

$$d_{ij}^k = \min(d_{ij}^{k-1}, d_{ik}^{k-1} + d_{kj}^{k-1})$$

Step 3: The entries  $q_{ij}^k$  of the predecessor matrix  $Q_k$  are defined as follows:

If  $d_{ij}^k \neq d_{ij}^{k-1}$  then  $q_{ij}^k = q_{kj}^{k-1}$  else  $q_{ij}^k = q_{ij}^{k-1}$ .

Step 4: If  $k = n$ , the algorithm is terminated. If  $k < n$ , increase  $k$  by 1, and return to step 2.

Now we take a look at the algorithm in a little more detail. In step 2, each time one goes through the algorithm, it is checked whether a shorter path exists between vertex  $i$  and vertex  $j$ . In step 3, if it is established that  $d_{ij}^k \neq d_{ij}^{k-1}$ , i.e., the length of the shortest path  $d_{ij}^k$  between vertices  $i$  and  $j$  is less than the length of the shortest path  $d_{ij}^{k-1}$ , it is required to change the immediate predecessor vertex to vertex  $j$ . Since the length of the new shortest path is:

$$d_{ij}^k = d_{ik}^{k-1} + d_{kj}^{k-1}$$

it is clear that here node  $k$  is the new immediate predecessor vertex to  $j$ , and therefore:

$$q_{ij}^k = q_{kj}^{k-1}$$

After passing through the algorithm  $n$  times, the entries  $d_{ij}^n$  of the final matrix  $D_n$  will constitute a shortest path going from vertex  $i$  to vertex  $j$ .

Matrix  $Q$  gives the immediate predecessor vertex to vertex  $j$  on the shortest path. To have all vertices of the shortest path between vertex  $i$  and  $j$ , starting from vertex  $j$  obtain the immediate predecessor one by one till vertex  $i$ .

The obtained shortest path is of course not unique in general.

## REFERENCES

- [1] [www.processing.org](http://www.processing.org)
- [2] Terzidis, K., Algorithms For Visual Design (Using the Processing Language), Wiley Publishing, Inc, Indianapolis, 2009.
- [3] Shekhawat, K., Automated space allocation using mathematical techniques, Ain Shams Engineering Journal, Elsevier, ASEJ402, 2015.
- [4] Shekhawat, K., Algorithm for constructing an optimally connected rectangular floor plan, Frontiers of architectural research, 3, 324-330, 2014.
- [5] Gross, J. L. and Yellen, J., Graph Theory and Its Applications (Second Edition), Chapman & Hall/CRC, Boca Raton, 2006.
- [6] Pemmaraju, S. and Skiena, S., Computational Discrete Mathematics (Combinatorics and Graph Theory with Mathematica), Cambridge University Press, 2003.